FACULDADE CATÓLICA SALESIANA DO ESPÍRITO SANTO

JOHN FREITAS

FRANKLIN DINIZ

SEGURANÇA E PERSISTÊNCIA:
UMA ANÁLISE DAS ARQUITETURAS DE DESENVOLVIMENTO DE SOFTWARE NAS PLATAFORMAS ANDROID E WINDOWS PHONE

VITÓRIA

2013

JOHN FREITAS

FRANKLIN DINIZ

SEGURANÇA E PERSISTÊNCIA:

UMA ANÁLISE DAS ARQUITETURAS DE

DESENVOLVIMENTO DE SOFTWARE

NAS PLATAFORMAS ANDROID E WINDOWS PHONE

Trabalho de Conclusão de Curso apresentado à Faculdade Católica Salesiana do Espírito Santo, como requisito obrigatório para obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas.

Orientador: Prof. Maurício Castro.

VITÓRIA

2013

JOHN FREITAS

FRANKLIN DINIZ

SEGURANÇA E PERSISTÊNCIA:
UMA ANÁLISE DAS ARQUITETURAS DE DESENVOLVIMENTO DE SOFTWARE NAS PLATAFORMAS ANDROID E WINDOWS PHONE

Trabalho de Conclusão de Curso apresentado à Faculdade Católica Salesiana do Espírito Santo, como requisito obrigatório para obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas.

Aprovado em ___ de _____de 2013, por:

_____

Prof. Maurício Castro - Orientador

_____

Prof. Rodrigo Sarmento

_____

Prof. Rostan Piccoli

Para nossos pais, por seu apoio incondicional; nossas companheiras por sua paciência e carinhos imensos e ao professor Maurício Barbosa e Castro por acreditar no nosso potencial mesmo quando nós duvidamos.

Não é nossa função controlar todas as marés do mundo, mas sim fazer o que pudermos para socorrer os tempos em que estamos inseridos, erradicando o mal dos campos que conhecemos, para que aqueles que viverem depois tenham terra limpa para cultivar. Que tempo encontrarão não é nossa função determinar.

John Ronald Reuel Tolkien (1892 - 1973) – O Senhor dos Anéis.

AGRADECIMENTOS

Ao nosso orientador, professor Maurício Barbosa e Castro, que nos apoiou em todos os momentos, desde a concepção da atual estrutura do trabalho até as conversas esclarecedoras que abriram nossas mentes para a concretização deste trabalho.

A professora Ludimila Monjardim Casagrande, que nos orientou e nos ajudou a definir o rumo inicialmente planejado para este trabalho.

Ao professor André Cipriano Costa por compartilhar do material de pesquisa necessário para o desenvolvimento deste estudo.

Aos nossos pais, irmãos e irmãs.

Às nossas companheiras.

Este é um sonho sendo realizado e um objetivo alcançado. Apenas um passo em direção a um futuro de muitos estudos e sucesso.

RESUMO

Este trabalho apresenta os conceitos chave necessários para o desenvolvimento de aplicativos móveis e tem como foco principal os aspectos de Segurança e Persistência nas plataformas de desenvolvimento móvel *Android* e *Windows Phone*. Devido à grande popularização dos dispositivos móveis, surge um ambiente propício aos desenvolvedores, um setor em grande expansão e torna-se necessário ao desenvolvedor a obtenção de conhecimentos relevantes ao desenvolvimento de aplicativos voltados a estas plataformas levando em consideração suas características limitadas de *hardware* e processamento. São apresentados neste estudo de forma clara e concisa detalhes relevantes sobre características supracitadas dos mesmos. Ainda apresenta-se um estudo analítico visando fornecer ao desenvolvedor a base inicial necessária para a escolha entre as plataformas com o objetivo de atender os requisitos do aplicativo móvel a ser desenvolvido. Ao ler este estudo o desenvolvedor contará com o referencial teórico necessário para tirar suas próprias conclusões, realizar uma escolha consciente da plataforma que mais se encaixa em suas necessidades e dar início aos seus estudos e ao desenvolvimento de aplicativos móveis.

Palavras-chave: *Android*. *Windows Phone*. Desenvolvimento Móvel. Segurança. Persistência.

ABSTRACT

This paper presents the key concepts necessary for developing mobile applications and is mainly focused on the aspects of Security and Persistence in developing mobile applications for the Android and Windows Phone platforms. Due to the popularity of mobile devices, there is a favorable environment for developers, a sector booming and it becomes necessary for developers to obtain knowledge relevant to application development focused on these platforms considering its feature limited hardware and processing. This study presents in clear and concise detail about relevant characteristics quoted above. Also presents an analytical study to provide the developer with an initial knowledge base needed when choosing platforms in order to meet the requirements of mobile application development. By reading this study, the developer will have the theoretical background needed to draw their own conclusions, make a conscious choice of the platform that best fits their needs and start studies and the development of mobile applications.

Keywords: Android. Windows Phone. Mobile Development. Security. Persistency.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

IDC – *International Data Corporation*

SDK – *Software Development Kit*

GNU – GNU *is Not* Unix

OHA – *Open Headset Alliance*

XML – *Extensible Markup Language*

POSIX – *Portable Operating System Interface*

TCB – *Trusted Computing Base*

ERC – *Elevated Rights Chamber*

SRC – *Standard Rights Chamber*

LPC – *Least Privileged Chamber*

API – *Application Programming Interface*

VOIP – *Voice over Internet Protocol*

PC – *Personal Computer*

UVC – *Universal Volume Control*

PDA – Personal Digital Assistant

UID – *User-Id*

LINQ – Language INtegrated Query

SQL – Structured Query Language

SUMÁRIO

1 INTRODUÇÃO

Esta é uma época de muitas mudanças e rápida evolução tecnológica, cada vez se faz mais presente em nosso cotidiano o uso de dispositivos móveis, a produção e consumo de informação através dos mesmos.

> O crescimento extraordinário que tem ocorrido nesta década nas áreas de comunicação celular, redes locais sem fio e serviços via satélite permitirão que informações e recursos possam ser acessados e utilizados em qualquer lugar e em qualquer momento. Dado o atual crescimento do segmento de computadores pessoais e PDAs, (*Personal Digital Assistants*) estima-se que em poucos anos, dezenas de milhões de pessoas terão um *laptop*, *palmtop* ou algum tipo de PDA (MATEUS, 1998, p. 01).

A utilização de aparelhos portáteis já se tornou parte da cultura global, seja em fotos, vídeos ou compartilhamentos em redes sociais, os indivíduos se conectam através de seus dispositivos móveis para interagir com todo o globo.

> As práticas contemporâneas ligadas às tecnologias da *ciber* cultura têm configurado a cultura contemporânea como uma cultura da mobilidade. Vários autores mostraram como as sociedades contemporâneas estão imersas em um processo de territorializações e desterritorializações sucessivas (Deleuze e Guattari, 1986), de práticas nômades e tribais, tanto em termos de subjetividade como de deslocamentos e afinidades (Maffesoli, 1997); de reconfiguração dos espaços urbanos (Mitchell, 2003; Horan, 2000; Meyrowitz, 2004) e de constituição de uma sociologia da mobilidade (Urry, 2000; Urry, 2003, Cooper, Green, Murtagh, Harper, 2002). No que se refere às novas tecnologias em interface com o espaço público, a ideia de mobilidade é central para conhecer as novas características das cidades contemporâneas (LEMOS, 2004, p. 41).

A imagem a seguir mostra a grande utilização de smartphones nos dias atuais.

Figura 1 – Popularização dos dispositivos móveis

Fonte: Exame. 2013

Segundo pesquisas do *International Data Corporation* (IDC) no segundo trimestre de 2013 foram vendidos 8,3 Milhões de *smartphones* no Brasil, e isso corresponde a um aumento de 110% em comparação com o mesmo período de 2012, desse estudo ainda pode se destacar que 90% dos aparelhos vendidos possuem o sistema móvel *Android*.

Uma pesquisa da empresa *Flurry*, que analisa o mercado móvel, revela que o Brasil duplicou seu número de *tablets* ativos entre abril de 2012 e abril de 2013 e que teve um crescimento de mercado que fica entre 100% e 199%. A pesquisa ainda aponta que outros países tiveram sua base móvel duplicada nesse período (EXAME, 2013).

A Agência de Notícias da Fundação de Amparo à Pesquisa de São Paulo (FAPESP) através de pesquisa divulgada em 2013 pelo Centro de Estudos sobre as Tecnologias da Informação e da Comunicação (CETIC) informa que o número de residências brasileiras que possuíam acesso à *internet* era de 36% em 2011 e ampliou-se para 40% em 2012, que o número de acessos à internet através de *lan houses* caiu para 19% em 2012. (FAPESP, 2013).

Em vista desses dados, nota se que há uma vasta área de mercado a ser explorada, por exemplo, a área de desenvolvimento de aplicativos para esses dispositivos. Este estudo se propõe a compilar os conceitos chave necessários para compreender este cenário e aprofundar o conhecimento técnico necessário para a escolha de uma plataforma móvel com o propósito de desenvolver aplicativos e usufruir de todo o potencial deste mercado em pleno crescimento.

Dentre as diversas plataformas móveis existentes atualmente algumas se destacam por sua grande parcela no mercado, dentre elas podemos citar as plataformas Android, iOS, Windows Phone, *Symbian*. Esse estudo mantém o foco apenas nas plataformas Android e Windows Phone devido a quantidade de informações disponíveis, dispositivos acessíveis e facilidade de compreensão.

Outro fato que impacta na escolha das plataformas Android e Windows Phone em detrimento às outras plataformas como, por exemplo, o iOS, é o fato de que essa segunda só funciona em hardware proprietário, o que dificulta o acesso dos pesquisadores ao material de pesquisa necessário.

A tabela a seguir apresenta dados de vendas de smartphones por empresas e SO, segundo a empresa de Pesquisa IDC em pesquisa realizada no terceiro semestre de 2012.

Tabela 1 - Tabela com dados de vendas por empresas e SO.

Operating System	3Q12 Shipment Volumes	3Q12 Market Share	3Q11 Shipment Volumes	3Q11 Market Share	Year Over Year Change
Android	136.0	75.0%	71.0	57.5%	91.5%
iOS	26.9	14.9%	17.1	13.8%	57.3%
BlackBerry	7.7	4.3%	11.8	9.5%	-34.7%
Symbian	4.1	2.3%	18.1	14.6%	-77.3%
Windows Phone 7 / Windows Mobile	3.6	2.0%	1.5	1.2%	140.0%
Linux	2.8	1.5%	4.1	3.3%	-31.7%
Others	0.0	0.0%	0.1	0.1%	-100.0%
Totals	181.1	100.0%	123.7	100.0%	46.4%

Fonte: Androidpit. 2012

O objetivo deste trabalho é analisar os aspectos de segurança e persistência da arquitetura de *software* para dispositivos móveis, tendo como foco o estudo das plataformas *Android* e *Windows Phone*. Através de uma revisão bibliográfica será realizada uma análise detalhada de cada aspecto anteriormente citado em cada uma dessas plataformas, abordando suas características, vantagens e desvantagens.

Para atingir este objetivo iremos investigar os aspectos (Segurança e Persistência) das plataformas *Android* e *Windows Phone*; realizar um estudo a fim de evidenciar ao leitor qual destas plataformas oferece melhores condições para o desenvolvimento de uma aplicação móvel e apresentar o conhecimento necessário para que o leitor tenha base para a escolha de uma das citadas plataformas na hora de iniciar o projeto de *software* para dispositivos móveis.

Este trabalho visa contribuir com conhecimentos relevantes para um desenvolvedor ao escolher uma plataforma de desenvolvimento para dispositivos móveis e fazer um aplicativo móvel com a plataforma escolhida. Como o embasamento teórico é fundamental para a obtenção de um conhecimento estrutural de modo a permitir a resolução de problemas, este trabalho se mostra relevante no sentido de guiar o desenvolvedor no caminho do conhecimento mais adequado à construção de um software móvel. Desta maneira, serão apresentados neste documento os principais conceitos para o desenvolvedor, além de um histórico das plataformas móveis Android e Windows Phone e uma análise dos aspectos de segurança e persistência em cada uma das mesmas para que o desenvolvedor possa com base nas análises efetuadas, formar a opinião necessária para o início do desenvolvimento.

2 REFERENCIAL TEÓRICO

Para os desenvolvedores que querem aprimorar seus conhecimentos na área de desenvolvimento para dispositivos móveis, deve se fazer uma análise do ambiente de computação móvel ao decidir por onde começar. Um dos percalços que podem ser encontrados quando falamos nos estudos sobre desenvolvimento para dispositivos móveis, são as plataformas ou sistemas operacionais para os mesmos. Pois como veremos no decorrer deste trabalho as plataformas se diferenciam nos métodos de construção de seus aplicativos e nos requisitos necessários para desenvolver os mesmos. Dessa maneira, a fim de evitar problemas quanto ao desenvolvimento e ao desempenho de execução dos aplicativos móveis é necessário conhecer a plataforma na qual o *software* será desenvolvido, além de se compreender os conceitos que permeiam as bases da computação móvel.

Um fator que leva desenvolvedores e curiosos a se aprofundar no ambiente de computação móvel, é o avanço tecnológico que torna a computação cada vez mais eficiente e necessária. Como referência ao início dessa evolução, por exemplo, temos a Lei de Moore, que estabelece um padrão para o avanço do processamento dos computadores.

> Em 1965, Gordon Moore, estabeleceu uma lei onde constatou que a cada dois anos aproximadamente, o número de transistores por unidade de área dobra e, consequentemente, o poder de processamento dos computadores. Esta lei ficou conhecida como Lei de Moore e vem se mantendo em vigor até os dias de hoje. Segundo esta lei, em 2020 a computação clássica atingirá seu limite, sendo possível a representação de um bit de informação em um átomo (CROTTI, 2008, p. 02).

O uso da computação cresceu ao ponto de ser necessário utilizar serviços e acessar informações com frequência, de qualquer lugar, mas com máquinas grandes e conectadas à rede fisicamente, não seria possível, os usuários ficariam sem mobilidade. “Uma rede de computadores é conexão de dois ou mais computadores para permitir o compartilhamento de recursos e a troca de informações entre as máquinas.” (CANTÚ, 2003, p. 06).

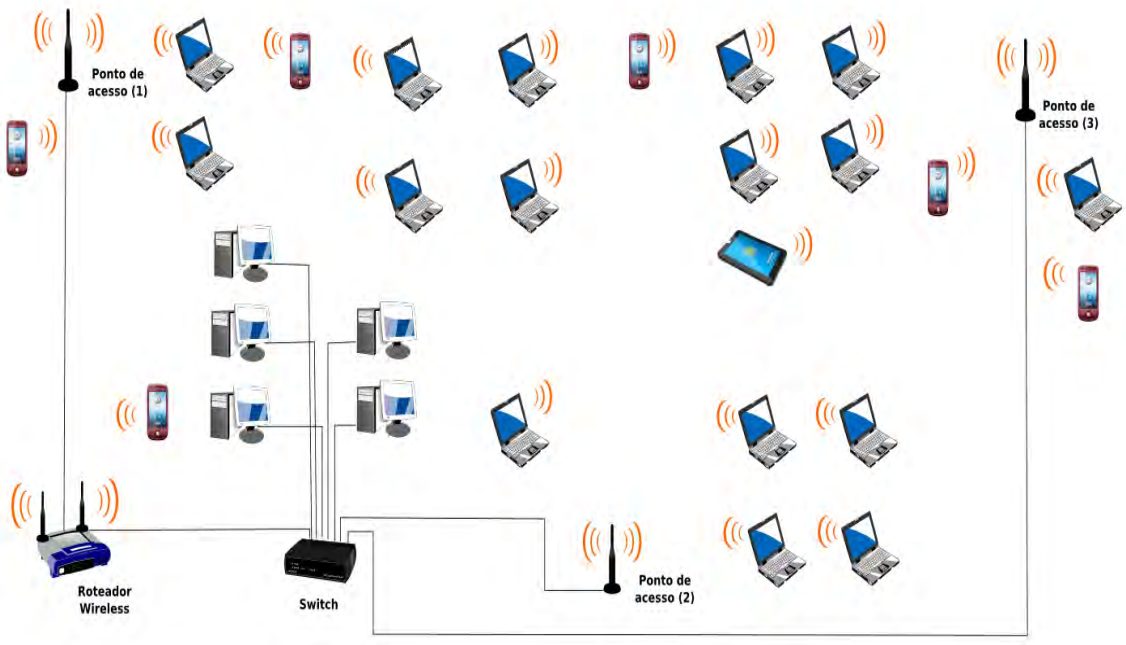
Uma revolução tecnológica que contribuiu com o surgimento da computação móvel, foi à criação das redes sem fio, de forma que não seria preciso estar conectado à rede fisicamente para ter acesso.

> Redes sem fio são redes de comunicação entre dispositivos, como por exemplo, computadores, telefones VOIP, impressoras, sem recurso à

> utilização de cabos para fazer a interligação desses dispositivos. Para estabelecer essa interligação são utilizadas radiofrequências ou infravermelhos. A sua utilização vai desde o uso de equipamentos de conversação como *walkie-talkies* até à utilização de satélites artificiais, sendo a sua utilização mais comum para redes de dados entre computadores pessoais, possibilitando dessa forma uma maior mobilidade e facilidade para aceder às redes de dados como, por exemplo, a *Internet* (CHAVES, 2009, p. 05).

A imagem a seguir demonstra através da topologia de rede a estrutura de uma rede *wireless*:

Figura 2 – Topologia de rede *wireless*



Fonte:Carlos E. Morimoto. 2011

Entretanto, o peso e o tamanho do *hardware* ainda eram problemas, tornando inviável locomover-se com eles, o que levou à necessidade de se desenvolver os dispositivos móveis (aparelhos pequenos, leves, portáteis e com acesso as redes sem fio).

> Vamos considerar que um dispositivo móvel é um equipamento digital que serve para realizar tarefas diárias de trabalho. Isso inclui todos os Pocket PCs, Palms, SmartPhones e relacionados. Além destes, outros dispositivos móveis que utilizam a plataforma Windows CE.NET.(KOVACS; MONTEIRO, 2000?, p. 04).

Figura 3 – Dispositivos móveis

Fonte: Techlider. 2013

Com o surgimento dos dispositivos móveis, sua capacidade de se conectar as redes sem fio e a possibilidade de se acessar dados a qualquer hora em praticamente qualquer lugar, criou o que entendemos pelo conceito de computação móvel.

> A computação móvel 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 ou computação nômade (TONIN, 2012, p. 03).

Possibilitada pelos dispositivos móveis a computação móvel amplia as fronteiras da computação levando-a a pontos antes inatingíveis, tornando-se parte do uso diário e do ambiente de trabalho. A portabilidade dos serviços computacionais expande a capacidade de utilização dos serviços que um computador oferece aos usuários, independentemente de sua localização.

A computação estando presente em todos os espaços de forma inteligente é entendida como computação Ubíqua, e surge da necessidade de integrar mobilidade com funcionalidade, enquanto em movimento junto ao usuário, o dispositivo é capaz de construir modelos computacionais dinamicamente, ou seja, configurar e ajustar o dispositivo de acordo com o ambiente e as necessidades do usuário.

> A Computação Ubíqua impulsiona a ideia de que os computadores estarão em todos os lugares e em todos os momentos auxiliando o ser humano sem que ele tenha consciência disso. Isto quer dizer que tais equipamentos se tornarão imperceptíveis não pelo seu tamanho, mas devido a sua capacidade de processar tais informações sem a intervenção de seu usuário sim de acordo com o ambiente (KAHL, 2011, p. 01).

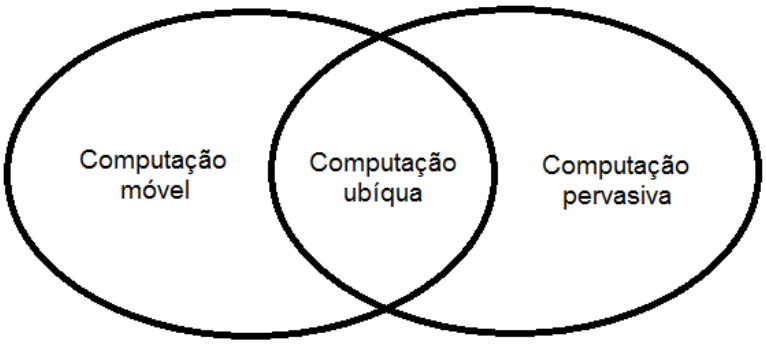
Desta forma, para compreender melhor todos os conceitos que permeiam o ambiente da computação móvel devemos analisar o conceito de computação pervasiva que surge para complementar e evoluir os conceitos já existentes de computação móvel e ubíqua.

A computação pervasiva surge da integração da mobilidade e da ubiquidade dos sistemas computacionais móveis com o intuito de desenvolver outros modelos computacionais dinamicamente, onde se fazem necessários sistemas mais adaptados e inteligentes, interagindo com o usuário de forma implícita.

A computação pervasiva interage com o ambiente, obtendo informações por si só, para processar e fornecer resultados, tornando-se o aparato computacional imperceptível, mas atendendo da mesma forma, ou melhor, as suas funções.

> O conceito de computação pervasiva implica que o computador está embarcado no ambiente de forma invisível para o usuário. Nesta concepção, o computador tem a capacidade de obter informação do ambiente no qual ele está embarcado e utilizá-la para dinamicamente construir modelos computacionais, ou seja, controlar, configurar e ajustar a aplicação para melhor atender as necessidades do dispositivo ou usuário (DE ARAUJO, 2003, p. 50).

Figura 4 – Relação entre computação Móvel, Ubíqua e Pervasiva



Fonte: Elaboração própria. 2013

2.1 SOFTAWARE E SISTEMA OPERACIONAIL

Tendo em vista que o uso dos dispositivos móveis se integra aos conceitos de computação móvel ubíqua e pervasiva, que para exercer suas funções é necessário um sistema operacional móvel atuando em conjunto com o *hardware* portátil através de softwares específicos para esse sistema (o que conhecemos como aplicativos móveis), e que sistema operacional móvel é um sistema operacional voltado a operação em dispositivos portáteis, faz se necessário a compreensão dos conceitos de software e sistema operacional.

Software é um conjunto lógico de instruções dispostas em passos e que serão interpretadas por um sistema computacional com o objetivo de executar tarefas específicas. De acordo com PRESSMAN a seguinte definição explica o que é um software:

> *Software* de computador é o produto que profissionais de *software* desenvolvem e ao qual dão suporte no longo prazo. Abrange programas executáveis em um computador de qualquer porte ou arquitetura, conteúdos (apresentados à medida que os programas são executados), informações descritivas tanto na forma impressa (*hard copy*) como na virtual, abrangendo praticamente qualquer mídia eletrônica (PRESSMAN, 1995, p. 29).

No sistema computacional a principal função do *software* é fornecer instruções para o *hardware*. Assim pode se separar os *softwares* de um sistema computacional (ou sistema operacional) em três tipos:

Software de sistema: Conjunto de instruções lógicas que são processadas internamente pelo sistema computacional com o objetivo de criar a interação entre o usuário e os dispositivos de entrada e saída do computador através de interface. Pode se citar nesta categoria de *softwares* os *drivers* dos dispositivos periféricos de um computador.

Software de programação: Conjunto de ferramentas (*softwares*) que permitem ao desenvolvedor criar outros *softwares* através das linguagens de programação dentro de um único sistema. Um exemplo são as IDE's (*Integrated Development Environment*) que são ambientes controlados e que oferecem todos os recursos necessários para o desenvolvedor.

Software de aplicação: São os programas criados pelos desenvolvedores através dos *softwares* de programação e que podem ser executados em computadores ou outros dispositivos que possuam poder de processamento. Esses programas tem o objetivo de interagir com o usuário de forma a receber dados, processar e devolver outras informações. Dentre estes, podemos citar jogos eletrônicos, editores de texto e imagens, *softwares* de execução de vídeos e músicas.

A dualidade na natureza do *software* fornece a compreensão de sua importância nos sistemas computacionais, visto que ele é, tanto um programa quanto pode originar outros programas, de acordo com o que foi programado para fazer. PRESSMAN evidencia da seguinte forma a natureza do *software*:

> Hoje, o *software* assume um duplo papel. Ele é um produto e ao mesmo tempo o veículo para distribuir um produto. Como produto, fornece o potencial computacional representado pelo *hardware* ou, de forma mais abrangente, por uma rede de computadores que podem ser acessados por *hardware* local. Independentemente de residir em um celular ou operar dentro de um *mainframe*, *software* é um transformador de informações - produzindo, gerenciando, adquirindo, modificando, exibindo ou transmitindo informações que podem ser tão simples quanto um único *bit* ou tão complexas quanto uma apresentação multimídia derivada de dados obtidos de dezenas de fontes independentes. Como veículo de distribuição do produto, o *software* atua como a base para o controle do computador (sistemas operacionais), a comunicação de informações (redes) e a criação e o controle de outros programas (ferramentas de *softwares* e ambientes) (PRESSMAN, 1995, p. 31).

Já o sistema operacional é um conjunto de *softwares* que operam em modo núcleo e que trabalham de forma intermediária entre *softwares* de usuários destinados a solucionar problemas lógicos e a camada de *hardware*.

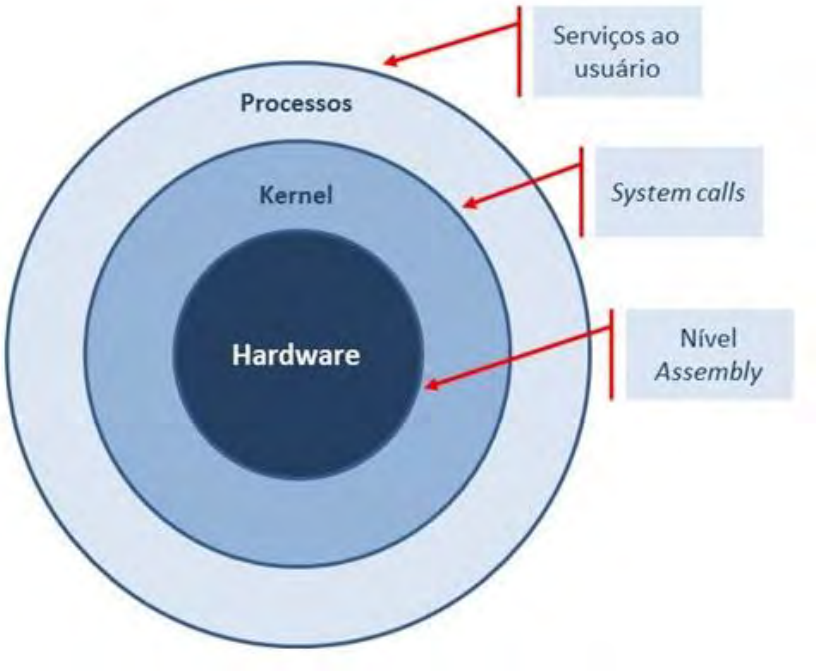
> É difícil definir o que é um sistema operacional além de dizer que é o *software* que executa em modo núcleo e mesmo nem sempre isso é verdade. Parte do problema ocorre porque os sistemas operacionais realizam basicamente duas funções não relacionadas: fornecer aos programadores de aplicativos (e aos programas aplicativos, naturalmente) um conjunto de recursos abstratos claros em vez de recursos confusos de *hardware* e gerenciar esses recursos de *hardware*. Dependendo do tipo de usuário, ele vai lidar mais com uma função ou com outra (TENEMBAUM, 2009, p. 02).

De acordo com DEITEL o conceito de sistemas operacionais pode ser definido da seguinte forma:

> Vemos um sistema operacional como os programas, implementados como *software* ou *firmware*, que tornam o *hardware* utilizável. O *hardware* oferece capacidade computacional bruta. Os sistemas operacionais disponibilizam convenientemente tais capacidades aos usuários, gerenciando

> cuidadosamente o *hardware* para que se obtenha um desempenho adequado (DEITEL, 1992, p. 03).

Figura 5 – Representação do sistema operacional



Fonte: J. R. Smolka. 2011

2.2 PLATAFORMAS DE DESENVOLVIMENTO MÓVEL

Após analisar os conceitos de software e sistema operacional é necessário que o desenvolvedor compreenda o ambiente móvel como um todo, sendo assim, podemos adentrar nos conceitos de plataforma móvel e plataforma de desenvolvimento móvel.

Plataforma móvel é o sistema operacional móvel que em geral é embarcado em aparelhos portáteis e no qual são disponibilizados programas computacionais, os aplicativos, que desempenham as mais variadas funcionalidades, desde enviar uma mensagem ou efetuar uma ligação até executar um procedimento de compras pela *internet*.

> Para a utilização de todo o potencial proporcionado pelos *smartphones* existe a necessidade de sistemas operacionais, que neste caso também são chamados de plataforma já que com raras exceções podem ser substituídos e são programados com adaptações específicas para cada modelo de celular (RODRIGUES, 2009, p. 65).

A plataforma de desenvolvimento móvel é um conjunto estruturado de ferramentas e *softwares* que dão todo o suporte ao desenvolvedor para a criação de novas aplicações e funcionalidades para os sistemas operacionais móveis.

> As plataformas de desenvolvimento para celulares são ambientes, ferramentas, técnicas, e linguagens de programação que possibilitam a codificação, simulação (ou emulação), depuração e teste de programas voltados para aparelhos celulares habilitados para tal (BOURBON, 2005, p. 02).

As plataformas de desenvolvimento móvel também são conhecidas como *Software Development Kit* (SDK) e seguem uma série de padrões estabelecidos pelas empresas desenvolvedoras dos sistemas operacionais móveis. Dentre as atuais plataformas móveis disponíveis no mercado, podemos citar duas: a plataforma *Android* e a plataforma *Windows Phone*. Kamada (2012) discorre sobre o *Windows Phone*:

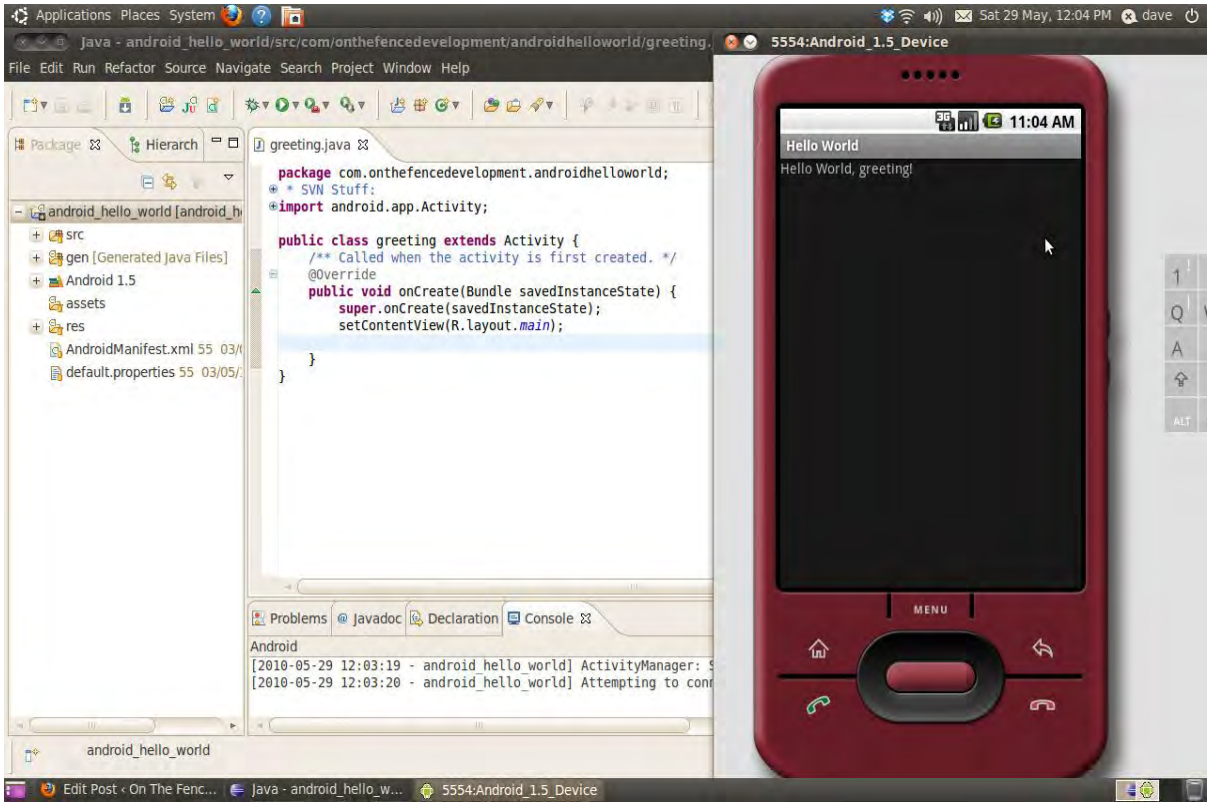
> Plataforma SDK, utilizada por desenvolvedores.NET (MSDN, 2012c), é uma ferramenta que permite a criação de aplicativos utilizando as tecnologias Microsoft. Ela é utilizada para projetar e criar aplicativos para *Windows Phone*. Esta ferramenta gera código para *Framework*.NET e suporta linguagens *Visual Basic* .NET, C#, C++, e J#. Trabalha com tecnologias como *Visual Studio*, *Silverlight* e .NET, WPF, *Silverlight* e XAML. Como processo de melhoria da plataforma, em 2011, surge a versão com o codinome "*Mango*", com mais de 500 recursos, destaque de melhoramento como o suporte a multitarefa, atualização da versão do *Silverlight* para o *Silverlight*4 e atualização do *Browser* para a versão do *Internet Explorer* 9 com suporte ao HTML5, mudanças tanto para o desenvolvedor como para o usuário. As melhorias incluem uma melhor integração com o *Skydrive*, com as funções de compartilhamento de fotos e de vídeos, com "*Upload*" em *background*, para o arquivamento nas nuvens.

Sobre a plataforma de desenvolvimento citamos Android (2012):

> O Eclipse (IDE) é uma ferramenta código aberto que agrega à plataforma de desenvolvimento os geradores de códigos Java, e por meio de extensões origina códigos em *Python* e C/C++. O Eclipse começou como tecnologia proprietária com o lançamento entre1999 e 2001 da versão 1.0, anunciado pela empresa americana IBM (*International Business Machines*), no valor de US$ 40 milhões.

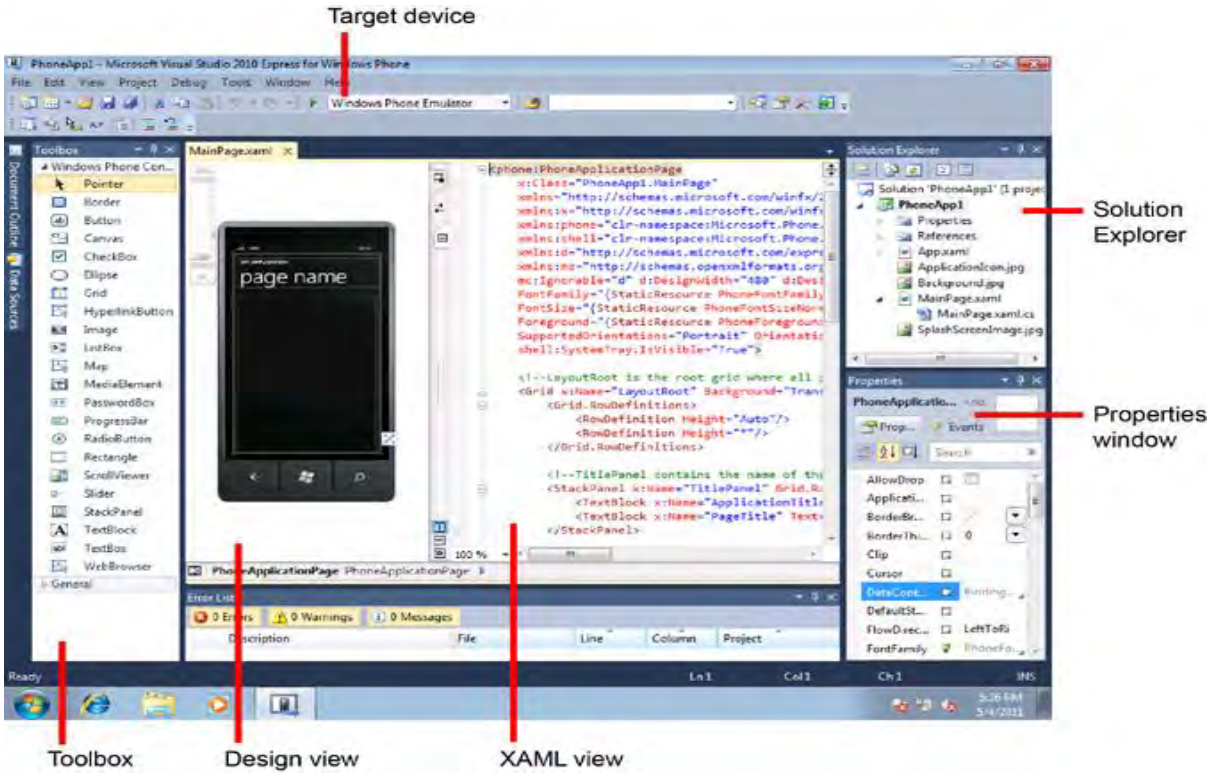
As imagens a seguir demonstram as plataformas de desenvolvimento *Android* e *Windows Phone* respectivamente:

Figura 6 – Plataforma de desenvolvimento *Android*



Fonte: Dave Carson. 2010

Figura 7 – Plataforma de desenvolvimento *Windows Phone*



Fonte: Msdn *Microsoft*. 2010

Depois de se conhecer os conceitos de plataforma móvel e plataforma de desenvolvimento móvel, deve se fazer uma retrospectiva a fim de conhecer a história das duas plataformas móveis escolhidas para este estudo. A seguir, serão mostrados os históricos das plataformas Android e Windows Phone.

2.2.1 Histórico da plataforma Windows Phone

Apresentado em 1994 o *Windows* CE foi concebido para ser utilizado nos primeiros dispositivos móveis, conhecidos com *Handhelds*. Eram aparelhos com telas sensíveis ao toque, monocromáticas que serviam como opções leves e baratas comparadas aos *notebooks* e *desktops*, permitindo a edição de documentos e execução de tarefas simples. Esses dispositivos só podiam ser conectados a *internet* por meio de *modems* discados através de linhas telefônicas, pois não existiam redes sem fio.

O *Windows* CE possuía interface semelhante a uma versão simplificada do *Windows* 95/98 e apresentava algumas aplicações, dentre elas as versões *pocket* do *Word*, *Excel* e *Internet Explorer*.

Após a *Palm*, empresa que lançou seu dispositivo móvel *Palm Pilot* em março de 1997, com formato de organizador pessoal, a *Microsoft* decidiu adaptar-se a esse novo formato de dispositivo, deixando o formato de mini-*notebook* que utilizava antes.

A seguir imagem de um dos primeiros dispositivos móveis, o *Palm Pilot*:

Figura 8 – *Palm Pilot*

Fonte: Donald Melanson. 2011

Com o lançamento do *Pocket* PC 2000, foi apresentada uma versão modificada do *Windows* CE 3.0 que contava com o *Design* de organizador pessoal. A partir de então o *Windows* CE continuou sendo desenvolvido para dispositivos embarcados[1] e o *Pocket* PC (precursor do *Windows Mobile*) passou a ser uma plataforma baseada nele, com aplicativos e drivers necessários para serem executados em *palmtops* e *smartphones*.

Após o *Pocket* PC 2000, veio o *Pocket* PC 2002, que deu origem aos primeiros *smartphones*. A partir de então a *Microsoft* mudou novamente o nome do sistema para *Windows Mobile* 2003. Com o tempo, o *Windows Mobile* 2003 deu lugar ao *Windows Mobile* 5.0 e a partir do *Windows Mobile*6 apresentado em 2007 a *Microsoft* decidiu segmentar sua plataforma móvel em duas vertentes: o *Windows Mobile*6*Standard* (versão para *smartphones* sem tela *touchscreen*) e o *Windows Mobile* 6 *Professional* (versão para *Pocket* PCs com suporte a tela *touchscreen*). A diferença entre essas duas versões se dava tanto na interface quanto na grade de aplicativos, que necessitavam de adaptações de uma versão para a outra.

[1]Dispositivos embarcados são conjuntos de circuitos integrados que possuem capacidade computacional, ou seja, são sistemas completos que possuem a capacidade de processar dados de forma independente.

A imagem a seguir retrata o *Pocket* PC 2002 uma grande revolução para sua época:

Figura 9 – *Pocket* PC 2002

Fonte: Judy Novotny. 2012

De forma geral o *Windows Mobile* 6 se diferenciou da versão 5 abandonando a interface baseada no *Windows* 95 para utilizar uma interface semelhante ao *Windows Vista*.

Em 2010 o *Windows* CE 6.0 estava pronto, e junto com ele veio o anuncio da plataforma *Windows Phone*7 a atualização do *Windows Mobile* 6. Rodando em dispositivos móveis mais potentes e com suporte a sensores e recursos, empresas da área de desenvolvimento móvel se tornaram parceiras da *Microsoft*, entre elas LG, HTC, *Samsung* e em 2011 a *Nokia*.

Figura 10 – Windows CE 6.0

Fonte: Chris Ziegler. 2007

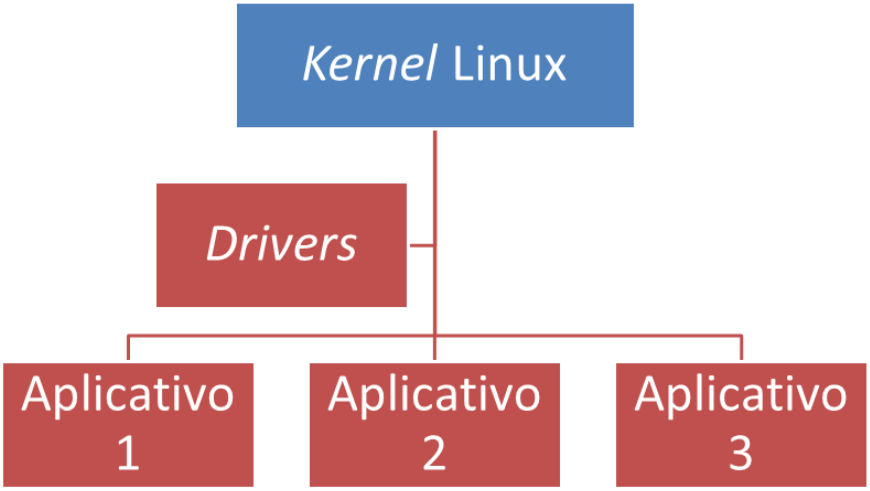
2.2.2 Histórico da plataforma Android

O *Android* é uma plataforma baseada no *Kernel* Linux ou GNU/Linux que é o núcleo do sistema operacional.

> O *kernel* é o núcleo do sistema operacional, responsável por vários serviços de gerenciamento de arquivos, compartilhamento de recursos da máquina, acesso a dispositivos, entre outros. É o *kernel* quem controla todos os dispositivos do computador e ele pode ser visto como uma interface entre os programas e todo o *hardware* (GUEDES, 2006, p. 09).

A imagem a seguir demonstra a estrutura do *Kernel* do Linux, sistema do qual a plataforma android é baseada.

Figura 11 – Esquema *Kernel*Linux



Fonte: Elaboração própria. 2013

Na década de 80 o programador Richard Stallman pediu demissão do Instituto de Tecnologia de *Massachusets* o MIT e começou o projeto *GNU is Not Unix* (GNU) para criação de um sistema operacional livre, de código aberto. Essa filosofia de *software* livre atraiu outros programadores que queriam quebrar as barreiras de cópias e distribuição de *softwares*. Assim na década de 90 o estudante Linus Torvalds desenvolveu o *Kernel* ou núcleo do sistema, que ainda não existia, chamando de Linux e disponibilizando na *internet*, para que os programadores interessados ajudassem a finalizar o projeto. Cerca de dois anos depois o Linux, também chamado GNU/Linux por trabalharem em conjunto, já era um sistema operacional estável.

Em 2005, a empresa *Android Inc* foi adquirida pela *Google*, interessada no desenvolvimento de um sistema operacional para celulares. Em 2007, com o objetivo de desenvolver um sistema com código aberto, fundou a *Open Headset Alliance* (OHA) junto com outras empresas dentre elas TI, LG e *Intel*, dando origem a uma plataforma para dispositivos móveis batizada de *Android*, com licença gratuita, código aberto e flexível. Proporcionando aos desenvolvedores de aplicativos a comunicação com o *hardware* para diversos aparelhos.

> *Android* é uma pilha de *softwares* para dispositivos móveis que inclui sistema operacional, e o *middleware* e aplicações. Entende-se por pilha de *software* um conjunto de aplicações que trabalham em conjunto para atingir

> determinado objetivo. Um Sistema Operacional, por exemplo, é uma pilha de *softwares*. Ele contém diversas aplicações trabalhando em conjunto para atingir seu objetivo, que é gerenciar o *hardware*. O termo pilha é usado porque numa pilha um elemento depende de outro. Essa ideia de dependência é o que se quer deixar claro no termo Pilha de *Softwares* (em inglês *Software Stack*), (PASSOS, 2009, p. 04).

Figura 12 – Dispositivo *Android*

Fonte: *Android.* 2013

O desenvolvimento de um aplicativo deve levar em consideração alguns fatores, são estes: a curva de aprendizado, o ambiente favorável de desenvolvimento e também de informações sobre a plataforma, como *Guidelines*[2] e referências bibliográficas. Esses fatores somados ao conhecimento da estrutura dos componentes dos aplicativos permitem o bom desenvolvimento de aplicativos e por este motivo serão apresentados dados que permitem uma análise dos componentes persistência e segurança, nas duas plataformas.

[2]*Guideline*, que pode ser traduzida por diretrizes, possui o objetivo de orientar o desenvolvimento de uma atividade ou tarefa seguindo uma série de padrões para a sua concretização. As *guidelines* podem ser aplicadas a qualquer forma de trabalho, método, atividade ou tarefa, com o objetivo de parametrizar seu desenvolvimento.

3 METODOLOGIA

Este trabalho científico é uma revisão bibliográfica que tem por objetivo investigar conceitos chave para o desenvolvimento de aplicativos para dispositivos móveis e tem como foco aspectos de segurança e persistência nas plataformas de desenvolvimento *Android* e *Windows Phone*. Objetiva também apresentar com base nas pesquisas efetuadas, dados para que os desenvolvedores possam decidir qual plataforma é mais adequada aos seus objetivos de desenvolvimento. Através da técnica de revisão bibliográfica as características desses aspectos foram colhidas e usando o método de pesquisa exploratória, os dados obtidos foram analisados e tratados a fim de se destacar semelhanças e diferenças, e baseado nessas análises concluímos sobre qual plataforma proporciona um melhor cenário para o trabalho de desenvolvimento de aplicações considerando fatores como o tempo de execução, a curva de aprendizado e a acessibilidade às ferramentas de desenvolvimento.

Espera se que este trabalho científico ajude a esclarecer qual das duas plataformas oferece um melhor cenário de desenvolvimento de um aplicativo móvel. Espera se também que seja revelado por que estes aspectos se destacam em cada plataforma a fim de formar uma conclusão concisa para o desenvolvedor na hora em que for desenvolver um aplicativo móvel.

Devido à grande fragmentação dos sistemas operacionais móveis, neste caso tanto *Android* quanto *Windows Phone* e dada à velocidade com a qual são realizadas as atualizações e lançamento de novas versões para estas plataformas, há a necessidade de definir quais versões serão abordadas neste estudo. Serão abordadas as versões 7,5 (*Mango*) da plataforma *Windows Phone* e 2.3 (*Gingerbread*) da plataforma *Android*.

4 RESULTADO DA PESQUISA

4.1 CONSIDERAÇÕES INICIAIS

A seguir, serão explorados os aspectos de persistência e segurança das plataformas de desenvolvimento *Android* e *Windows Phone,* onde serão analisadas as características desses aspectos e por fim será realizada uma análise comparativa. Tal análise se faz necessária para que se obtenha uma clara visão de qual plataforma se torna uma melhor opção para o desenvolvedor na hora de criar um aplicativo, levando em conta a relevância dos aspectos apresentados para atingir os requisitos esperados pela aplicação.

4.2 SEGURANÇA

Uma das preocupações dos desenvolvedores de aplicativos é a segurança. Devido ao uso diário dos dispositivos moveis que se conectam à internet e por esta interação ser implícita aos usuários, que acabam não acreditando nos riscos ou os desconhecendo, a segurança se torna um aspecto importante devido à quantidade de aplicativos mal intencionados, redes sem fio inseguras onde é possível a troca de informações sem o conhecimento do usuário, por exemplo, via *Bluetooth*. Segundo SACHSE o seguinte cenário é apresentado:

> Visto que as redes sem fio são usadas para transmitir pacotes de dados sem a necessidade do uso de cabos são mais facilmente interceptadas, podendo infectar o *smartphone* com *malware*, que pode ativar facilmente funcionalidades físicas nativas do *smartphone*, como o caso das câmeras, microfones ou até mesmo comprometer a capacidade de armazenamento de dados que existe no dispositivo (SACHSE, 2010, p. 09).

As vulnerabilidades dos dispositivos móveis se assemelham as de um computador já que eles efetuam o processamento de informações de maneira semelhante. Deste modo, são apresentadas a seguir as vulnerabilidades mais comuns nesses dispositivos.

Bluebug: Vulnerabilidade que explora uma brecha de segurança do Bluetooth dos aparelhos. Esta brecha permite a interação do vírus com o dispositivo de modo a criar um ambiente onde o vírus adquire o controle das chamadas do telefone, fazendo assim com que o aparelho faça ligações indevidas sem o conhecimento de seu utilizador.

Vírus: um aplicativo que corrompe arquivos e se espalha pelo sistema. De acordo com EIRAS (2004) temos a seguinte definição de vírus:

> Vírus é um programa capaz de infectar outros programas e arquivos de um computador. Para realizar a infecção, o vírus embute uma cópia de si mesmo em um programa ou arquivo, que quando executado também executa o vírus, dando continuidade ao processo de infecção.

Mobile Malware: um aplicativo malicioso, designado a danificar o dispositivo e aplicativos. Também conhecido como *malware*, vírus ou código malicioso.

Rogue Software: consiste em um aplicativo ilegal criado para induzir o usuário a comprar um falso aplicativo de antivírus. Estes aplicativos fazem uma falsa análise do sistema e apresentam por meio de avisos, falhas e erros inexistentes no dispositivo, os quais forçam o usuário a uma compra desnecessária. (DUNHAM, 2009).

Tais vulnerabilidades permitem que pessoas mal intencionadas tenham acesso a diversos recursos dos dispositivos móveis que estejam desprotegidos. Sendo assim é essencial a implementação adequada da segurança nos aplicativos desenvolvidos. Os aplicativos são responsáveis por gerenciar diversos recursos do dispositivo tais como fotos, vídeos, mensagens, contatos e outros que variam de acordo com o usuário. Considerando os riscos as vulnerabilidades dos aplicativos devem ser avaliadas a fim de prever uma manipulação segura dos dados do usuário.

> Os aplicativos para dispositivos móveis manipulam diversas informações sensíveis dos usuários, tais como contatos, mensagens recebidas e enviadas, histórico de ligações, e-mails, documentos de escritório, fotos, vídeos, histórico de navegação, informações de localização, credenciais de acesso a serviços online e informações financeiras. A possível exposição de tais informações pode causar danos ao usuário, aumentando os riscos associados a estes aplicativos. Adicionalmente, podem existir vulnerabilidades nos aplicativos que possibilitem a execução remota de código ou o vazamento das informações mantidas por ele para outros aplicativos, ou até mesmo para outras entidades remotas (BRAGA, 2012, p. 85).

A seguir apresenta se o aspecto de segurança nas plataformas de desenvolvimento *Android* e *Windows Phone* objetivando a compreensão de suas características e a necessidade de sua utilização.

4.2.1 Segurança no *Android*

Por ser baseado em *Kernel* Linux, o *Android*, segue o mesmo modelo de segurança voltado a permissões de usuário, onde cada usuário recebe um *user-id* (UID) e o acesso aos recursos do sistema é controlado por usuário.

Os recursos do sistema Linux como, por exemplo, rede de *internet* ou tocador de músicas possuem três tipos de permissão, leitura, escrita e execução que são aplicadas a cada usuário de forma isolada conforme o grupo ao qual pertence, podendo ser mais ou menos privilegiado.

O *Android* por sua vez trata cada aplicativo como um usuário e atribui a ele um UID único no memento em que é instalado. Também é criado um diretório no sistema de arquivos onde os dados do aplicativo serão armazenados e somente o UID do aplicativo possui total acesso a esse diretório, outros aplicativos não possuem permissões de acesso.

> A plataforma *Android* aproveita a gestão de utilizadores providenciada pelo *kernel* Linux para atacar o problema do isolamento de aplicações. O *Android* atribui um identificador de utilizador (UID) a cada aplicação na altura da sua instalação, não partilhado por qualquer outra aplicação (CIBRÃO, 2012, p. 06).

As permissões do aplicativo são declaradas em um arquivo de extensão *Extensible Markup Language* (XML) chamado *Android Manifest* que deve obrigatoriamente existir em todos os aplicativos, para que o *Android* tenha conhecimento dos recursos necessários para esse aplicativo antes de sua execução. Suas principais funções, dentre outras, são: dar um nome que serve como identificador único para o aplicativo, declarar as permissões do aplicativo, quais permissões outros aplicativos tem para interagir com o mesmo e listar as bibliotecas necessárias para o aplicativo.

> Projetos *Android* incluem um arquivo de manifesto, que acompanha o *software* e os recursos do projeto quando este é construído e que diz ao sistema *Android* como instalar e utilizar o *software*. O arquivo de manifesto está em linguagem XML [...] (MEDNIEKS, 2012, p. 38).

Figura 13 – *Android Manifest*

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.jsupport"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="15" />

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />

    <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name" >
        <activity
            android:name=".CheckConnectionActivity"
            android:label="@string/title_activity_check_connection" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Fonte: Elaboração própria. 2013

Responsável por gerenciar o acesso aos recursos e arquivos do sistema o *Sandbox*[3] é criado quando o aplicativo é executado. Os processos são executados de forma isolada ao conteúdo interno do *Sandbox* de modo a impedir que o aplicativo acesse informações externas e protegendo assim o sistema e outros aplicativos de serem alterados, a menos que uma permissão tenha sido concedida previamente no *Android Manifest* do aplicativo, por exemplo, no momento de sua instalação quando o aplicativo indica quais os conteúdos externos poderão ser acessados, no entanto essa permissão não pode ser alterada posteriormente, somente com uma nova instalação do aplicativo.

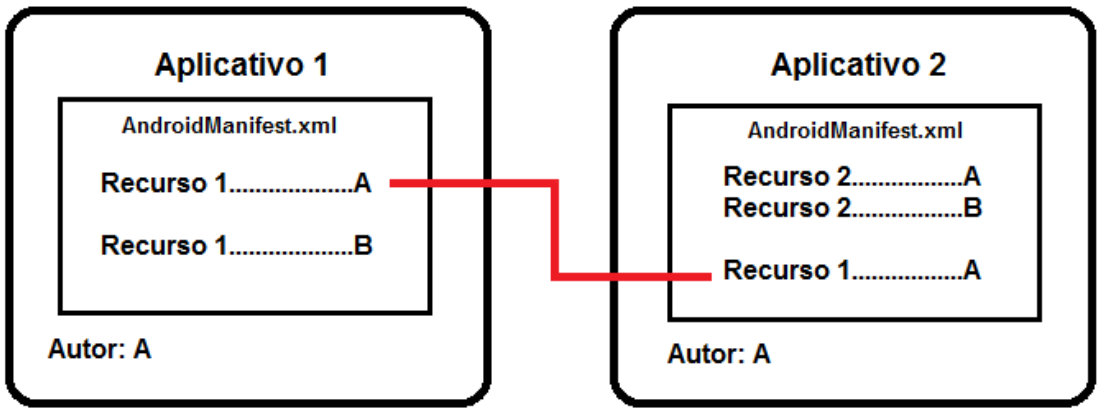
[3] O Sandbox é uma área controlada criada para conter um programa ou recurso, ela simula todos os requisitos necessários ao funcionamento do programa, para que dentro dela esse programa possa fazer o que quiser. Sendo assim o programa não pode sair dessa caixa de confinamento e todos os arquivos por ele criado são apenas temporários ou cópias dos originais que por sua vez permanecem intactos enquanto as cópias são apagadas ao final da sessão.

Um *Sandbox* único para cada aplicativo utilizando o UID, assim impedindo a interação entre aplicativos, caso um deles possua permissão no *Android Manifest* para usar o mesmo processo, os dois serão tratados como apenas um, com o mesmo identificador e as mesmas permissões.

> Cada processo de uma aplicação é executado numa *sandbox*. De modo a compreender o conceito de *sandbox*, é necessário imaginar uma "caixa" na qual se pode utilizar o que estiver lá dentro, mas não se pode utilizar nada que esteja no seu exterior, protegendo assim todo o ambiente que envolve a "caixa". Com o uso de uma *sandbox* é possível evitar aplicações de se poderem danificar ou acenderem a ficheiros de outras, a menos que exista uma declaração no ficheiro do *Android Manifest* da aplicação (SACHSE, 2010, p.35).

No momento da publicação do aplicativo o programador cria uma assinatura digital no mesmo. Essa assinatura está presente em todos os aplicativos *Android* que identificam o autor e permite sua avaliação por parte dos usuários dizendo se é confiável ou não. A assinatura permite também que dois aplicativos de mesmo autor possam interagir entre si, podendo até mesmo compartilhar um UID desde que seja corretamente configurado no *Android Manifest* de ambos.

Figura 14 – Esquema de compartilhamento de permissões no *Android*



Fonte: Elaboração própria. 2013

4.2.2 Segurança no Windows Phone

O *Windows Phone* tem como base uma estrutura de segurança constituída por camadas de encapsulamento e permissões de acesso. Cada camada possui uma política de segurança e um limite de acesso próprio, que define como cada processo

será executado e como vai se comportar. Foram definidas quatro camadas no sistema, sendo três formadas por conjuntos de permissões e uma quarta de capacidade à navegação. Essa quarta são acessos dados na hora da instalação do aplicativo e não podem ser alterados a menos que seja feita uma nova instalação.

Para um melhor entendimento de como as camadas se comportam vamos a uma descrição sucinta dessas camadas da mais privilegiada a menos privilegiada:

The Trusted Computing Base (TCB): camada mais privilegiada permite o maior acesso dos recursos do sistema pela aplicação, pode modificar sua política de segurança tal que drivers do *Kernel* possam ser executados na mesma, é importante minimizar as aplicações nessa camada para maior segurança no *Windows Phone*.

The Elevated Rights Chamber (ERC): pouco menos privilegiada permite acesso a todos os recursos exceto políticas de segurança, é mais indicada para aplicações que utilizam funcionalidades do aparelho como telefonia.

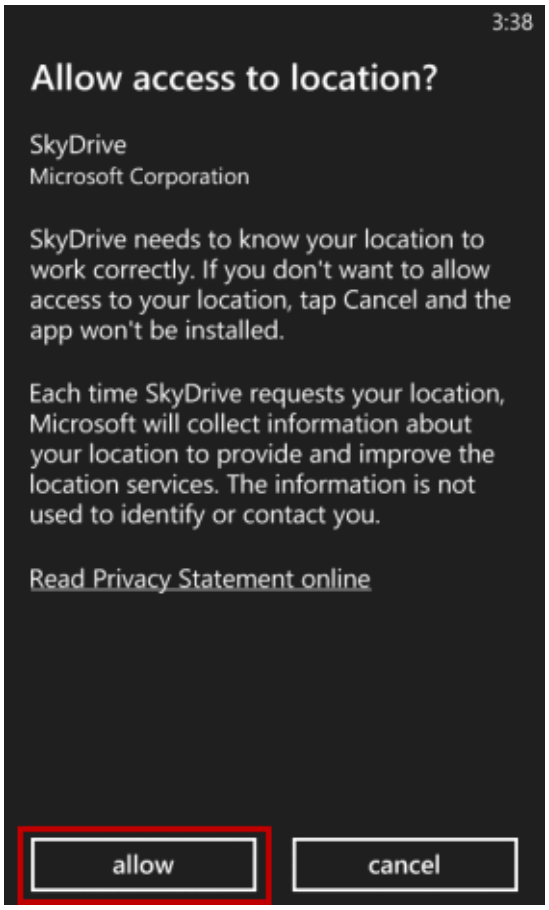
The Standard Rights Chamber (SRC): camada padrão de aplicativos nativos, ou seja, pré-instalados, todos os aplicativos *Microsoft* já disponíveis no *Windows Phone* rodam nessa camada.

The Least Privileged Chamber (LPC): camada com menor privilégio é a padrão para todos os aplicativos que não são da *Microsoft* disponíveis no *Marketplace Hub* (também conhecido como *Windows Phone Marketplace*).

Por serem responsáveis por gerenciar os privilégios de acesso a recursos do sistema as *Capabilities* possuem informações sobre as necessidades de acesso dos aplicativos. Dentre os aplicativos que são gerenciados pelas *Capabilities* temos os de localização, câmera, rede e os aplicativos instalados na LPC, que por sua vez define o menor privilégio de acesso que virá por padrão na instalação. O uso do LPC vem em função de reduzir ataques, de modo que cada aplicativo tem permitido o acesso necessário para sua execução sendo divulgado durante sua instalação na *Marketplace* e na primeira ativação de um desses recursos. Após a aceitação do que foi solicitado na instalação do aplicativo o nível de acesso não pode ser alterado em tempo de execução. A *Capability* necessária à aplicação é medida por ferramentas distribuídas pela *Microsoft* para desenvolvedores. Para que os aplicativos trabalhem com esses acessos de forma isolada é usado o *Sandbox*. Sendo assim os recursos que o aplicativo necessita são definidos pela *Capability* e o *Sandbox* é criado de

acordo com as necessidades do mesmo. Os aplicativos possuem permissões mínimas, como acesso a armazenamento isolado de arquivos, não podendo interagir entre si. As permissões são isoladas de tal forma que nem o acesso à memória, utilizada por uma aplicação pode ser cedida à outra ou até mesmo o *cache* do teclado. Outro uso do *Sandbox* pelo *Windows Phone* é o desligamento automático de aplicativos que não pertencem à *Marketplace*, esses não podem ficar ativos em segundo plano caso não utilizados, isso garante restrição ao uso dos recursos críticos do sistema como serviços com acesso à *internet*. (MICROSOFT, 2010).

FIGURA 15 – Permissão de acesso a dados de localização no *Windows Phone*



Fonte: Elaboração própria. 2013

4.2.3 Análise comparativa

O *Android* consegue implementar bem seu modelo de segurança baseado em *Kernel* Linux, voltado a restrições de acessos pelos aplicativos, onde nenhum

aplicativo pode executar funções que entrem em conflito ou danifiquem outro aplicativo ou o próprio sistema. Também restringindo o aplicativo de ler, escrever, ou acessar arquivos pertencentes a outro. Os principais recursos utilizados para esse modelo de segurança são o *Sandbox*, o UID e a assinatura digital do aplicativo.

No momento em que é solicitada a instalação do aplicativo são exibidas para o usuário as informações das necessidades de permissões do mesmo. Caso o usuário aceite, durante a instalação é criado um UID único de acesso para o aplicativo que será associado a um *Sandbox* criado em um diretório de armazenamento quando executado. Utilizando o UID de acesso o *Sandbox* permite que apenas esse aplicativo acesse o diretório, preservando assim o sistema e os arquivos externos que só poderão ser acessados caso exista uma permissão no *Android Manifest*.

Figura 16 – Solicitação de acesso a recursos do dispositivo no *Android*

Fonte: Elaboração própria. 2013

A imagem abaixo mostra um aplicativo já instalado no sistema e suas permissões:

Figura 17 – Aplicativo *Android* instalado no sistema e suas permissões.

Fonte: Elaboração própria. 2013

A principal vulnerabilidade neste caso está em permitir que o usuário aceite as permissões impostas pelo aplicativo, uma vez que o *Android* possui um serviço de depuração que pode ser ativado em suas configurações permitindo a instalação de aplicativos com maior privilégio.

> Existe um serviço de depuração em todo sistema *Android* que pode ser ativado em suas configurações. [...] Esse depurador possibilita a instalação e desinstalação de aplicativos, o gerenciamento de logs, a execução de comandos de *shell*, a cópia de arquivos de e para o dispositivo, a geração e restauração de *backups*, entre outros (BRAGA, 2012, p. 09).

Caso motivado o usuário pode conseguir acesso de *super user* também conhecido como *root* que permite total acesso ao sistema podendo esse ser passado ao aplicativo caso permitido pelo usuário.

> Como o *Android* é derivado do Linux, fazer o *rooting* equivale a obter permissões de acesso administrativo no dispositivo, ou seja, as permissões da conta *root*. As motivações para a habilitação de tal acesso são várias, como por exemplo: Instalação de versões modificadas do *Android* (sendo a *ClockWork Mod* a mais famosa delas); Uso de temas personalizados; Executar modificações no *kernel*; *Backup* de todos os dados, pois é necessário acesso administrativo para se obter tais dados; Ativar funcionalidades que foram bloqueadas por operadoras (como o NFC) (BRAGA, 2012, p. 62).

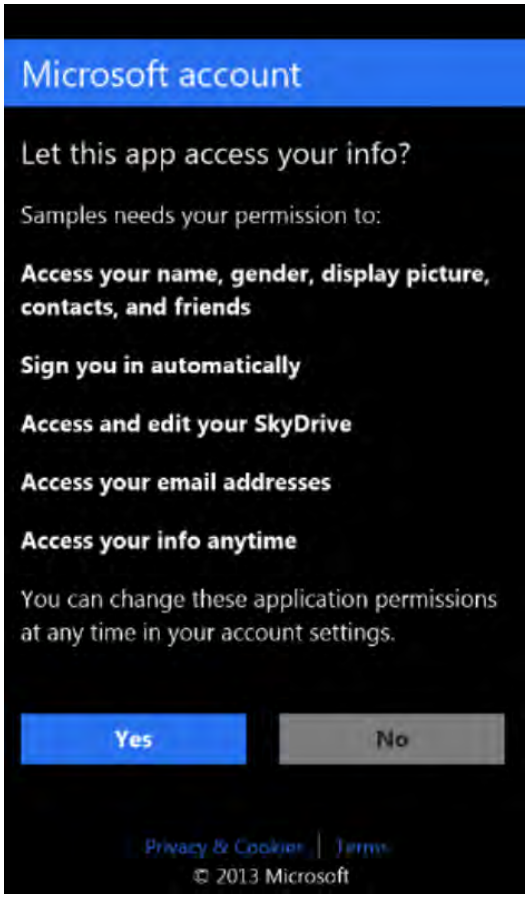
Figura 18 – Opções para habilitar acesso *roote* depuração USB no *Android*.

Fonte: Elaboração própria. 2013

O *Windows Phone* tem sua visão mais focada em permissões de acesso aos recursos do sistema, visto que utiliza uma estrutura em camadas, onde cada uma possui padrões de segurança pré-definidos restringindo os aplicativos. Esse modelo de segurança consiste em separar os aplicativos de acordo com seu nível de acesso aos recursos necessários definidos na *Capability* e informados ao usuário no momento da instalação, caso sejam aceitas as permissões o próprio sistema define

a qual camada o aplicativo irá pertencer e no momento da execução um *Sandbox* é criado de acordo com as necessidades do mesmo.

Figura 19 – Pedido de acesso a recursos do sistema no *Windows Phone*.



Fonte: Elaboração própria. 2013

Por se tratar de um sistema desenvolvido pela *Microsoft,* portanto de código fechado, o *Windows Phone* não permite a instalação de aplicativos que não estejam na *Marketplace* e não possui por padrão opções que possam habilitar tais privilégios, como depuração USB e fontes desconhecidas que existem no *Android*. Porém, caso motivado o usuário pode conseguir informações via *internet* de como realizar *jailbreak* que consiste em desbloquear o sistema a fim de permitir a instalação de aplicativos que não foram aprovados e publicados na *Marketplace*. Contudo o método *jailbreak* não libera total acesso aos recursos do sistema e pode causar danos ao dispositivo.

> Um grupo de desenvolvedores lançou na última quinta-feira (25/11) o primeiro *jailbreak* para o recém-lançado *Windows Phone* 7. Com o método, é possível desbloquear os *smartphones* com a plataforma e instalar aplicativos que não tenham sido publicados na loja *Marketplace* (IDG *Now*, 2010).

Portanto encontra-se na plataforma *Windows Phone* vantagem em desenvolver aplicativos que sejam voltados à segurança ou que necessitem de um grau mais elevado de proteção, como aplicativos de localização ou que efetuem compras pela *internet*. São encontradas grandes restrições na plataforma da *Microsoft* que contribuem para um eficiente modelo de segurança, tais como o valor de registro dos *softwares* necessários e a restrição do material disponível a respeito da plataforma. No entanto também é encontrada uma grande dificuldade quando buscamos informações aprofundadas nesse aspecto, o que pode dificultar o desenvolvimento de aplicativos com uma segurança melhor implementada, ficando a cargo da plataforma maior responsabilidade em gerenciar a segurança dos aplicativos.

A dificuldade em obter informações sobre o desenvolvimento de aplicativos na plataforma *Windows Phone* acaba por inibir novos desenvolvedores o que torna mais difícil a propagação de meios para violar sua segurança e colocar em risco os dados do usuário, porém reduz a comunidade de desenvolvedores que podem ajudar a melhorar este aspecto criando novos modelos de segurança para essa plataforma.

Já a plataforma *Android* tem seu modelo de segurança menos restrito e por padrão possui opções que podem liberar a instalação de aplicativos de terceiros. A plataforma da *Google* pode parecer insegura, mas tem um modelo de segurança bem implementação, contudo devido à pouca restrição permite que mesmo aplicativos inseguros e mal intencionados sejam instalados.

Diferente do *Windows Phone* o conteúdo para desenvolvimento *Android* é facilmente encontrado. Por possuir código aberto, as informações sobre a plataforma e o material para criação de aplicativos é disponibilizado via *internet* tornando assim a comunidade de desenvolvedores mais extensa, logo, fica sob maior responsabilidade do desenvolvedor estudar as possíveis formas de segurança e implementá-las da melhor maneira possível.

4.3 PERSISTÊNCIA

A persistência de dados é um dos pontos importantes do desenvolvimento de aplicativos, pois consiste na capacidade de manter o estado de um objeto em um meio não volátil (persistente). A persistência pode ser implementada de diferentes formas, como por exemplo, o armazenamento de dados de *login* de um usuário em arquivos ou em um banco de dados, de forma que ao utilizar novamente o aplicativo, tais informações já estejam presentes no contexto do aplicativo. Tais dados poderão ser consultados, excluídos ou modificados de forma transparente, bastando que o usuário interaja com o aplicativo.

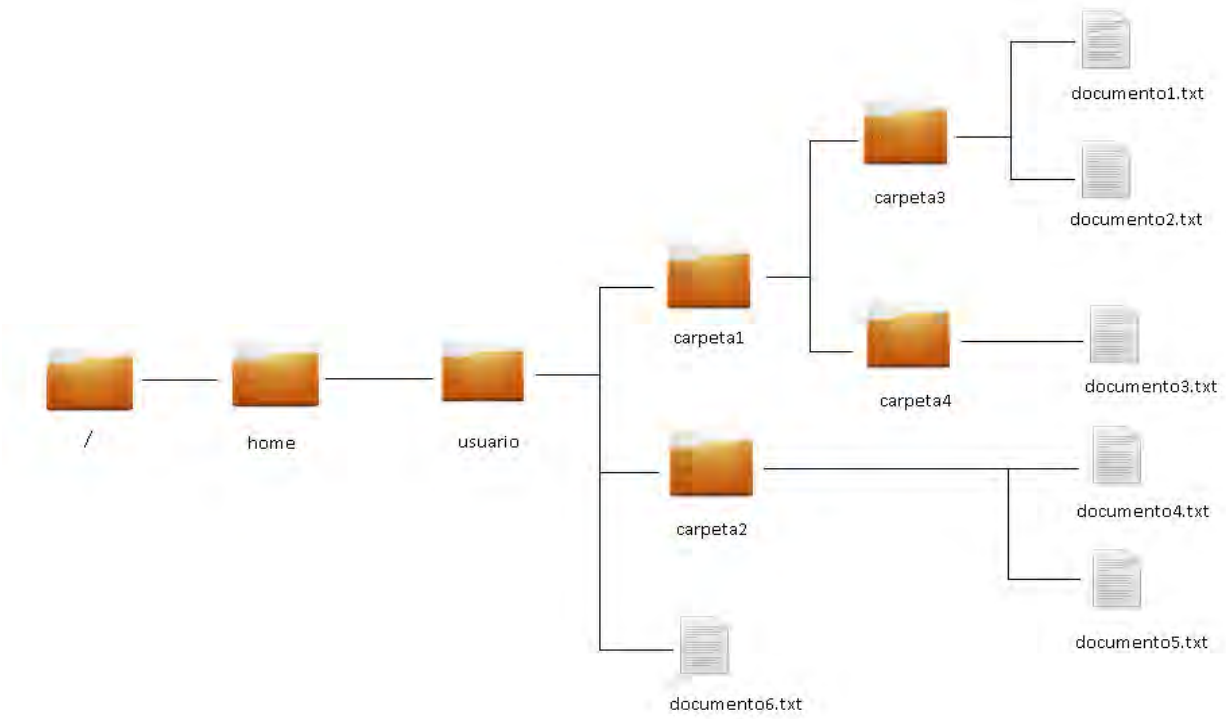
> Persistência é um dos conceitos fundamentais em desenvolvimento de aplicativos. Se um sistema de informação não preservasse dados inseridos por usuários quando a máquina central fosse desligada, o sistema seria de pequeno uso prático (BAUER; KING, 2005, p. 05).

Levando em consideração as limitações dos dispositivos móveis em processar e armazenar as informações dos usuários e dos aplicativos, torna-se imprescindível a compreensão das formas de armazenamento e recuperação de tais informações, portanto é necessário que se tenha o entendimento sobre o que são sistemas de arquivos, bancos de dados, sistemas gerenciadores de bancos de dados (SGBD) e armazenamento na nuvem.

Sistema de Arquivos: o Sistema de arquivos é o meio que o sistema operacional utiliza para armazenar dados relevantes aos processos das aplicações de forma duradoura. Dessa forma, quando uma aplicação precisa armazenar dados, ela cria no disco de memória do dispositivo um arquivo que contêm tais dados, posteriormente quando é necessário recuperar esses dados, a aplicação que criou o arquivo ou aplicações que possuam as devidas permissões acessarão o arquivo e extrairão os dados necessários.

> Arquivos são unidades lógicas de informação criadas por processos. Em geral, um disco contém milhares de arquivos, um independente do outro. Na verdade, os arquivos também são uma espécie de espaço de endereçamento, mas eles são usados para modelar o disco e não a memória RAM.
>
> Os processos podem ler os arquivos existentes e criar novos, se necessário. A informação armazenada em arquivos deve ser persistente, isto é, não pode ser afetada pela criação e pelo término de um processo. Um arquivo só desaparecerá quando seu proprietário removê-lo explicitamente (TANEMBAUM, 2008, p. 158).

Figura 20 – Sistema de arquivos



Fonte: Fernando Ureña Gómez. 2013

Banco de Dados: Um banco de dados é uma coleção de informações lógicas e coerentes que representa algumas características do mundo real. Um banco de dados por vezes pode ser conhecido como universo de discurso (UoD) ou mini-mundo. Geralmente os bancos de dados são idealizados, criados e populados por dados para satisfazer uma proposta específica, seja armazenar um agrupamento de nomes e telefones ou ainda armazenar informações sobre saques e depósitos das contas de um banco. Os bancos de dados podem possuir tamanhos e complexidades variadas, pois sua criação e utilização irá variar de acordo com o propósito pelo qual eles foram idealizados.

> Um banco de dados é uma coleção de dados relacionados. Os dados são fatos que podem ser gravados e que possuem um significado implícito. Por exemplo, considere nomes, números telefônicos e endereços de pessoas que você conhece. Esses dados podem ter sido escritos em uma agenda de telefones ou armazenados em um computador, por meio de programas como o *Microsoft Access* ou *Excel*. Essas informações são uma coleção de dados com um significado implícito, consequentemente, um banco de dados (NAVATHE, 2005, p. 04).

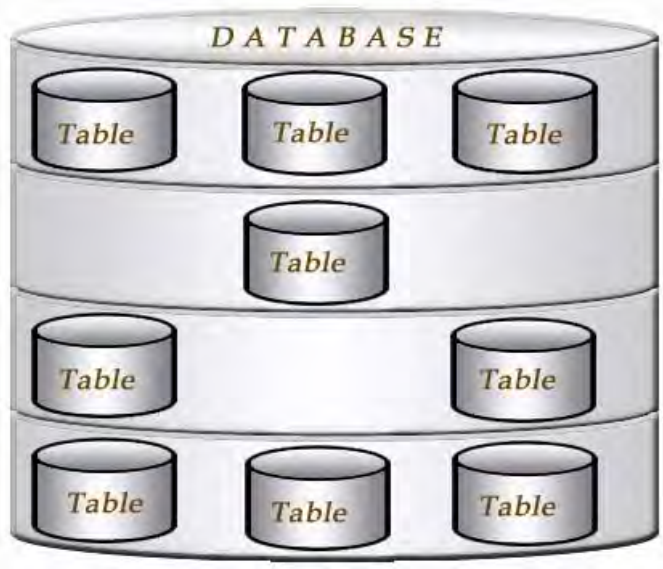
Os bancos de dados podem ser gerados e mantidos manualmente ou automaticamente. Essa geração varia de acordo com a necessidade que da utilização do banco. De acordo com NAVATHE temos a seguinte afirmação a respeito dos bancos de dados:

> Um banco de dados pode ser gerado e mantido manualmente ou pode ser automatizado (computadorizado). Por exemplo, um catálogo de cartões bibliotecários é um banco de dados que oferece a possibilidade de ser criado e mantido manualmente. Um banco de dados computadorizado pode ser criado e mantido tanto por um grupo de aplicativos escritos especialmente para essa tarefa como por um sistema gerenciador de banco de dados(NAVATHE, 2005, p. 04).

A empregabilidade de um banco de dados varia de acordo com as necessidades do sistema ou do usuário. Deste modo, fica a cargo do desenvolvedor do sistema verificar a necessidade de utilização do mesmo em detrimento da utilização de, por exemplo, o sistema de armazenamento em arquivos.

Dentre todas as características da abordagem de um banco de dados, algumas merecem ser citadas. São elas: A natureza auto descritiva que um banco de dados possui; o desacoplamento entre os programas e dados, além da abstração deste segundo; o apoio a múltiplas visões de dados; transações entre vários usuários ocorrendo ao mesmo tempo somado ao processamento também compartilhado.

Figura 21 – Esquema de banco de dados



Fonte: W3resource. 2010

Sistema Gerenciador de Banco de dados (SGBD): O sistema gerenciador de banco de dados é um conjunto de *softwares* que tem o objetivo de aperfeiçoar todo o tratamento que é feito em um banco de dados. Ele cuida da idealização, criação, utilização e compartilhamento de bancos de dados.

> Um Sistema Gerenciador de Base de Dados (SGBD) é uma coleção de programas que permitem aos usuários criarem e manipularem uma base de dados. Um SGBD é, assim, um sistema de *software* de propósito geral que facilita o processo de definir, construir e manipular bases de dados de diversas aplicações (DATE, 2004, p. 15).

Entretanto, para NAVATHE temos a seguinte definição do que é um sistema gerenciador de banco de dados:

> Um sistema gerenciador de banco de dados (SGBD) é uma coleção de programas que permite aos usuários criar e manter um banco de dados. O SGBD é, portanto, um sistema de *software* de propósito geral que facilita os processos de definição, construção, manipulação e compartilhamento de bancos de dados entre vários usuários e aplicações. A definição de um banco de dados implica especificar os tipos de dados, as estruturas e as restrições para os dados a serem armazenados em um banco de dados(NAVATHE, 2005, p. 04).

Assim, é de grande valia o entendimento do desenvolvedor de como manipular um sistema gerenciador de banco de dados. É através deste que pode ser feita principalmente, a proteção e a manutenção de um banco de dados.

Esta manipulação do SGBD conta ainda com outras funções, como geração de relatórios dos dados, recuperação de dados específicos, gerenciamento do acesso concorrente. A proteção do banco de dados é feita pelo SGBD tanto emnível de *hardware* quanto de *software*, garantindo assim que não ocorram *Crashes* (falhas) ou acessos a dados indevidos do banco.

Armazenamento de dados na nuvem: Por Souza pode ser definido da seguinte forma:

> Computação em nuvem é uma tendência recente de tecnologia cujo objetivo é proporcionar serviços de Tecnologia da Informação (TI) sob demanda com pagamento baseado no uso. Tendências anteriores à computação em nuvem foram limitadas a uma determinada classe de usuários ou focadas em tornar disponível uma demanda específica de recursos de TI, principalmente de informática [Buyyaet al. 2009]. Computação em nuvem pretende ser global e prover serviços para as massas que vão desde o usuário final que hospeda seus documentos pessoais na *Internet* até empresas que terceirizam toda infraestrutura de TI para outras empresas. Nunca uma abordagem para a utilização real foi tão global e completa: não

> apenas recursos de computação e armazenamento são entregues sob demanda, mas toda a pilha de computação pode ser aproveitada na nuvem. A computação em nuvem surge da necessidade de construir infraestruturas de TI complexas, onde os usuários têm que realizar instalação, configuração e atualização de sistemas de *software*. Em geral, os recursos de computação e *hardware* são propensos a ficarem obsoletos rapidamente e a utilização de plataformas computacionais de terceiros é uma solução inteligente para os usuários lidarem com a infraestrutura de TI. Na computação em nuvem os recursos de TI são fornecidos como um serviço, permitindo que os usuários o acessem sem a necessidade de conhecimento sobre a tecnologia utilizada. Desse modo, os usuários e as empresas passaram a acessar os serviços sob demanda e independente de localização, o que aumentou a quantidade de serviços disponíveis (SOUZA, 2010, p.101-130).

O armazenamento de dados na nuvem é uma solução desenvolvida para que empresas pudessem evitar problemas com a infraestrutura necessária para a persistência dos dados de suas aplicações. Tal artificio garante a integridade dos dados de modo que mesmo que ocorra o mau funcionamento dos aplicativos, os dados continuem seguros desta forma as empresas possuem a garantia de que os dados criados ou coletados por seus *softwares* se mantenham parcial ou até mesmo integralmente intactos.

O armazenamento na nuvem evoluiu dos serviços de tecnologia da informação sob demanda, conhecida como *Utility Computing* que tinha o objetivo proporcionar itens básicos ao funcionamento de sistemas *online*, como largura de banda e processamento. As empresas que usam a nuvem, portanto, não se preocupam com a escalabilidade do sistema porque nesse serviço, o armazenamento é fornecido de maneira transparente ao usuário para que esse não precise se preocupar com limitações de armazenamento.

A utilização do armazenamento na nuvem se torna um artifício de grande valor para a tecnologia móvel, pois através desse armazenamento, aplicativos dentro de um pequeno dispositivo móvel podem ter acesso a um vasto repositório de dados, sejam eles músicas, vídeos, livros ou outro tipo de mídia ou informação. Esses dados podem ser gerenciados pelos SGBD’s, uma vez que não necessitam ser armazenados no aparelho. Este fato traz uma infinidade de possibilidades, como o *streaming* de jogos, vídeos e músicas para dispositivos móveis. Dessa forma, o usuário não depende do armazenamento interno de seu aparelho, somente da sua conexão com a *internet*. De acordo com SOUZA ainda podemos ter mais uma série de vantagens ao utilizar um SGBD na nuvem.

> A infraestrutura de SGBDs em nuvem possui várias vantagens para os usuários: (i) previsibilidade e custos mais baixos, proporcional à qualidade do serviço (QoS) e cargas de trabalho reais, (ii) complexidade técnica reduzida, graças a interfaces de acesso unificado e a delegação de *tuning* e administração de SGBDs e (iii) a elasticidade e escalabilidade, proporcionando a percepção de recursos quase infinitos. Por outro lado, o provedor tem que garantir (i) a ilusão de recursos infinitos, sob cargas de trabalho dinâmicas e (ii) minimizar os custos operacionais associados a cada usuário (SOUZA, 2010, p.101-130).

Figura 22 – Armazenamento de dados na nuvem

Fonte: Daniel Pavani. 2011

4.3.1 Persistência no Android

Na plataforma *Android* existe mais de uma solução para o desenvolvimento da persistência, isso ocorre, pois cada aplicação difere uma da outra no quesito necessidade de armazenamento de dados e isso faz com que o desenvolvedor possa decidir por qual optar na hora de criar seu projeto de aplicativo.

No sistema de persistência da Plataforma *Android* os dados dos aplicativos ficam disponíveis somente para a aplicação que os persistiu, garantindo assim a segurança dos dados, e para esses dados sejam acessados por outras aplicações deve se utilizar componentes específicos para tal ação.

> Diferentemente de sistemas operacionais para *desktop*, que geralmente disponibilizam um sistema de arquivos comum, no *Android* todos os dados são visíveis apenas para a aplicação dona. Para que estas informações sejam acessadas por outras aplicações, deve ser utilizado um componente do tipo provedor de conteúdo (MARTINS, 2009, p. 11).

As opções de persistência são *Shared Preferences, Internal Storage, External Storage, SQLite data base* e *Network connection*. Que podem ser descritas da seguinte forma:

Shared Preferences: É um armazenamento de dados que permite que o desenvolvedor salve e recupere pares de tipos primitivos de dados no formato chave-valor. Com ele o desenvolvedor pode gravar em um XML do sistema informações relevantes e recuperá-las quando necessário. Atualmente a *Shared Preferences* não suporta o uso em múltiplos processos simultâneos. O *Shared Preferences* permite ao aplicativo armazenar as preferências do utilizador do aplicativo.

Internal Storage: Persiste os dados do aplicativo dentro da memória interna do aparelho, os dados armazenados desta forma são privados e, por conseguinte não podem ser acessados por outros aplicativos ou pelo usuário, somente pelo aplicativo que os armazenou.

External Storage: Por sua vez, persiste os dados do aplicativo dentro da memória externa do aparelho caso ela exista. Se não existir os dados serão armazenados dentro do armazenamento interno do aparelho. Os dados são armazenados de forma pública e tanto outros aplicativos quanto o usuário podem acessar.

Armazenamento Temporário: A plataforma pode valer-se do armazenamento temporário para persistir os dados das aplicações, destinando uma parte reservada do sistema para armazenar dados que serão utilizados mais frequentemente. Porém esse espaço é bem limitado, podendo variar entre 512kb a 1mb. Cabe ao desenvolvedor utilizar este tipo de persistência com certo cuidado, pois dependendo da situação do armazenamento em disco do aparelho, o sistema pode eliminar automaticamente os dados armazenados em *cache* (OGLIARI, 2011).

Content Providers: responsável por prover às aplicações os dados necessários para as mesmas funcionarem. As aplicações podem acessar diretamente um banco de dados, porém o *Content Providers* garante uma manipulação transparente, dando a elas a possibilidade de manter o foco nas interações dos usuários. O *Content Providers* também controla a interação entre aplicações que necessitam

compartilhar informações, encapsulando os dados e fornecendo assim meios para garantir a segurança dos mesmos (MEDNIEKS, 2011).

> Quando da necessidade de compartilhamento dos dados de sua aplicação com outras aplicações, utiliza-se o *contente provider*, que é basicamente a forma de compartilhar dados entre os aplicativos. Este permite que outras aplicações possam armazenar e recuperar dados no mesmo repositório utilizado por aquela aplicação (PEREIRA, 2009, p. 15).

SQLite data base: Se faz necessário armazenar várias informações com a mesma estrutura. *SQLite* é um banco de dados privado e de pequeno porte, muito utilizado em plataformas embarcadas, pois consome pouco espaço e traz as mesmas funcionalidades de qualquer banco de dados. O *SQLite* atua diferentemente da maioria dos bancos de dados, pois lê e escreve diretamente em discos de armazenamento comuns (memória *flash* ou *HD's*). Como em qualquer outro banco de dados, é possível incluir, excluir, alterar e consultar dados usando o *SQLite*.

A plataforma *Android*, para facilitar a vida dos desenvolvedores e dos usuários de aplicativos que precisam persistir dados de alguma forma, traz a biblioteca *SQLite* de forma nativa, isso significa que a qualquer momento durante o ciclo de vida da aplicação é possível criar um banco de dados completo para inclusão, alteração, consulta e deleção de dados. Na plataforma *Android*, o *SQLite* não precisa de nenhum procedimento de configuração específica, basta executar os comandos SQL necessários para a criação e atualização do banco de dados e após isso a própria plataforma gerencia o banco de dados (GARGENTA, 2011).

O *SQLite* é um banco de dados *open source* que suporta os mesmos recursos dos bancos de dados relacionais padrão, ou seja, compreende a linguagem (SQL) e instruções provenientes da mesma. O *SQLite* se comporta como um mini Sistema Gerenciador de Banco de dados (SGBD) e portanto possui as mesmas funcionalidades deste.

Uma vantagem do uso do *SQLite* na plataforma *Android* é o seu baixo consumo de espaço pois com todos seus recursos ativados o tamanho da biblioteca ainda pode ser inferior a 500KB, isso significa que ocupará pouco espaço em disco. Além disso, o *SQLite* requer pouca memória em tempo de execução (cerca de 250 *kbytes*), o que o torna uma boa escolha para atuar em conjunto com o tempo de execução das aplicações de um dispositivo móvel, que em geral possui pouca memória RAM.

Ainda pode se ressaltar que o arquivo de banco de dados do *SQLite* é multiplataforma, ou seja, pode se exportar o arquivo de banco de dados para utilizar em diferentes plataformas (SQLITE, 2013).

Figura 23 – Código *SQLite* no *Android*

```
1  package br.org.h4a.test.model;
2
3  import br.com.softctrl.h4android.orm.annotation.ddl.Column;
4  import br.com.softctrl.h4android.orm.annotation.ddl.Entity;
5  import br.com.softctrl.h4android.orm.annotation.ddl.Id;
6  import br.com.softctrl.h4android.orm.annotation.ddl.Table;
7  import br.com.softctrl.h4android.orm.annotation.ddl.Version;
8  import br.com.softctrl.h4android.orm.enumeration.validation.TypeColumn;
9
10 @Entity
11 @Table("TBL_CONTATO_")
12 public class Contato {
13
14     @Id
15     private Integer id;
16     @Version
17     private Integer version;
18     @Column(name = "_NOME", length = 50, allowNulls = false, typeColumn = TypeColumn.VARCHAR)
19     private String nome;
20     @Column(name = "_TELEFONE", length = 14, allowNulls = false, typeColumn = TypeColumn.VARCHAR)
21     private String telefone;
22     @Column(name = "_CELULAR", length = 14, typeColumn = TypeColumn.VARCHAR)
23     private String celular;
24     @Column(name = "_EMAIL")
25     private String email;
26
27 }
```

Fonte: Carlos Timoshenko Rodrigues Lopes. 2010

4.3.2 Persistência no Windows Phone

A persistência normalmente é implementada por duas vias no *Windows Phone*, o *Persistent Data* e o *Transient Data*.

Transient Data: É o meio de armazenar temporariamente os dados que não farão parte do ciclo de vida da aplicação. Tais dados são mantidos temporariamente na memória e descartados quando não estão mais em utilização. Para exemplificar, podemos considerar como dados temporários o texto digitado em campos de um formulário de uma aplicação. No caso de o usuário clicar em um botão de voltar sem ter realizado a operação de salvar, a informação contida nos campos do formulário serão perdidas e o usuário precisará preenchê-los novamente. A *Transient Data*

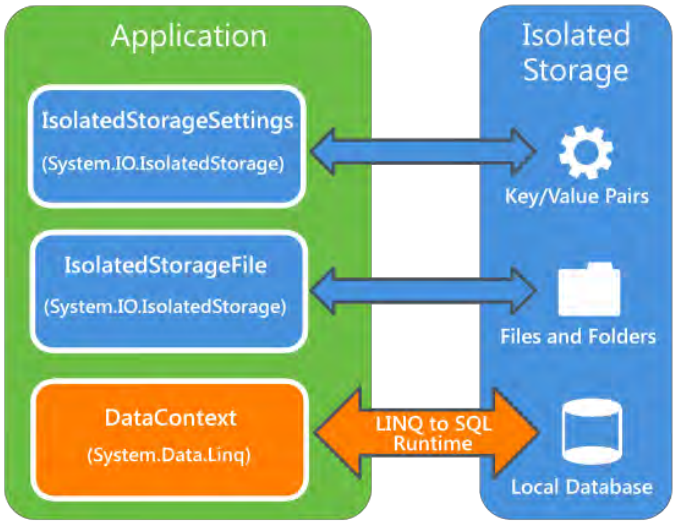
possui um limite de armazenamento por página de 2mb e 4mb para toda a aplicação (SATO, 2011).

Persistent Data: Usando *Persistent Data* é possível através do *Isolated Storage*, manipular as informações do aplicativo. O *Isolated Storage* é o mecanismo que permite que os aplicativos criem e controlem arquivos e pastas no armazenamento do dispositivo. Sendo assim o *Isolated Storage* possui também a habilidade necessária para salvar e ler dados de arquivos que são armazenados dentro do dispositivo.

> O *Isolated Storage* nada mais é do que um mecanismo que permite que as aplicações criem e gerenciem um repositório local de informações. Este conceito já existe há muito tempo nas aplicações .NET, mas começou a ser amplamente utilizada com o surgimento do *Silverlight*, uma vez que todas as manipulações de I/O só eram possíveis por meio da utilização de *Isolated Storage*, restringindo qualquer acesso direto ao sistema de arquivos do sistema operacional, garantindo assim a segurança dos dados de nosso usuário (WINDOWSPHONEBRASIL, 2010).

Ao criar um arquivo no *Isolated Storage*, operações de entrada e saída de dados são restritas ao local da aplicação, não sendo possível o acesso através de qualquer outro sistema de arquivos ou aplicação. Assim os dados do aplicativo são protegidos contra acessos não autorizados. Para compartilhar dados entre aplicações é sugerido o armazenamento de dados na nuvem.

Figura 24 – Formas de armazenamento de dados usando *Isolated Storage*



Fonte: Thiago Lunardi. 2012

A imagem a seguir apresenta um trecho da implementação da persistência na *Isolated Storage:*

Figura 25 – Implementação *Isolated Storage*

```
using System.IO.IsolatedStorage;

public partial class MainPage : PhoneApplicationPage
{
  IsolatedStorageSettings IsoSet = IsolatedStorageSettings.ApplicationSettings;
  public MainPage()
  {
    TextBlock.Text = "Ultimo acesso em " + UltimoAcesso.ToString();
    UltimoAcesso = DateTime.Now;
  }

  private DateTime UltimoAcesso
  {
    get
    {
      if(!IsoSet.Contains("UltimoAcesso")) IsoSet.Add("UltimoAcesso", DateTime.Now);
      return (DateTime)IsoSet["UltimoAcesso"];
    }
    set
    {
      IsoSet["UltimoAcesso"] = value;
    }
  }
}
```

Fonte: Elaboração própria. 2013

Dentre os recursos que o *Isolated Storage* possui podemos destacar a criação de pastas especificas para diferentes tipos de mídias e de utilização, são elas:

Shared – A pasta *Shared* têm o propósito de hospedar mídias, arquivos de sistema e pastas a serem compartilhadas.

Shared/Media – Aplicações podem usar esta pasta para exibir a arte de um álbum no *Universal Volume Control* (UVC) durante a reprodução em *background* (plano de fundo).

Shared/ShellContent – Tem como função armazenar imagens de *background*. Essas imagens devem estar contidas no *Isolated Storage* sempre dentro desta pasta ou de uma subpasta.

Shared/Transfers – Aplicativos podem salvar arquivos nesta pasta em segundo plano mesmo que a aplicação não esteja sendo executada em *background*.

Figura 26 – Armazenamento no *Windows Phone*

Fonte: Elaboração própria. 2013

As pastas apresentadas anteriormente possuem características em comum de serem criadas na hora da instalação do aplicativo e passíveis de deleção a qualquer momento de seu ciclo de vida.

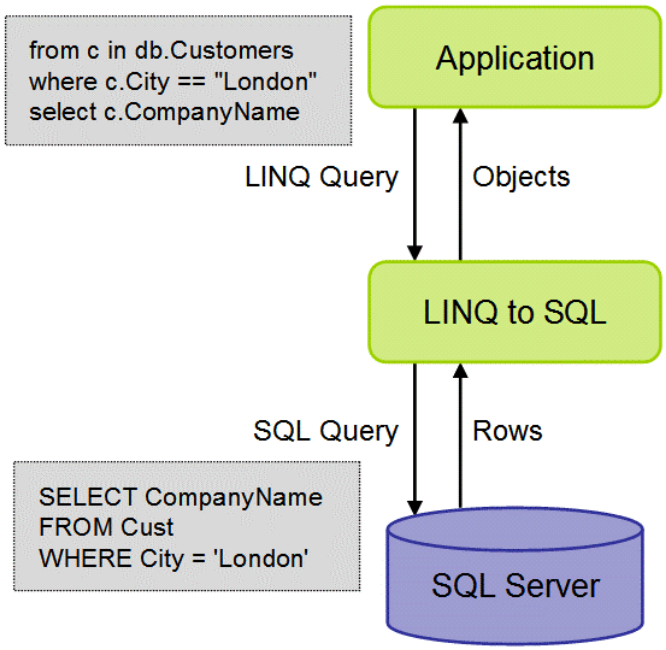
Ainda na *Persistent Data* a partir da versão *Mango* do *Windows Phone*, foi adicionado o *Local Data Base* que é na verdade uma versão própria do *Windows Phone* do Banco de dados SQL CE. Essa versão se destaca devido ao suporte a LINQ *to* SQL que provê maior produtividade para operações em base de dados.

> O local *database* permite armazenar dados relacionais dentro de um ambiente residente no *Isolated Storage*, por estar dentro do *Isolated Storage*, ele somente pode ser acessado pela aplicação correspondente, e roda somente no processo do aplicativo, não sendo executada com um serviço continuo, a manipulação de dados locais são acessados apenas através de LINQ *to* SQL, não tendo suporte ao *Transact*-SQL (SATO, 2011, MSDN).

O LINQ *to* SQL faz a conversão de consultas escritas na linguagem de programação C# ou *Visual Basic* em um SQL dinâmico. Essa conversão proporciona o mapeamento dos objetos do banco de dados e com isso gera as classes necessárias para a realização de operações usando a linguagem *Language Integrated Query* (LINQ). O LINQ *to* SQL provê a possibilidade de se efetuar o mapeamento objeto relacional, criando dessa forma as classes que representam as tabelas do banco de dados e também criando novas consultas e fazendo alterações no próprio banco.

> O LINQ (*Language Integrated Query*) é um conjunto de extensões introduzido pela *Microsoft* no. NET *Framework* 3.5 e nas novas versões das linguagens *Visual* C# e *Visual Basic* que tem como objetivo unificar o modelo de acesso à dados armazenados na memória principal de um computador ou em um repositório de dados externo como um banco de dados ou arquivos XML. Sua sintaxe é equivalente à sintaxe da linguagem SQL usada na comunicação com um sistema gerenciador de banco de dados. Usando o LINQ não é necessário escrever comandos expressos na linguagem SQL embutidos no código de acesso à dados escrito em *Visual* C# ou *Visual Basic*, isto é, pode-se escrever o código de acesso a dados diretamente em *Visual* C# ou *Visual Basic*. Além disso, os dados não precisam estar armazenados somente em um banco de dados (MAGRI, 2012, p. 157-166).

Figura 27 – Esquema de execução da arquitetura do LINQ to SQL



Fonte: Wriju. 2007

4.3.3 Análise comparativa

A persistência nas plataformas *Android* e *Windows Phone* é implementada de maneiras diferentes, pois as duas plataformas levam em consideração que cada aplicativo possui uma necessidade única de armazenamento de dados e que a padronização de um tipo específico de persistência pode causar uma inflexibilidade desnecessária à aplicação tornando-a assim menos eficiente e produtiva.

O *Android* implementa níveis diferentes de persistência levando em consideração o diretório onde aplicação será instalada e o tipo de relacionamento que a aplicação terá com outras aplicações do sistema. Já o *Windows Phone* trabalha somente com dois tipos de persistência, a *Transient Data*, onde os dados persistidos serão armazenados temporariamente na memória e a *Persistent Data* que fará a persistência independentemente do local de armazenamento e do tipo de armazenamento utilizado, seja ele através de arquivos ou banco de dados.

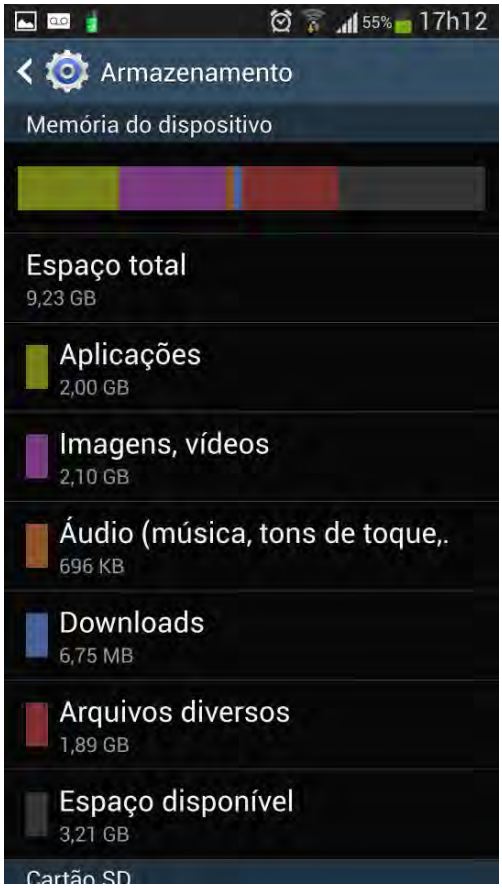
As duas plataformas garantem a restrição de acessos aos dados persistidos, mantendo desta forma a integridade dos dados dos aplicativos, estejam eles na memória interna ou externa do dispositivo.

Tanto o a plataforma *Android* quanto a plataforma *Windows Phone* fornecem ao desenvolvedor a opção de implementar a persistência utilizando bancos de dados. Tais bancos de dados são versões aperfeiçoadas e compactas de outros sistemas de bancos de dados, contudo possuem as mesmas funcionalidades.

A plataforma *Android* Utiliza o banco de dados *SQLite*, este banco é um pequeno SGBD que se comporta da mesma forma que outros mais robustos disponíveis no mercado, porém ele possui as vantagens de ser compacto, requerer uma quantidade menor de memória em tempo de execução e ser multiplataforma. Todas estas características combinadas tornam este sistema de banco de dados uma boa opção de persistência para dispositivos que possuem uma quantidade limitada de espaço em disco, memória RAM e capacidade de processamento.

A imagem a seguir retrata a subdivisão que a plataforma *Android* utiliza para apresentar ao usuário como os dados são persistidos:

Figura 28 – Subdivisão do espaço de armazenamento no *Android*.



Fonte: Elaboração própria. 2013

Já a plataforma *Windows Phone*, utiliza o *Local Data Base* que por sua vez é uma versão simplificada do banco de dados SQL CE e possui suporte ao LINQ *to* SQL. O LINQ *to* SQL atua fazendo um mapeamento objeto-relacional, isso significa que ele consegue representar através de classes as tabelas do banco tornando assim mais simples a interação entre o aplicativo e os dados persistidos. O LINQ *to* SQL também provê a funcionalidade de não ser necessário utilizar a linguagem SQL para criar comandos no banco de dados sendo assim, pode-se escrever comandos para a manipulação de dados diretamente na linguagem utilizada para desenvolver o aplicativo, tornando o código mais uniforme.

Levando em consideração todas as maneiras de se persistir dados, pode-se observar que a plataforma *Android* oferece mais recursos tanto nos tipos de armazenamento quanto na qualidade do sistema gerenciador de banco de dados. Já a plataforma *Windows Phone* não possui tantas formas de persistência, porém, trata

de separar claramente os tipos de pastas para determinados tipos de arquivos e mídias, fornece também a possibilidade de uma escrita única na criação de sua base de dados, o que simplifica o desenvolvimento da persistência em bancos de dados.

CONCLUSÃO

Tendo em vista que o atual ambiente de desenvolvimento de *softwares* está voltado para colocar todos os recursos computacionais em um só lugar de maneira que possa ser movido junto ao usuário de maneira prática e natural, e que a utilização de dispositivos móveis com acesso à *internet* está cada vez maior, buscamos através deste estudo esclarecer eventuais dúvidas que possam surgir durante a caminhada do desenvolvedor rumo ao desenvolvimento de aplicativos móveis. Sendo assim pudemos perceber o quão importante são os conceitos fundamentais da computação móvel e como conhecer as características das plataformas ajuda ao desenvolvedor a decidir sobre qual plataforma utilizar.

Podemos perceber como a compreensão das vulnerabilidades mais comuns podem afetar os dispositivos e como os aplicativos nas plataformas estudadas fazem para proteger e armazenar os dados a acessos indevidos.

Considerando o cenário apresentado neste trabalho e de acordo com os dados pesquisados quanto à sua disponibilidade, verificamos que o desenvolvedor pode ficar em um impasse ao escolher uma opção de plataforma de desenvolvimento móvel. Isto ocorre, pois algumas outras perguntas surgem ao planejar o desenvolvimento de um aplicativo, como por exemplo, qual plataforma possui maior conteúdo acessível, qual atenderia melhor os requisitos do aplicativo e qual plataforma apresenta uma gama maior de recursos disponíveis para o desenvolvimento.

De acordo com a proposta inicial deste trabalho, visando realizar uma pesquisa bibliográfica a fim de evidenciar qual destas plataformas oferece melhores condições para o desenvolvimento de uma aplicação móvel, foram pesquisados e analisados os aspectos de segurança e persistência das plataformas *Android* e *Windows Phone*. Tais aspectos foram selecionados, por serem os que mais apresentaram conteúdo de pesquisa disponível em livros, artigos e referencias da *internet* e por serem importantes dentro do contexto de desenvolvimento.

Sendo assim, chegamos à conclusão de que a plataforma *Android* se destaca em relação ao conteúdo disponível pelo fato de que por ser de código aberto, formou-se uma grande comunidade de desenvolvedores e, portanto todo o conteúdo

necessário para o desenvolvimento de um aplicativo pode ser facilmente encontrado tanto na literatura quanto está disponível *online*. Já a plataforma *Windows Phone* se mostra menos acessível para os desenvolvedores, pois ela é construída principalmente sobre tecnologias prioritárias, que por sua vez precisam de licenças com custos de registro elevado, sua comunidade de desenvolvedores é muito escassa e por vezes as informações das referências na *internet* e nos livros são desencontradas.

Tendo em vista o conteúdo estudado em relação à segurança, para o desenvolvimento de aplicativos que necessitam de maior segurança a plataforma da *Microsoft* é ideal, tanto por possuir uma melhor implementação do modelo de segurança, quanto pelo fato de que mesmo que seja feito o *jailbrake*, os aplicativos não terão acesso total as configurações de alto privilégio no sistema operacional. Outro ponto que torna a segurança na plataforma mais interessante pode também ser um pequeno transtorno para desenvolvedores, dado que o conteúdo para estudo é escasso e as ferramentas de desenvolvimento são pagas, além da comunidade de desenvolvedores ser muito escassa. Isso faz com que seja difícil para o desenvolvedor se aprofundar em todas as nuances da implementação da segurança para os dispositivos com *Windows Phone,* o que acaba deixando a cargo do SO o gerenciamento da segurança dos aplicativos. Ocorre que essa mesma dificuldade em obter o conhecimento necessário para se aprofundar na segurança da plataforma *Windows Phone* garante sua integridade, já que contamos com pouco conteúdo para a mesma.

Já a plataforma *Android* possui acesso fácil a todo seu material de implementação além da comunidade ser vasta e bem ativa, sem contar que o sistema operacional é de código aberto. Isso faz com que seja de certa maneira mais fácil e rápido implementar a segurança dos aplicativos na plataforma *Android*, porém esta por sua vez deve ser feita de maneira mais consciente e responsável já que o fato de o sistema permitir a depuração USB a instalação de fontes desconhecidas e a grande quantidade de acesso a informações de *root* torna o sistema muito mais vulnerável a ataques.

Sendo assim conclui se que a melhor opção com relação à implementação de segurança é a plataforma *Windows Phone*.

Ao analisar o aspecto de persistência, a plataforma *Android* se mostra superior em relação à plataforma *Windows Phone*. Isto devido à maior flexibilidade de implementação da persistência na plataforma *Android*, de modo que o desenvolvedor possui a opção de criar um aplicativo de forma simples e ainda conseguir definir com facilidade o local de armazenamento para os dados da aplicação, seja no armazenamento interno, externo, espaço de *cache* ou mesmo uma base de dados.

A plataforma *Android* conta também com o suporte nativo ao *SQLite*, que de acordo com o que foi apresentado neste estudo, é um SGBD completo que fornece toda a estrutura necessária para a criação de uma base de dados compacta e de rápido processamento, algo essencial visto que os dispositivos móveis ainda sofrem com a limitação de seus *hardwares*.

A plataforma *Windows Phone* trabalha somente com duas vias de implementação de sua persistência, sendo uma delas, a *Transiente Data*, que armazena somente dados que são consumidos rapidamente pelo sistema e que na *Persistent Data*, o tratamento dos dados armazenados é feito através da criação de pastas que comportam os tipos de arquivos e mídias criados pela aplicação, no entanto estas pastas devem estar contidas dentro de outra chamada *Isolated Storage*. A plataforma *Windows Phone* dentro da *Persistent Data* conta ainda com a *Local Data Base*, que no caso é somente uma versão modificada para o *Windows Phone* do antigo SQL CE.

O único diferencial da *Local Data Base* é o fato de trabalhar com o LINQ *to* SQL que provê o mapeamento objeto-relacional das tabelas do banco de dados, criando assim classes com a estrutura das tabelas do banco e de se valer da utilização da linguagem de programação comum do aplicativo ao invés de embutir comandos SQL no código, tornado desta forma a programação do aplicativo mais uniforme.

A partir das considerações aqui apresentadas, o desenvolvedor deve analisar os requisitos de sua aplicação a fim de definir se ele necessita de uma plataforma que forneça uma melhor implementação de segurança ou persistência. Se for a primeira ele deverá optar pela plataforma Windows Phone, porém se a necessidade for uma melhor implementação da persistência, é indicado o uso da plataforma Android.

Doravante, cabe ao desenvolvedor que deseja iniciar seus estudos no desenvolvimento para dispositivos móveis, analisar o conteúdo deste trabalho e criar suas próprias conclusões a respeito de qual plataforma deve ser escolhida para criar um aplicativo.

REFERÊNCIAS

ANDROID, Android.com. **About Feature Widgets**. 2013.

BOURBON, B. C. **Um Framework para desenvolvimento de aplicativos em Windows Mobile**. Trabalho de Graduação. Universidade Federal de Pernambuco. 2005.

BAUER, C.; KING, G. **Hibernate in Action**. Greenwich: Manning, 2005.

CANTÚ, E. **Redes de computadores e internet**. São José, 2003.

CROTTI, A. J. et al. **Computação Quântica**. Artigo Científico, Universidade do Extremo Sul Catarinense – UNESC.2006.

CAVALHEIRO, J; CHAVES, M. **Redes Sem Fio**. Oeiras. Piauí. 2009.

CARSON, D. **Installing Eclipse and the Android SDK on Ubuntu** 10.04/10.10. 2010.

DEITEL, H. M. **An Introduction to Operating Systems**, 2 edição. Addison-Wesley, Reading, MA, 1992.

DE ARAUJO, R. B. **Computação Ubíqua**: princípios, tecnologias e desafios. In: XXI Simpósio Brasileiro de Redes de Computadores. 2003.

DATE, C. J. **Introdução a Sistemas de Bancos de Dados**. Elsevier. Rio de Janeiro RJ. 2004.

EIRAS, M. C. **Engenharia Social e Estelionato Eletrônico. IBPI Internet School**. 2004

EXAME, Exame.com. **Crescimento móvel no Brasil ultrapassa 100%, diz estudo**. São Paulo. 2013.

FELKER, D; DOBBS, J. **Android Application Development for Dummies**. Wiley Publishing Inc. 2011.

GARGENTA, M. **Learning Android**. First Edition. O'Reilly. Sebastopol CA. 2011

GÓMEZ, F. U. **Creararchivosen Linux desde el terminal**. 2013.

SMOLKA, J. R. **Sistemas de Computação 7 - Programação e sistemas operacionais**. 2011.

KOVACS, B. P. U. **Um estudo prático das ameaças de segurança em dispositivos portáteis com Windows Mobile**. Monografia. Departamento de Informática – PUC-Rio. 2006.

KAHL, M; FLORIANO, D. **Computação Ubíqua**: tecnologia sem limites. Vale do Itajaí SC. 2012.

KAMADA, T. P. B. **Análise das Plataformas de Desenvolvimento Mobile aplicados na Área Educacional, usando Android e Windows Phone**. Estudo de Caso: Aplicativo Planetas no Windows Phone. Novas Tecnologias na Educação. Curitiba PR. 2012.

LEMOS, A. **Cibercultura e Mobilidade**: a era da conexão. Razon y palavra. Lago de Guadalupe. México. 2004.

LOPES, C. T. R. **Persistencia fácil em SQLite no Android 2.2**. 2010.

LEE, W. **Beginning Android 4 Application Development**.Wiley.Com. Indianápolis. Indiana. 2012.

LUNARDI, T. **Armazenando dados no Windows Phone com Isolated Storage**. 2012.

MATEUS, G. R; LOUREIRO, A. A. F. **Introdução à Computação Móvel**. DCC/IM, COPPE/UFRJ. Rio de Janeiro RJ. 1998.

MARTINS, R. J. W. A. **Desenvolvimento de Aplicativo para Smartphone com a Plataforma Android**. Rio de Janeiro RJ. 2009.

MSDN, msdn.microsoft. **Visual Studio 2010 Express for Windows Phone**. 2009.

MEDNIEKS, Z; DORNIN, L; MEIKE, G. B; NAKAMURA, M. **Programming Android**. First Edition. O'Reilly. Sebastopol CA. 2011.

MORIMOTO, C. E. **Redes wireless**. 2011.

MELANSON, D. **USRoboticsPalmPilotPersonalReview**. 2011.

MAGRI, J. A. **Acesso a Coleções de Dados Usando LINQ**. Augusto Guzzo Revista Acadêmica, v. 1, n. 9, p. 157-166, 2012.

NAVATHE, S. B.; ELMASRI, R. **Sistemas de Banco de Dados**. Sham, Addison. Ribeirão Preto SP. 2005.

NOVOTNY, J. **Toshiba launched the Pocket PC e335 handheld computer**. 2012.

OGLIARI, R. **Persistência de dados com Android**: Muito além do *SQLite*. Mobile Magazine. Edição 36. Ano 5. 2011.

PASSOS, T. S. **Android**: Arquitetura e Desenvolvimento. Poços de Caldas MG. 2009.

PEREIRA, L. C. O; DA SILVA, M. L. **Android para Desenvolvedores**. Brasport. Rio de Janeiro RJ. 2009.

PRESSMAN, R. S. **Engenharia de software**. McGraw Hill Brasil. New York EUA. 1995.

RODRIGUES, G. R. **Smartphones e suas tecnologias**. Trabalho de conclusão de curso. Universidade de São Paulo. 2009.

PAVANI, D. **Entenda a Cloud Computing, a computação que levará todos às nuvens**. 2011.

SACHSE, N. R. S. **Avaliação Comparativa do Modelo de Segurança do Android**. Fernando Pessoa. Paraíba. 2010.

SOUSA, F. R. C. et al. **Gerenciamento de dados em nuvem**: Conceitos, sistemas e desafios. Tópicos em sistemas colaborativos, interativos, multimídia, web e bancos de dados, Sociedade Brasileira de Computação. p. 101-130. 2010.

TANEMBAUM, A. S. **Sistemas Operacionais Modernos**, 3 edição. São Paulo: Pearson Education Inc. 2008.

TONIN, G. S; GOLDMAN, A. **Tendências em Computação Móvel**. São Paulo. 2012.

TECHLIDER, Techlider.com. **Reservas por meio de dispositivos móveis triplicam na booking.com**. 2013.

WRIJU. **LINQ to SQL Execution Architecture**. 2007.

W3RESOURCE. w3resource.com. **MySQL Database and Tables**. 2010.

ZIEGLER, C. **Palm Treo 750 running Windows Mobile 6**. 2007.

IDC: Android detém 75% do mercado mundial de smartphones. Disponível em: <http://www.androidpit.com.br/idc-android-mercado-mundial-smartphones> . Acesso em: 07 Dezembro de 2013.

APRESENTANDO LINQ To SQL. Disponível em: <http://www.macoratti.net/07/12/vbn5_lqs.htm>. Acesso em: 28 de Outubro de 2013.

ABOUT SQLite. Disponível em: <http://www.sqlite.org/about.html>. Acesso em: 15 de Outubro de 2013.

VENDAS de smartphones têm crescimento espetacular no Brasil. Disponível em: <http://exame.abril.com.br/tecnologia/noticias/8-3-milhoes-de-smartphones-sao-vendidos-no-segundo-trimestre>. Acesso em: 16 de Outubro de 2013.

CRESCIMENTO móvel no Brasil ultrapassa 100%, diz estudo. Disponível em: <http://exame.abril.com.br/tecnologia/noticias/crescimento-movel-no-brasil-ultrapassa-100-diz-estudo>. Acesso em: 16 de Outubro de 2013.

WINDOWS Phone - Novidades da versão Mango. Disponível em: <http://msdn.microsoft.com/pt-br/library/hh273474.aspx#localdb>. Acesso em: 19 de Outubro de 2013.

FILE manipulation with Isolated Storage Fileon Windows Phone. Disponível em: <http://developer.nokia.com/Community/Wiki/File_manipulation_with_IsolatedStorageFile_on_Windows_Phone>. Acesso em: 19 de Outubro de 2013.

SMARTPHONES: a história do Windows Mobile. Disponível em: <http://www.hardware.com.br/dicas/historia-windows-mobile.html>. Acesso em: 12 de Outubro de 2013.

A História da plataforma Mobile. Disponível em: <http://channel9.msdn.com/posts/A-Histria-da-plataforma-Mobile>. Acesso em: 12 de Outubro de 2013.

WINDOWS Phone 7: o revolucionário da Microsoft. Disponível em: <http://www.superdownloads.com.br/materias/windows-phone-7-revolucionario-da-microsoft.html>. Acesso em: 12 de Outubro de 2013.

SHARED Preferences. Disponível em: <http://developer.android.com/reference/android/content/SharedPreferences.html>. Acesso em: 15 de Outubro de 2013.

ANDROID SQLite data base and contente provider. Disponível em: <http://www.vogella.com/articles/AndroidSQLite/article.html>. Acesso em: 21 de Outubro de 2013.

SAVING anImage as Transient Data. Disponível em: <http://mobile.dzone.com/news/saving-image-transient-data>. Acesso em: 24 de Outubro de 2013.

WINDOWS Phone 7 for IT professional. Disponível em: <http://www.microsoft.com/en-us/download/details.aspx?id=8842>. Acesso em: 03 de Outubro de 2013.

DESENVOLVEDORES lançam primeiro *jailbreak* para *Windows Phone* 7. Disponível em: <http://idgnow.uol.com.br/mobilidade/2010/11/26/desenvolvedores-lancam-primeiro-jailbreak-para-windows-phone-7/>. Acesso em: 02 de Novembro de 2013.

SALVANDO e restaurando o Application State no Windows Phone 7. Disponível em: <http://www.windowsphonebrasil.net/windowsphonebrasil/post/2010/10/08/Salvando-e-restaurando-o-Application-State-no-Windows-Phone-7.aspx>. Acesso em: 03 de Novembro de 2013.

CRESCE o acesso à internet por dispositivos móveis. Disponível em: <http://www.agencia.fapesp.br/17463>. Acesso em: 03 de Novembro de 2013.