

UNIVERSIDADE REGIONAL DE BLUMENAU
CENTRO DE CIÊNCIAS EXATAS E NATURAIS
CURSO DE CIÊNCIA DA COMPUTAÇÃO – BACHARELADO

**ANÁLISE COMPARATIVA DE FERRAMENTAS DE
DESENVOLVIMENTO DE APLICATIVOS MÓVEIS
MULTIPLATAFORMA**

NIKSON BARTH

BLUMENAU
2014

2014/1-19

NIKSON BARTH

**ANÁLISE COMPARATIVA DE FERRAMENTAS DE
DESENVOLVIMENTO DE APLICATIVOS MÓVEIS
MULTIPLATAFORMA**

Trabalho de Conclusão de Curso submetido à
Universidade Regional de Blumenau para a
obtenção dos créditos na disciplina Trabalho
de Conclusão de Curso II do curso de Ciência
da Computação — Bacharelado.

Prof. Marcel Hugo, Mestre - Orientador

**BLUMENAU
2014**

2014/1-19

**ANÁLISE COMPARATIVA DE FERRAMENTAS DE
DESENVOLVIMENTO DE APLICATIVOS MÓVEIS
MULTIPLATAFORMA**

Por

NIKSON BARTH

Trabalho aprovado para obtenção dos créditos na disciplina de Trabalho de Conclusão de Curso II, pela banca examinadora formada por:

Presidente: _____
Prof. Marcel Hugo, Mestre – Orientador, FURB

Membro: _____
Prof. Mauro Marcelo Mattos, Doutor – FURB

Membro: _____
Prof. Cláudio Ratke, Mestre – FURB

Blumenau, 07 de julho de 2014

Dedico este trabalho a minha mãe, Ivone Barth
e a meu pai Wilson Barth.

AGRADECIMENTOS

À minha família, pois a família é o porto seguro de todos. A família é o que nos faz seguir em frente e não desistir apesar dos obstáculos no caminho.

Aos meus amigos, que no decorrer dessa jornada, foram sempre amigos, transmitindo alegria e ajudando uns aos outros, Boas amizades que duram a vida.

Ao meu orientador, Marcel Hugo, que ao longo do trabalho se mostrou sempre disposto a ajudar e encontrar a melhor solução para os problemas. Otimista sempre, me motivando e acreditando junto comigo na conclusão do trabalho.

RESUMO

Este trabalho apresenta uma análise comparativa entre as ferramentas de desenvolvimento móvel multiplataforma Delphi XE5, Xamarin e PhoneGap. Os critérios para avaliação foram determinados com base na norma NBR ISO/IEC 25000. Para permitir a comparação foram implementados dois aplicativos utilizando as ferramentas, sendo o primeiro simples e o segundo utilizando mais recursos do dispositivo, a fim de medir a diferença de benefícios que cada ferramenta oferece.

Palavras-chave: Delphi XE5. Xamarin. PhoneGap. Comparação. Desenvolvimento móvel multiplataforma.

ABSTRACT

This work presents a comparative analysis among the multiplatform mobile development tools Delphi XE5, Xamarin and PhoneGap. The evaluation criteria were based on NBR ISO / IEC 25000. To allow comparison two applications were implemented using the tools, the first being simple and the second using more features of the device in order to measure the difference in benefits of each tool.

Keywords: Delphi XE5. Xamarin. PhoneGap. Comparison. Cross platform mobile development.

LISTA DE ILUSTRAÇÕES

Quadro 1 – Modelo de qualidade da ISO 25010	17
Figura 1 – Projetos pré-construídos.....	18
Figura 2- Visual Studio e Xamarin.....	19
Figura 3 – Atualização dos sistemas atuais	20
Figura 4 – Android e iOS com aplicativo gerado em Delphi XE5.....	21
Quadro 2 – Questionário de avaliação.....	26
Figura 5 – Diagrama de casos de uso do Aplicativo 1.	27
Figura 6 - Diagrama de casos de uso do Aplicativo 2.....	28
Quadro 3 – Requisitos funcionais.....	28
Quadro 4 – Requisitos não funcionais.....	28
Figura 7 – Criação de projeto.	29
Figura 8 – Tipos de aplicação.....	30
Figura 9 – Projeto Delphi XE5.....	30
Figura 10 – Projeto do Aplicativo 1 em Android e iOS	31
Figura 11 – Gerenciador de projetos	32
Figura 12– Ícone de instalação e ícone do <i>software</i> instalado	32
Figura 13 – Execução do RAD PAServer XE5.....	33
Figura 14 – Propriedade <i>iOS Device</i>	33
Figura 15 – Adicionar SDK no Delphi XE5	34
Figura 16 – Configuração de perfil de conexão	34
Figura 17 – Finalização da configuração do perfil de conexão.....	35
Figura 18 – Conclusão da configuração de nova SDK.....	35
Figura 19 – Conclusão das propriedades da plataforma.....	36
Figura 20 – Componentes SQL no Delphi XE5	36
Quadro 5 – Linhas de código SQL no Delphi XE5.....	36
Figura 21 – Componente WSDL Importer	37
Figura 22 – Classe criada a partir da configuração do componente WSDL Importer.....	37
Quadro 6 – Utilização de um método do Web Service na ferramenta Delphi XE5	37
Figura 23 – TMotionSensor e TTimer.....	37
Quadro 7 – Exemplo de código utilizando TMotionSensor	37
Figura 24 – Componente TCameraComponent.....	38

Quadro 8 – Exemplos de utilização do TCameraComponent	39
Figura 25 – TLocationSensor	40
Quadro 9 – Utilização do TLocationSensor	40
Figura 26 – Criar projeto Android utilizando Xamarin.....	40
Figura 27 – Criar projeto iOS utilizando Xamarin.....	41
Figura 28– Configurar equipamento com o sistema OS X para construção das aplicações. ...	42
Figura 29 – Processo de configuração no sistema operacional OS X	42
Figura 30 – Início de configuração na ferramenta Visual Studio 2012.....	43
Figura 31 – Conclusão da configuração na ferramenta Visual Studio 2012	43
Figura 32 – Mobile Apps.....	44
Quadro 10 – Acelerômetro na plataforma iOS	44
Quadro 11 – Acelerômetro na plataforma Android.....	45
Quadro 12 – Utilização da câmera na plataforma iOS	45
Quadro 13 – Utilização da câmera na plataforma Android início.....	46
Quadro 14 – Utilização da Câmera na plataforma Android fim.....	46
Quadro 15 – GPS na plataforma iOS	47
Quadro 16 – GPS na plataforma Android	47
Quadro 17 – GPS na plataforma Android	48
Quadro 18 – GPS na plataforma Android	48
Figura 33 – Simulação do aplicativo 1 PhoneGap	49
Quadro 19 – Configuração do Path	50
Quadro 20 – Comandos e descrições.....	50
Figura 34 – Comando executado na plataforma Windows.....	50
Figura 35 – Comando executado na plataforma OS X.....	51
Figura 36 – Criar projeto PhoneGap a partir de um código existente	52
Figura 37 – Importar o projeto Phonegap.....	52
Quadro 21 – Uso do banco de dados com a ferramenta PhoneGap	53
Quadro 22 – Busca de dados de um Web Service com a ferramenta Phonegap	53
Quadro 23 – PhoneGap – getCurrentAcceleration	54
Quadro 24 – Phonegap – watchAccelerometer	54
Quadro 25 – Utilização da câmera com a ferramenta PhoneGap.....	55
Quadro 26 – Utilização do GPS com a ferramenta PhoneGap.....	56
Quadro 27 – Questionário de avaliação.....	58
Quadro 28 – Comparativo de características	60

LISTA DE TABELAS

Tabela 1 – Tempo em min de instalação no dispositivo Android	57
Tabela 2 – Tempo em min de instalação no emulador Android.....	57
Tabela 3 – Tempo em min de instalação no emulador iOS.....	57
Tabela 4 – Tempo em min de compilação dos projetos	57
Tabela 5 – Recurso utilizando no dispositivo Android	57
Tabela 6 – Recurso utilizando pelas ferramenta no computador.....	57

LISTA DE SIGLAS

API – Application Programming Interface

NBR – Norma Brasileira

ISO – International Organization for Standardization

IEC – International Electrotechnical Commission

SquaRE – Software Product Quality Requirements and Evaluation

APP – Application

SDK – Software development kit

IDE – Integrated development environment

HTML5 – Hypertext Markup Language versão 5

GPS – Global Positioning System

SQL – Structured Query Language

WSDL – Web Service Definition Language

SUMÁRIO

1 INTRODUÇÃO.....	12
1.1 OBJETIVOS DO TRABALHO	12
1.2 ESTRUTURA DO TRABALHO	13
2 FUNDAMENTAÇÃO TEÓRICA	14
2.1 DESENVOLVIMENTO MÓVEL MULTIPLATAFORMA.....	14
2.2 NBR ISO/IEC 25000 SQUARE: ENGENHARIA DE SOFTWARE.....	14
2.3 FERRAMENTA DE DESENVOLVIMENTO XAMARIN	18
2.4 FERRAMENTA DE DESENVOLVIMENTO PHONEGAP	19
2.5 FERRAMENTA DE DESENVOLVIMENTO DELPHI XE5	20
2.6 TRABALHOS CORRELATOS	21
2.6.1 Análise comparativa entre Groovy e Java.....	21
2.6.2 Desenvolvimento de aplicativo móvel multiplataforma integrado ao sistema de alerta de cheias da bacia do Itajaí	22
3 DESENVOLVIMENTO	24
3.1 NBR ISO/IEC 25000.....	24
3.2 MEIO DE AVALIAÇÃO DOS CRITÉRIOS	25
3.3 ESPECIFICAÇÃO	27
3.4 IMPLEMENTAÇÃO	29
3.4.1 Delphi XE5	29
3.4.2 Xamarin.....	40
3.4.3 PhoneGap	49
3.5 RESULTADOS E DISCUSSÃO	56
3.5.1 Resultado do questionário de avaliação	56
4 CONCLUSÕES.....	61
4.1 EXTENSÕES	62
REFERÊNCIAS	63

1 INTRODUÇÃO

Quando um aplicativo é desenvolvido para um público alvo, não se tem controle sobre quais dispositivos seu público utiliza. Logo, é aparente a necessidade de muitas empresas criar um aplicativo multiplataforma, ou seja, que funcione corretamente nas diversas plataformas existentes (Android, iOS, Windows Phone etc.) (NUNES, 2013).

Desenvolver um aplicativo para as diversas plataformas, com o intuito de atender a maioria dos dispositivos, requer retrabalho, pois é grande a diversidade de plataformas e linguagens utilizadas no desenvolvimento de aplicações nativas (NUNES, 2013).

Em tais circunstâncias, uma decisão deve ser tomada: desenvolver uma aplicação distinta para cada plataforma que se quer atender, ou utilizar uma ferramenta de desenvolvimento móvel multiplataforma para desenvolver este aplicativo, dessa forma, desenvolver o aplicativo uma única vez? Para auxiliar na decisão, empresas estão trabalhando à frente e oferecendo este tipo de ferramentas de desenvolvimento, entre elas estão: Rhodes, Appcelerator, JQuery Mobile, XUI, PhoneGap (GUINThER, 2011).

Segundo Gesmar Júnior (2013), o HTML5 pode ser uma solução quando se fala em desenvolvimento móvel multiplataforma. No entanto, Jomar Silva (2013) afirma que aplicações em HTML5 possuem limitações como: não ter total acesso ao hardware do dispositivo via *Application Programming Interface* (API) e não poder acessar diretamente o sistema operacional do dispositivo.

Diante do exposto, uma análise comparativa de ferramentas de desenvolvimento móvel multiplataforma, com base na NBR ISO/IEC 25000 SQuaRE, é desenvolvida nesse trabalho. O principal intuito é identificar as características de cada ferramenta, para com isso facilitar o entendimento e alcance das ferramentas. As ferramentas selecionadas para a comparação são Delphi XE5, Xamarin e PhoneGap.

1.1 OBJETIVOS DO TRABALHO

O objetivo desse trabalho é realizar uma análise comparativa entre ferramentas de desenvolvimento de aplicativos móveis multiplataforma.

Os objetivos específicos do trabalho são:

- a) definir os critérios para a análise comparativa, baseados na norma NBR ISO/IEC 25000;
- b) especificar um aplicativo que utilize o banco de dados do dispositivo e uma conexão com Web Service, a fim de simular uma aplicação comercial simples;
- c) especificar um segundo aplicativo que utilize funcionalidades mais específicas do

dispositivo como: acelerômetro, GPS, Bluetooth, giroscópio, câmera, com o propósito de simular uma aplicação mais elaborada;

- d) desenvolver os aplicativos especificados em todas as ferramentas selecionadas para a análise comparativa, observando os critérios elencados;
- e) buscar informações sobre o custo financeiro das ferramentas utilizadas.

1.2 ESTRUTURA DO TRABALHO

O texto está estruturado em quatro capítulos. No segundo capítulo está a fundamentação teórica do trabalho que consiste em apresentar a norma NBR ISO/IEC 25000 SQuaRE, utilizada como base na análise comparativa, junto com as características do desenvolvimento móvel multiplataforma. O capítulo apresenta também as ferramentas Delphi XE5, Xamarin e PhoneGap.

No capítulo seguinte, é apresentada a definição dos critérios de avaliação e o questionário desenvolvido para a avaliação. O capítulo também apresenta o desenvolvimento da análise comparativa, que consiste na especificação de requisitos, diagramas de casos de uso, as ferramentas que serão avaliadas e a implementação.

Por fim, é apresentada a aplicação do questionário de avaliação, assim como os resultados obtidos a partir do questionário.

No último capítulo são apresentadas as conclusões e sugestões para trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

No presente capítulo alguns aspectos teóricos relacionados ao trabalho são apresentados. A seção 2.1 consiste no estudo inicial sobre o desenvolvimento móvel multiplataforma. Na seção 2.2 é apresentada a norma NBR ISO/IEC 25000 SQuaRE e suas definições. Em sequência, a seção 2.3 apresenta uma visão global sobre a ferramenta de desenvolvimento Xamarin. A seção 2.4 aborda a ferramenta de desenvolvimento PhoneGap, enquanto a seção 2.5 apresenta a ferramenta de desenvolvimento Delphi XE5. Finalizando o capítulo, a seção 2.6 descreve os trabalhos correlatos.

2.1 DESENVOLVIMENTO MÓVEL MULTIPLATAFORMA

Um aplicativo ser multiplataforma significa que irá funcionar em diferentes dispositivos com seus respectivos sistemas operacionais (FERREIRA, 2012).

A opção por criar um aplicativo multiplataforma baseia-se em não ter a necessidade de reescrever o mesmo código para cada plataforma, tornando maior o foco nas regras de negócio da aplicação (RIBEIRO, 2012).

Algumas vantagens do desenvolvimento móvel multiplataforma são (CYGNIS MEDIA, 2013):

- a) maior alcance de consumidores;
- b) um código para muitas plataformas;
- c) facilidade de desenvolvimento;
- d) redução no custo do desenvolvimento.

Também podem ser listadas algumas desvantagens, tais como (CYGNIS MEDIA, 2013):

- a) possíveis particularidades em relação à interface de usuário para cada plataforma;
- b) dificuldades no uso de recursos nativos de cada plataforma;
- c) dificuldade para atingir desempenho semelhante entre plataformas;
- d) perda da flexibilidade para cada plataforma.

2.2 NBR ISO/IEC 25000 SQUARE: ENGENHARIA DE SOFTWARE

Dentro da engenharia de software a qualidade de software é uma área que visa garantir bons produtos a partir de bons processos. Então, dois aspectos da qualidade são ressaltados: a qualidade do produto em si e a qualidade do processo. Embora um bom processo não possa garantir a produção de um bom produto, no geral admite-se que uma equipe com um bom

processo produzirá um produto melhor do que se não tivesse processo algum. (WAZLAWICK, 2011)

O modelo de qualidade SQuaRE avalia quatro tipos de indicadores de qualidade (WAZLAWICK, 2011):

- a) medidas de qualidade do processo;
- b) medidas de qualidade internas;
- c) medidas de qualidade externas;
- d) medidas de qualidade do software em uso.

As qualidades internas permitem que o desenvolvimento do produto seja feito de forma eficiente e atinja seus objetivos. Enquanto as qualidades externas e de uso permitem que o usuário final atinja seus objetivos (WAZLAWICK, 2011).

O modelo SQuaRE é um conjunto de normas, dividido da seguinte forma (WAZLAWICK, 2011):

- a) ISO/IEC 2500n – Divisão gestão de qualidade;
- b) ISO/IEC 2501n – Divisão modelo de qualidade;
- c) ISO/IEC 2502n – Divisão medição da qualidade;
- d) ISO/IEC 2503n – Divisão requisitos de qualidade;
- e) ISO/IEC 2504n – Divisão avaliação da qualidade.

A divisão de gestão de qualidade apresenta os modelos comuns, padrões básicos, termos e definições usadas pela SQuaRE. O modelo de qualidade apresenta as características e subcaracterísticas de qualidade interna, externa e de uso do software. Os padrões de medidas de qualidade cobrem as definições matemáticas e o detalhamento da aplicação de medidas práticas de qualidade interna, externa e de uso. A divisão de requisitos de qualidade contém o padrão de especificação de requisitos de qualidade, tanto para elicitação dos requisitos de qualidade quanto a entrada para o processo de avaliação da qualidade do software. A divisão de avaliação da qualidade provê ferramentas para avaliar a qualidade de um software tanto por desenvolvedores, compradores ou avaliadores independentes (WAZLAWICK, 2011).

O modelo de qualidade da ISO/IEC 25010 define um conjunto de oito características, subdivididas em subcaracterísticas e mais cinco características de software em uso (WAZLAWICK, 2011).

O modelo resultante é mostrado no quadro 1. As características internas e externas fazem parte do conjunto de características do produto, pois sua avaliação pode ser feita no

ambiente de desenvolvimento. Já as características do software em uso só podem ser avaliadas no contexto de uso do sistema (WAZLAWICK, 2011).

Quadro 1 – Modelo de qualidade da ISO 25010

Características do produto	Adequação funcional	Completude funcional
		Corretude funcional
		Funcionalidade apropriada
	Confiabilidade	Maturidade
		Disponibilidade
		Tolerância a falhas
		Recuperabilidade
	Usabilidade	Apropriação reconhecível
		Inteligibilidade
		Operabilidade
		Proteção contra erro de usuário
		Estética de interface com usuário
		Acessibilidade
	Eficiência de desempenho	Comportamento em relação ao tempo
		Utilização de recursos
		Capacidade
	Segurança	Confidencialidade
		Integridade
		Não repúdio
		Rastreabilidade de uso
		Autenticidade
	Compatibilidade	Coexistência
		Interoperabilidade
	Capacidade de manutenção	Modularidade
		Reusabilidade
		Analisabilidade
		Modificabilidade
Testabilidade		
Portabilidade	Adaptabilidade	
	Instalabilidade	
	Substituibilidade	
Características do uso	Efetividade	Efetividade
	Eficiência	Eficiência
	Satisfação	Utilidade
		Prazer
		Conforto
		Confiança
	Uso sem riscos	Mitigação de risco econômico
		Mitigação de risco a saúde e segurança
		Mitigação de risco ambiental
	Cobertura de contexto	Completude de contexto
		Flexibilidade

Fonte: adaptado de Wazlawick (2011, p.233).

2.3 FERRAMENTA DE DESENVOLVIMENTO XAMARIN

O Xamarin foi criado com o intuito de disponibilizar uma maneira melhor para criar aplicativos, escrevendo um código C#, com base e interfaces de usuário distintas e específicas de cada plataforma. O produto promete *apps* nativos, com interfaces nativas, que partilham lógica de negócio, acesso a dados, código de rede etc., nas plataformas Android, iOS e Windows Phone (XAMARIN INC, 2014).

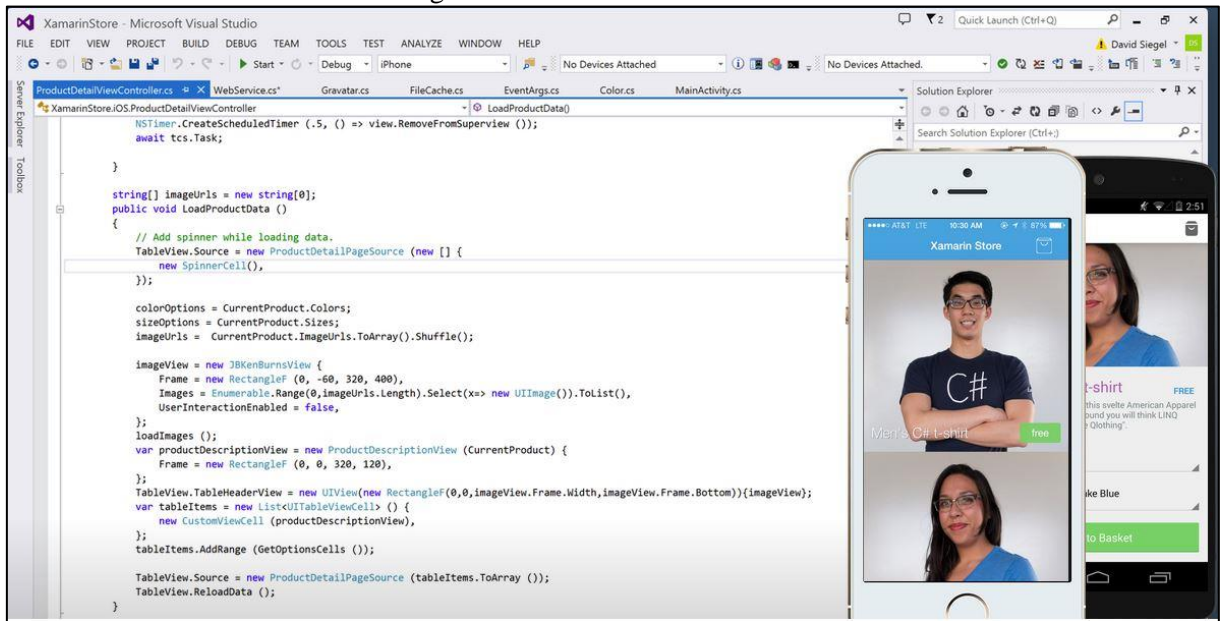
A figura 1 representa dois projetos pré-construídos disponíveis no site do fabricante. Enquanto a figura 2 demonstra a ferramenta sendo utilizada em conjunto com o Visual Studio (XAMARIN INC, 2014).

Figura 1 – Projetos pré-construídos



Fonte: Xamarin Inc (2014).

Figura 2- Visual Studio e Xamarin



Fonte: Xamarin Inc (2014).

O fabricante afirma que a compilação nativa rende desempenho inflexível até para os cenários mais exigentes, incluindo jogos e visualização de dados.

Xamarin.iOS executa uma compilação prévia para produzir o binário ARM para a App Store da Apple. Xamarin.Android usa a compilação *just-in-time* para otimizações sofisticadas. Em ambos os casos, o aplicativo é um binário nativo, e não interpretado. Isso faz com que o desempenho da ferramenta seja elevado, assim como o aplicativo final (XAMARIN INC, 2014).

2.4 FERRAMENTA DE DESENVOLVIMENTO PHONEGAP

O PhoneGap é uma ferramenta destinada a construção rápida e multiplataforma de aplicativos móveis usando HTML5, Javascript e CSS (ADOBE SYSTEMS INC, 2014).

A ferramenta teve início no ano de 2012, dentro da Apache Software Foundation. É uma ferramenta livre e de código aberto sob a licença da Apache (ADOBE SYSTEMS INC, 2014).

PhoneGap é indicada para desenvolvedores de aplicativos móveis que pretendem estender seus aplicativos a mais plataformas; desenvolvedores web que desejam partir para o desenvolvimento móvel; assim como desenvolvedores de aplicativos móveis que têm interesse em misturar componentes nativos com recursos web (ADOBE SYSTEMS INC, 2014).

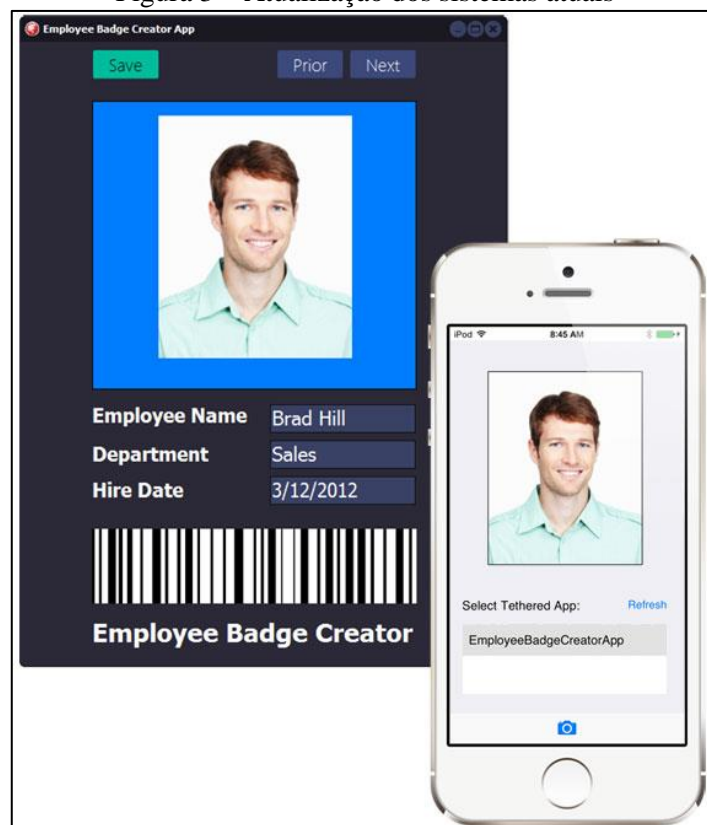
O PhoneGap gera código multiplataforma para Android, iOS, webOS, Windows Phone, Symbian, BlackBerry e bada. Para isso a ferramenta precisa ser utilizada em conjunto com outra, por exemplo: Eclipse e Xcode (ADOBE SYSTEMS INC, 2014).

2.5 FERRAMENTA DE DESENVOLVIMENTO DELPHI XE5

A ferramenta Delphi XE5 da Embarcadero promete ser o modo mais rápido para desenvolver aplicações verdadeiramente nativas para Android, iOS etc. a partir de um único código fonte. Segundo o fabricante o ambiente de desenvolvimento aumenta a produtividade dos desenvolvedores e é compatível com todos os recursos disponíveis nos dispositivos das plataformas Android e iOS e possui conectividade universal a banco de dados (EMBARCADERO TECHNOLOGIES INC, 2014).

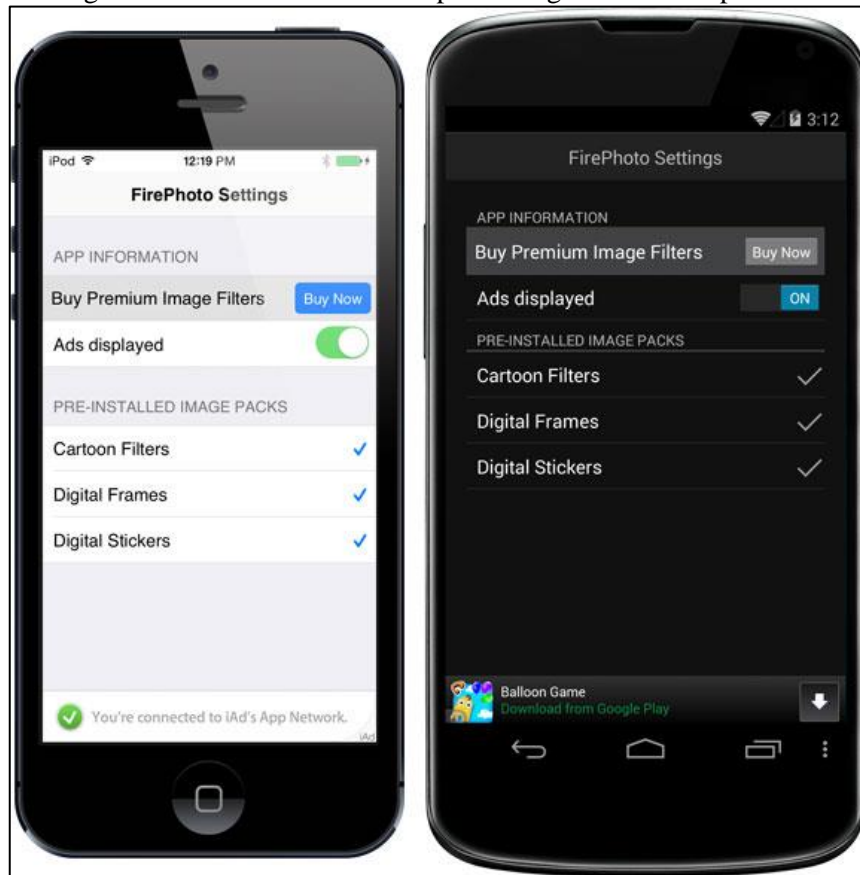
O Delphi XE5 apresenta a proposta de atualizar os sistemas desenvolvidos na linguagem Delphi, propondo adicionar aos sistemas funcionalidades de dispositivos móveis nas plataformas Android e iOS. As figuras 3 e 4 ilustram essa proposta.

Figura 3 – Atualização dos sistemas atuais



Fonte: Embarcadero Technologies Inc (2014).

Figura 4 – Android e iOS com aplicativo gerado em Delphi XE5



Fonte: Embarcadero Technologies Inc (2014).

2.6 TRABALHOS CORRELATOS

Dentre os trabalhos encontrados na pesquisa realizada, os relacionados a seguir contém temas correlatos. O primeiro apresenta a análise comparativa feita por Rezende (2011) entre as linguagens de programação Groovy e Java. Na sequência é apresentado o trabalho de Souza (2012), o desenvolvimento de aplicativo móvel multiplataforma integrado ao sistema de alerta de cheias da bacia do Itajaí.

2.6.1 Análise comparativa entre Groovy e Java

Rezende (2011) apresenta uma análise comparativa de produtividade no desenvolvimento de softwares web, entre as linguagens de programação Groovy e Java. Para determinar os critérios de avaliação ele utiliza como base a norma NBR 13596.

Para efetuar a comparação, o autor implementou um aplicativo para estudo de caso em ambas linguagens, a fim de medir a diferença de produtividade no desenvolvimento dos casos de usos. Também foi efetuado o cálculo de Pontos de Caso de Uso (*Use Case Points* – UCP) dos casos de uso definidos, além da análise e aplicação da norma NBR 13596.

Rezende (2011) propõe um meio de avaliação para os critérios, onde classifica as características em dois grupos distintos, denominados de “estático” e “dinâmico”. O grupo

“estático” reúne as características que dispensam implementações para a realização da comparação. Neste grupo estão atribuídos os seguintes critérios: usabilidade, manutenibilidade e confiabilidade. Os critérios “dinâmicos” incluem os itens eficiência e produtividade, que por sua vez são calculados com base nos valores mensurados utilizando os UCP.

Rezende (2011) conclui apresentando os resultados do questionário de avaliação baseado na norma NBR 13596, o cálculo da produtividade por UCP e um comparativo de desempenho dos estudos de caso. Por fim, afirma que:

- a) Groovy é 35% mais produtivo para web que Java;
- b) Java é aproximadamente 10% mais veloz no processamento das páginas web;
- c) Java consome menos de 50% da memória utilizada pelo Groovy na execução dos mesmos casos de uso;
- d) Groovy aloca cerca de 2500 classes a mais que Java.

2.6.2 Desenvolvimento de aplicativo móvel multiplataforma integrado ao sistema de alerta de cheias da bacia do Itajaí

Souza (2012) apresenta o desenvolvimento de um aplicativo multiplataforma, que informa em um mapa interativo, os últimos dados registrados pelo sistema de alerta de cheias da bacia do Itajaí. É composto por um aplicativo cliente desenvolvido no *framework* Titanium SDK.

Este trabalho foi desenvolvido em dois projetos distintos. Em um projeto, o aplicativo cliente foi desenvolvido de forma independente. No outro, o servidor foi desenvolvido de modo a atender as funcionalidades do primeiro, afirma o autor.

Segundo Souza(2012), as funcionalidades principais do aplicativo cliente estão relacionadas a visualização das informações dos registros coletados nas estações do sistema de alerta, a visualização e registro de ocorrência e a integração com a rede social Facebook.

Um mapa com marcadores em formato de bandeiras é apresentado na tela inicial, representando as estações do sistema de alerta do CEOPS, posicionados segundo suas respectivas geolocalizações. Além destas marcações há também as ocorrências registradas pelos usuários, representadas por ícones azuis, que no Facebook são apresentadas em forma de postagem no mural do grupo TCC Carlos Souza (2012), explica o acadêmico.

Souza (2012) relata um problema em relação a funcionalidade da utilização da câmera do dispositivo através da ferramenta Titanium SDK. Ele afirma que existe uma instabilidade, que foi informada aos desenvolvedores e mantenedores do *framework*. Portanto a foto da

ocorrência passou a ser capturada da galeria de imagens do dispositivo. Por outro lado, afirma que, após realizar testes do aplicativo cliente em dispositivos e simuladores das plataformas Android e iOS, concluiu que não houve alteração da interface gráfica, desde que estivessem utilizando a mesma versão do sistema operacional móvel em que foi desenvolvida a aplicação.

O autor ainda relata outras limitações no ambiente de desenvolvimento. O *framework* Titanium, na versão 2.1.2, não dá suporte à implementação de gráficos, assim como apresenta instabilidade em relação ao uso da câmera. Souza também aponta a ausência de alertas como uma das limitações do trabalho, principalmente para informar a desatualização dos dados exibidos ao usuário.

O acadêmico afirma que o *framework* Titanium mostrou-se bastante amplo. Apesar de suas limitações citadas anteriormente, a metodologia e padronização utilizadas no desenvolvimento nesse ambiente justificam a sua utilização. Ressalta também a ampla documentação e o fato de o ambiente possuir um SDK fortemente integrado ao Eclipse IDE.

3 DESENVOLVIMENTO

Este capítulo apresenta o processo desenvolvimento da análise comparativa, onde foram seguidos os seguintes passos:

- a) estudo da norma NBR ISO/IEC 25000 e desenvolvimento dos critérios de avaliação;
- b) definição do meio de avaliação dos critérios;
- c) especificação do casos de uso dos aplicativos;
- d) implementação dos aplicativos;
- e) resultado da análise comparativa.

3.1 NBR ISO/IEC 25000

A NBR ISO/IEC 25000 serve de referência básica para a avaliação da qualidade de software. Define qualidade através de características que são, por sua vez, subdivididas em subcaracterísticas, conforme apresentado no quadro 1 (seção 2.2).

Então, tornou-se necessário elencar as subcaracterísticas relevantes à comparação deste trabalho. Foram definidas as seguintes subcaracterísticas como critérios de avaliação, a serem aplicados na análise comparativa:

- a) adequação funcional: nas ferramentas de desenvolvimento móvel multiplataforma pode-se avaliar se o mesmo código é compilado para as plataformas Android e iOS, assim como o quanto a interface gráfica da ferramenta corresponde ao apresentado no dispositivo;
- b) confiabilidade: pode-se avaliar o funcionamento sem problemas durante a codificação, a disponibilidade das ferramentas, assim como a tolerância a falhas e recuperabilidade;
- c) usabilidade: a apropriação das ferramentas, facilidade de aprendizagem dos conceitos, a facilidade de usar e controlar as ferramentas, assim como a proteção contra erros de usuário;
- d) eficiência de desempenho: pode-se avaliar quanto tempo o software consome para processar suas funções, avaliando também se a utilização de recursos é aceitável, assim como a capacidade do produto de atender os requisitos;
- e) compatibilidade: a capacidade das ferramentas de coexistir e a capacidade de interagir com outros sistemas conforme esperado;
- f) portabilidade: pode-se avaliar se as ferramentas são facilmente adaptadas em outros ambientes;

- g) efetividade: avalia se o software permite o desenvolvimento dos aplicativos especificados por completo;
- h) eficiência: avalia se a compilação é rápida, assim como se a instalação em dispositivos e emulador é rápida;
- i) satisfação: pode-se avaliar se o usuário possui satisfação em utilizar as ferramentas, assim como o conforto na utilização;
- j) uso sem riscos: pode-se avaliar se as ferramentas minimizam os riscos financeiros;
- k) cobertura de contexto: avalia a completude de contexto das ferramentas, assim como a flexibilidade.

3.2 MEIO DE AVALIAÇÃO DOS CRITÉRIOS

Estabelecidas as ferramentas de desenvolvimento móvel multiplataformas Delphi XE5, Xamarin e PhoneGap, foi proposto um meio de avaliação com base em subcaracterísticas. Para isso foi desenvolvido o questionário de avaliação apresentado no quadro 2. Em destaque no quadro estão as subcaracterísticas identificadas como subjetivas, pois podem ter avaliações diferentes para cada avaliador. As respostas tem peso 0, 5 e 10 que significam respectivamente “Não”, “Parcial” e “Sim”. Para as questões subjetivas os valores são 0, 2.5 e 5, diminuindo seu peso relativo em função da subjetividade.

Quadro 2 – Questionário de avaliação

Adequação funcional	Completude funcional	Compila o mesmo código para as plataformas Android e iOS?
		Compila no emulador?
		Compila no dispositivo?
		Executa a construção rapidamente?
	Corretude funcional	Possui suporte e documentação?
		A interface gráfica corresponde ao apresentado no dispositivo?
	Funcionalidade apropriada	O código funciona corretamente?
O software possui <i>IntelliSense</i> ?		
Confiabilidade	Maturidade	O software possui uma interface de modelagem?
	Disponibilidade	Funciona sem problemas durante a codificação?
	Tolerância a falhas	Quando necessário, o software está operacional?
		Se algo inesperado ocorrer durante a instalação no dispositivo, o software continua funcionando?
	Se algo inesperado ocorrer durante a instalação no emulador, o software continua funcionando?	
Recuperabilidade	Quando ocorre um defeito, o software consegue recuperar os dados que estavam sendo trabalhados?	
Usabilidade	Apropriação reconhecível	O software é apropriado para o desenvolvimento dos aplicativos especificados?
	Inteligibilidade	Os conceitos-chave do software são de fácil aprendizagem?
	Operabilidade	É fácil instalar o aplicativo no emulador?
		É fácil instalar o aplicativo no dispositivo?
		É intuitiva a criação de projetos?
	Proteção contra erro de usuário	O software auxilia o usuário a não cometer erros?
	Estética de interface com usuário	A interface do software proporciona prazer e interação satisfatórios?
Acessibilidade	O software foi projetado para atender usuários com necessidades especiais?	
Eficiência de desempenho	Comportamento em relação ao tempo	Tempo necessário para compilar o projeto.
		Tempo necessário para instalar no dispositivo.
		Tempo necessário para instalar no emulador.
	Utilização de recursos	Recursos utilizados no computador.
		Recursos utilizados com emulador.
		Recursos utilizados no dispositivo.
Capacidade	O software atendeu todos os requisitos especificados nos aplicativos?	
Compatibilidade	Coexistência	O software é capaz de coexistir com outros softwares do computador?
		O software é capaz de coexistir com outros softwares do dispositivo?
	Interoperabilidade	O software é capaz de interagir com o Mac OS? (Necessário para compilar para iOS)
Portabilidade	Adaptabilidade	O software funciona na plataforma Windows?
		O software funciona na plataforma Mac OS?
	Instalabilidade	O software é facilmente instalado?
Substituibilidade	O software pode substituir outro?	
Efetividade	Efetividade	O software permite o desenvolvimento dos aplicativos

		especificados por completo?
Eficiência	Eficiência	A compilação é rápida? Ou faz o usuário esperar?
Satisfação	Utilidade	O usuário possui satisfação ao utilizar o software?
	Prazer	O usuário sente prazer em usar o software?
	Conforto	O uso de funções e componentes simples é de fácil acesso?
Cobertura de contexto	Completeness de contexto	O software pode ser usado com efetividade, eficiência, sem riscos e com satisfação em todos os contextos especificados?
	Flexibilidade	O software pode ser usado com efetividade, eficiência, sem riscos e com satisfação em todos os contextos especificados?

Para avaliar as subcaracterísticas “Comportamento em relação ao tempo” e “Utilização de recursos”, será contruída uma tabela comparativa.

Após o desenvolvimento, são obtidos os resultados de cada aplicativo em cada ferramenta, podendo assim compará-los.

3.3 ESPECIFICAÇÃO

Os aplicativos desenvolvidos para a execução da análise comparativa, seguem os casos de uso ilustrados nas figuras 5 e 6, seguidos dos requisitos funcionais e não funcionais.

Figura 5 – Diagrama de casos de uso do Aplicativo 1.

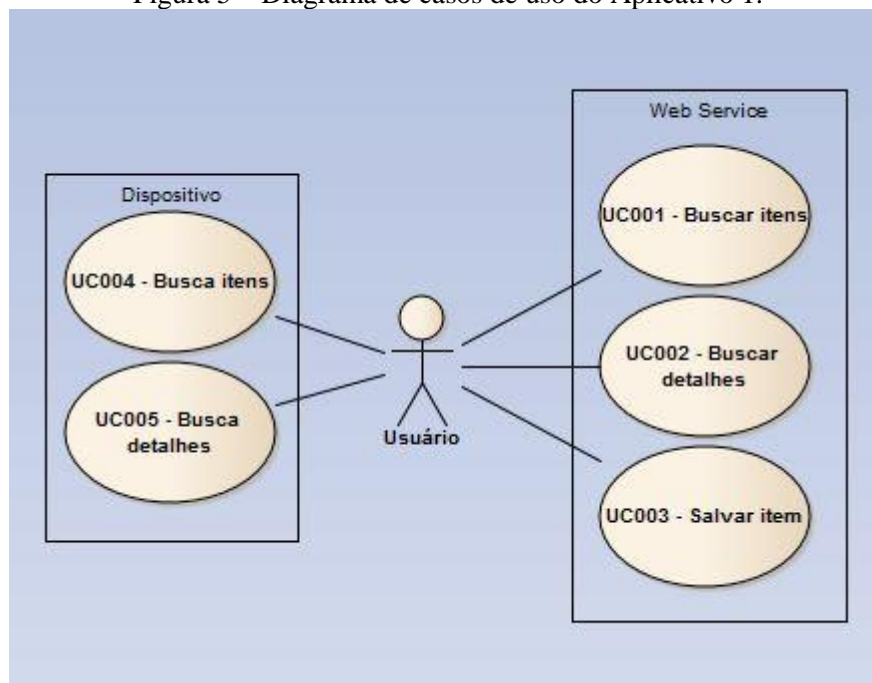
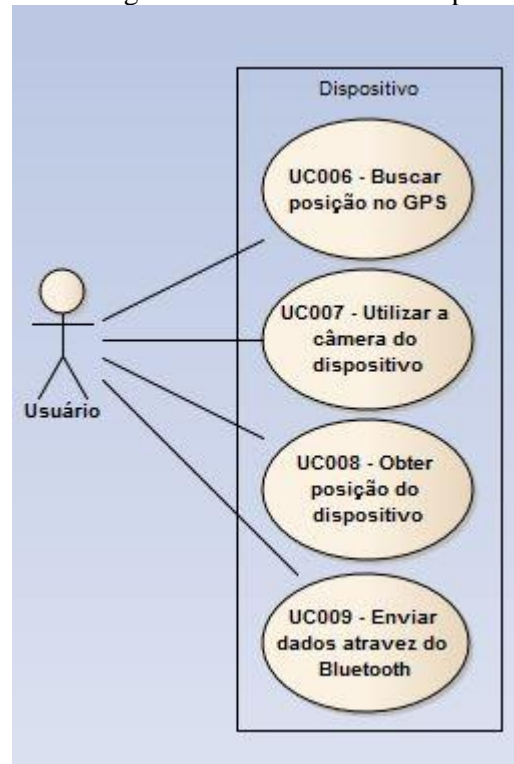


Figura 6 - Diagrama de casos de uso do Aplicativo 2.



Quadro 3 – Requisitos funcionais

REQUISITOS FUNCIONAIS	CASO DE USO
RF001 – Buscar registros em um Web Service	UC001 e UC002
RF002 – Inserir registro no banco de dados do dispositivo móvel	UC003
RF003 – Buscar registros no banco de dados do dispositivo móvel	UC004 e UC005
RF004 – Buscar geolocalização do dispositivo através do GPS	UC006
RF005 – Realizar o envio de dados através do recurso Bluetooth do dispositivo móvel	UC009
RF006 – Buscar a posição do dispositivo móvel utilizando os recursos acelerômetro e giroscópio.	UC008
RF007 – Efetuar a captura de uma imagem utilizando a câmera do dispositivo.	UC007

Quadro 4 – Requisitos não funcionais

REQUISITOS NÃO FUNCIONAIS
RNF001 – Utilizar ferramentas de desenvolvimento móvel multiplataforma para desenvolver os aplicativos que serão utilizados nas análises comparativas
RNF002 – Utilizar a norma NBR ISO/IEC 25000 SQuaRE para elencar os critérios que serão avaliados na análise comparativa
RNF003 – Efetuar a análise comparativa entre as ferramentas de desenvolvimento móvel multiplataforma.

Em relação a RF005 e o UC009, foi identificado a possibilidade de utilizar o recurso Bluetooth em todas as ferramentas analisadas. Porém como este mostrou-se ser uma tarefa complexa que comprometeria o cronograma do projeto, optou-se por incluí-la nas extensões do presente trabalho.

3.4 IMPLEMENTAÇÃO

Efetuada pesquisa sobre as ferramentas de desenvolvimento móvel multiplataforma presentes no mercado, foram selecionadas três para que fosse realizada a análise comparativa. São elas: Delphi XE5, Xamarin e PhoneGap.

3.4.1 Delphi XE5

A avaliação da ferramenta é iniciada a partir da sua instalação. O Delphi XE5 é de fácil instalação. Está disponível no site da Embarcadero (<http://www.embarcadero.com/br/products/delphi>). Basta obter o instalador e seguir os passos indicados por ele. Para a análise comparativa, a ferramenta foi instalada na plataforma Windows, versão 8.1.

Ao iniciar a ferramenta, o usuário já percebe que há um tutorial de como realizar o desenvolvimento em Android e iOS.

A criação de projetos móvel multiplataforma é bastante simplificada, como apresenta a figura 7. A figura 8 demonstra os tipos de aplicação que o usuário pode optar. Após efetuar a seleção do tipo de aplicação, o usuário se depara com o projeto, ilustrado na figura 9. Nesse momento está criado o projeto que produzirá código para as plataformas Android e iOS.

Figura 7 – Criação de projeto.

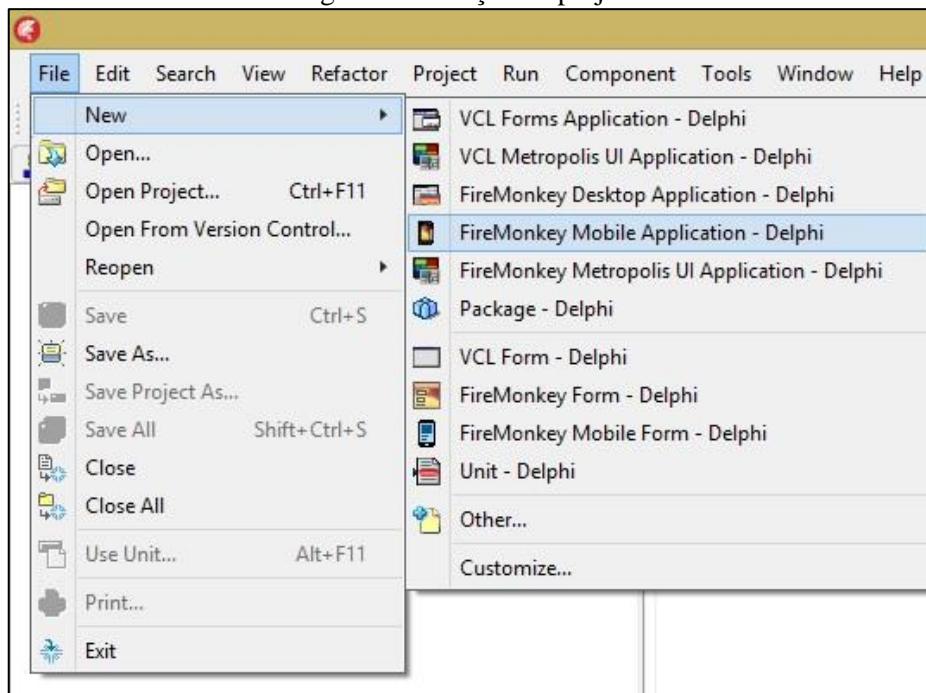


Figura 8 – Tipos de aplicação

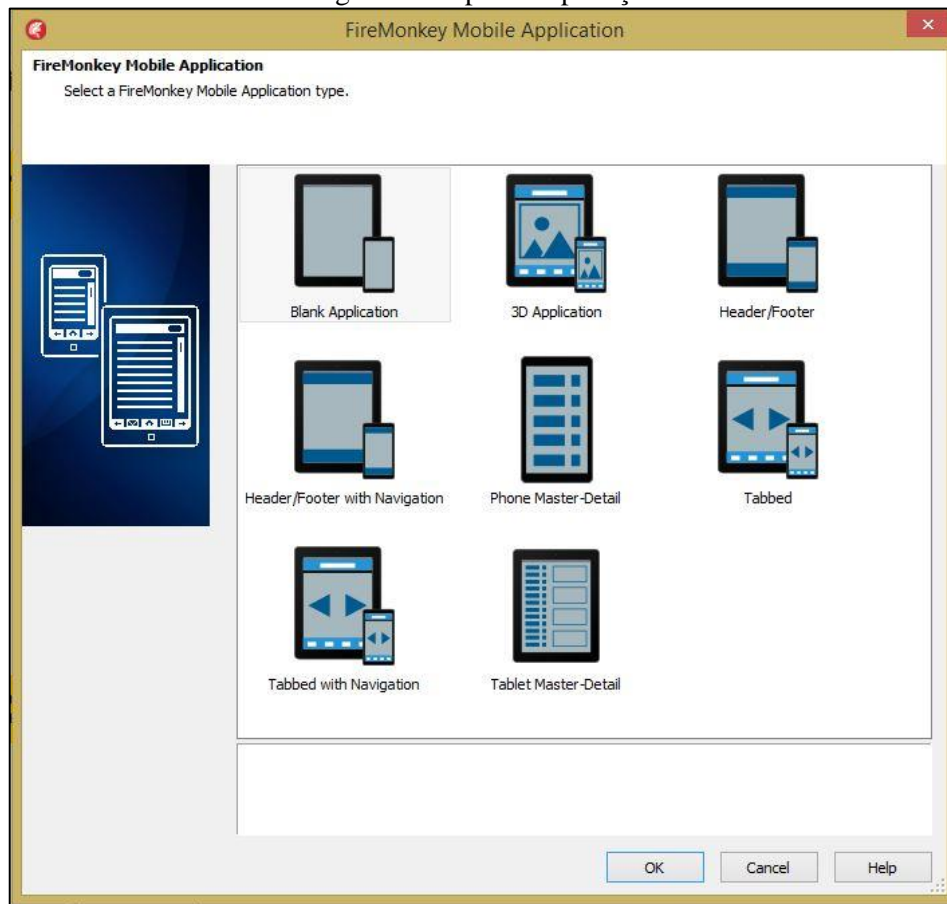
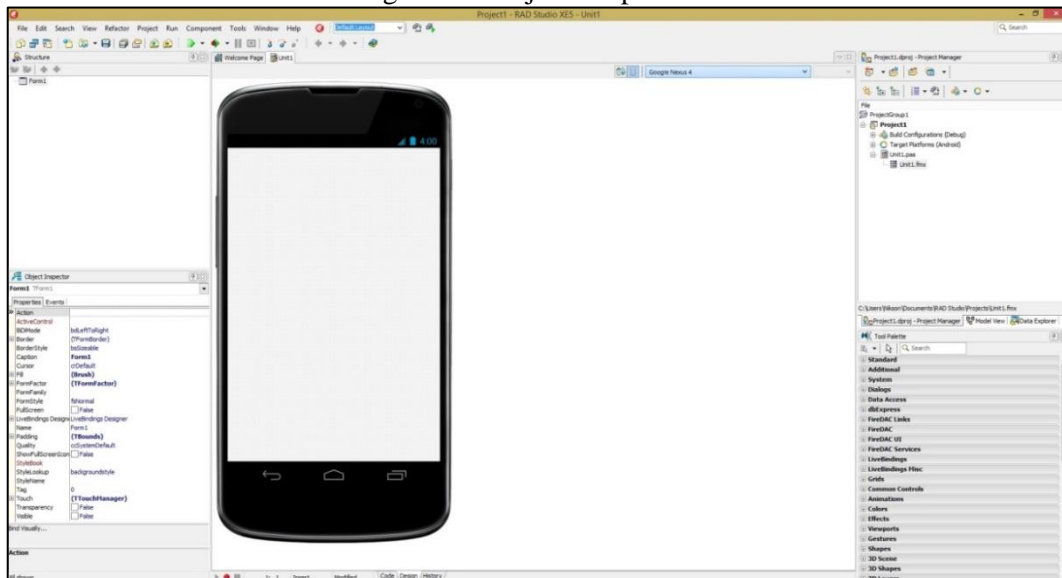


Figura 9 – Projeto Delphi XE5



A figura 10 ilustra o aplicativo 1 (casos de uso na figura 5) em ambas plataformas. É possível perceber que mesmo o projeto sendo compatível com as plataformas, é necessário realizar alguns ajustes em relação ao tamanho da tela e plataforma.

O gerenciador de projetos do Delphi XE5 contém as configurações de *build*, as telas e classes do projeto, assim como as plataformas em que o projeto pode ser executado, podendo o usuário optar por simuladores ou dispositivos físicos (figura 11). Observe também que para a plataforma Android, os simuladores e dispositivos físicos se encontram no mesmo local, já para o iOS, são locais diferentes. Porém para ambos o uso é o mesmo.

Figura 10 – Projeto do Aplicativo 1 em Android e iOS

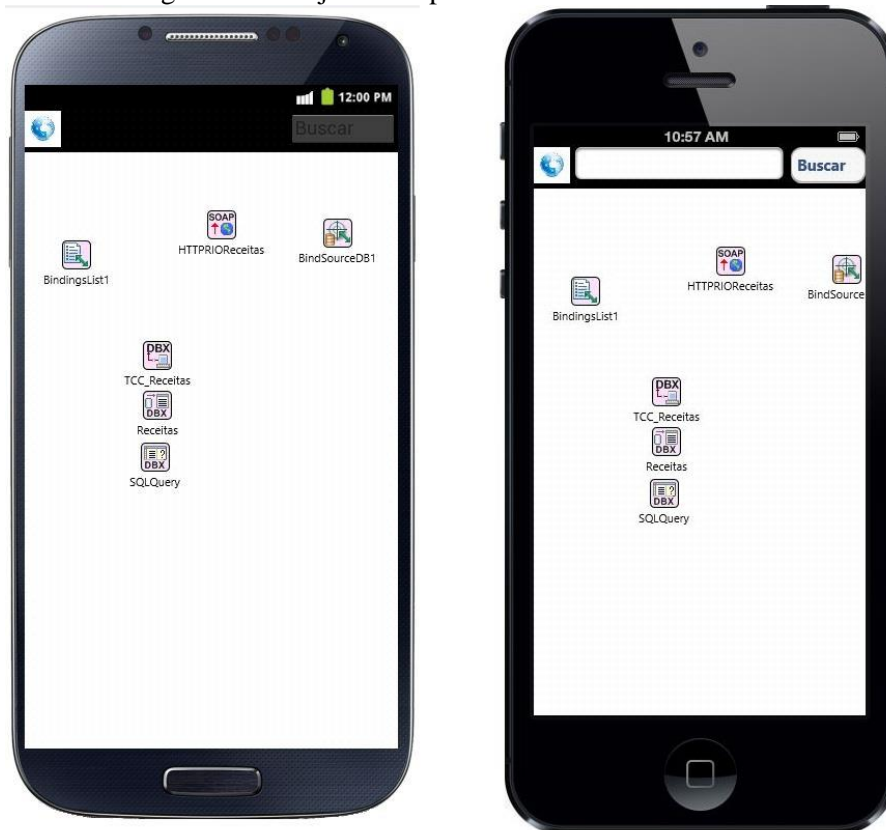
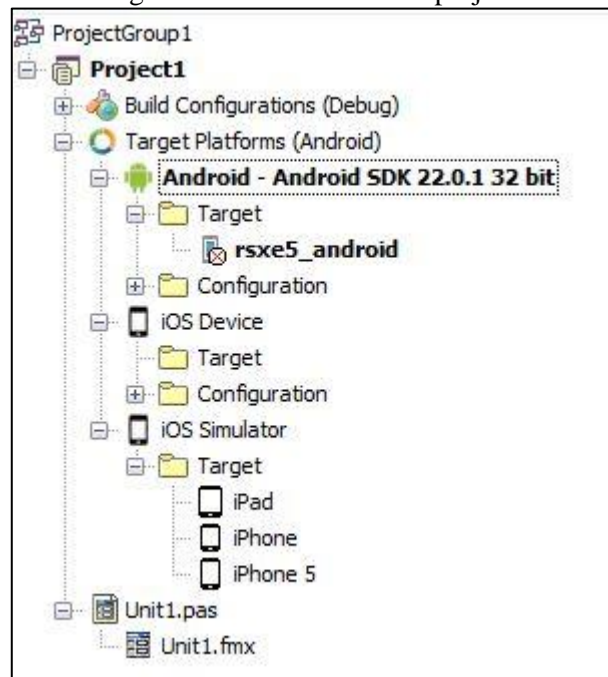


Figura 11 – Gerenciador de projetos



Para a plataforma iOS, é necessário efetuar uma configuração de acesso a uma máquina com o sistema operacional OS X, pois a ferramenta a utilizará para efetuar o *build*.

Essa configuração é iniciada pela instalação do pacote RADPAServerXE5.pkg em um computador com o sistema operacional OS X, localizado na pasta que foi instalado a ferramenta, geralmente em “C:\Program Files (x86)\Embarcadero\RAD Studio\12.0\PAServer”. O ícone de instalação e o ícone do *software* instalado podem ser visualizados na figura 12, assim como a execução na figura 13.

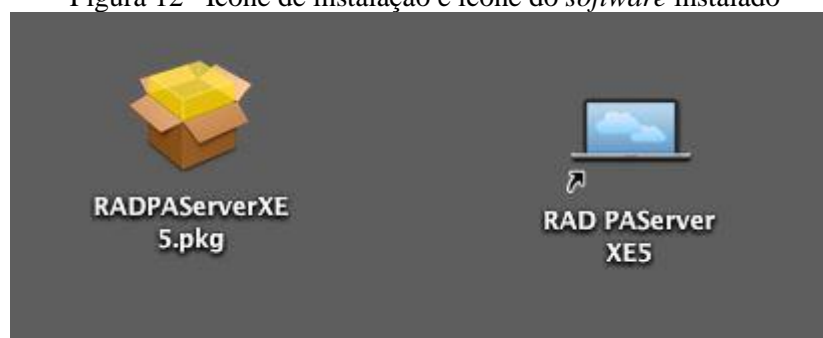
Figura 12– Ícone de instalação e ícone do *software* instalado

Figura 13 – Execução do RAD PAserver XE5



```
Last login: Tue Jun 10 20:40:27 on console
niksons-Mac:~ nikson$ /Applications/RAD\ PAserver\ XE5.app/Contents/MacOS/paserver ; exit;
Platform Assistant Server Version 4.2.0.05
Copyright (c) 2009-2013 Embarcadero Technologies, Inc.

Connection Profile password <press Enter for no password>:

Acquiring permission to support debugging...succeeded

Starting Platform Assistant Server on port 64211

Type ? for available commands
>i
FE80:0:0:0:20C:29FF:FE54:DEF4
192.168.118.128
>
```

A configuração necessária na ferramenta RAD Studio XE5, inicia nas propriedades do *iOS Device* ou *iOS Simulator* (figura 14). Na sequência, é necessário adicionar uma nova SDK (figura 15), para isso é configurado um perfil de conexão, como pode ser visto nas figura 16 e 17. São ilustrados nas figuras 18 e 19 o resultado das configurações. Feitas as configurações, a ferramenta está pronta para o desenvolvimento multiplataforma.

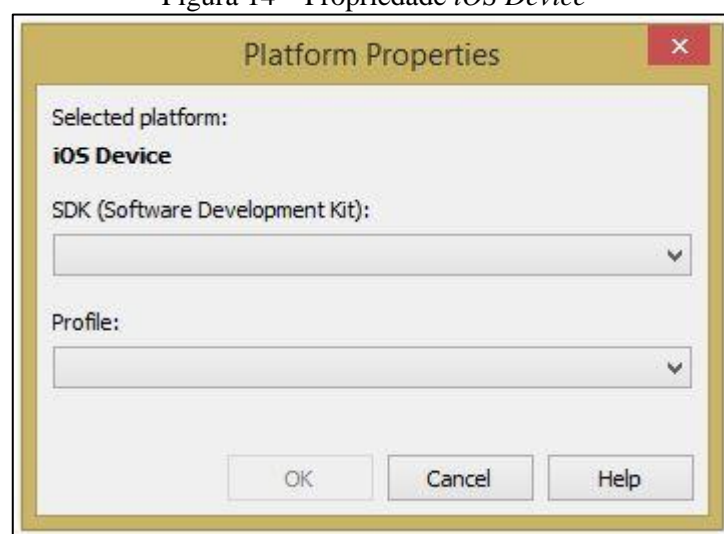
Figura 14 – Propriedade *iOS Device*

Figura 15 – Adicionar SDK no Delphi XE5

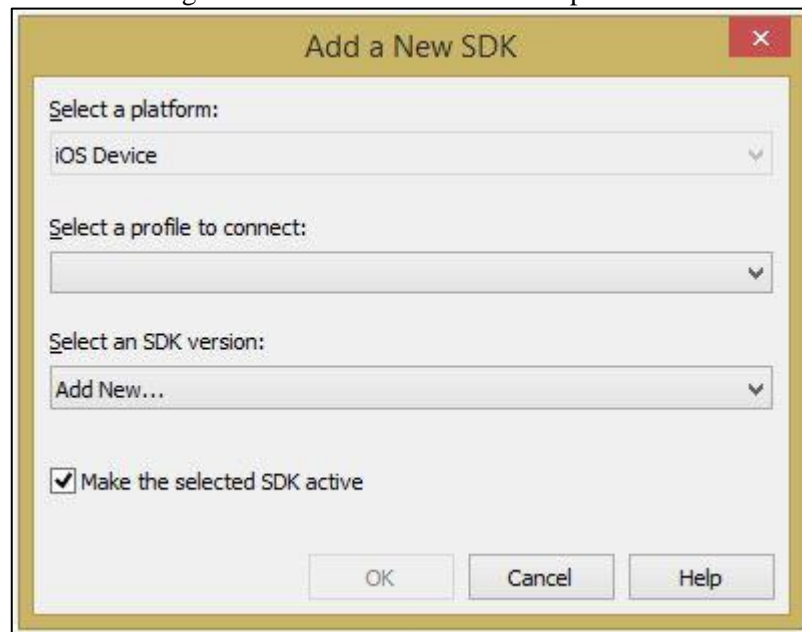


Figura 16 – Configuração de perfil de conexão

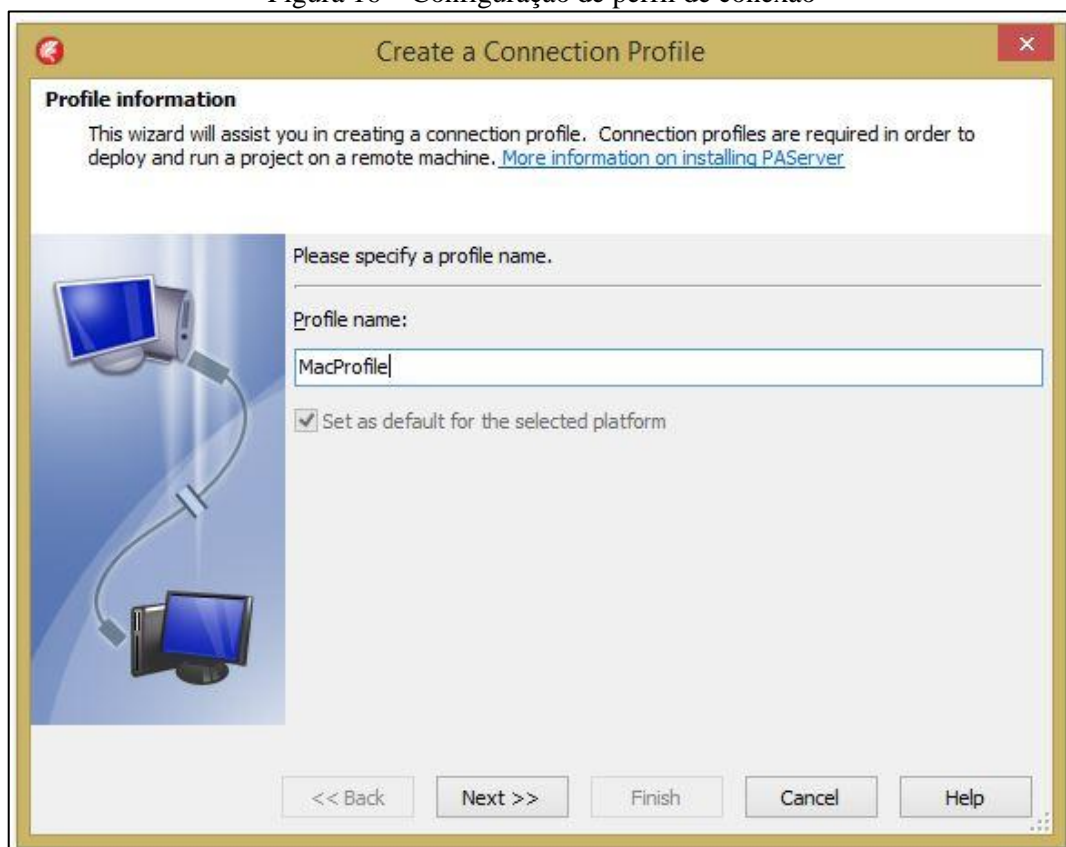


Figura 17 – Finalização da configuração do perfil de conexão

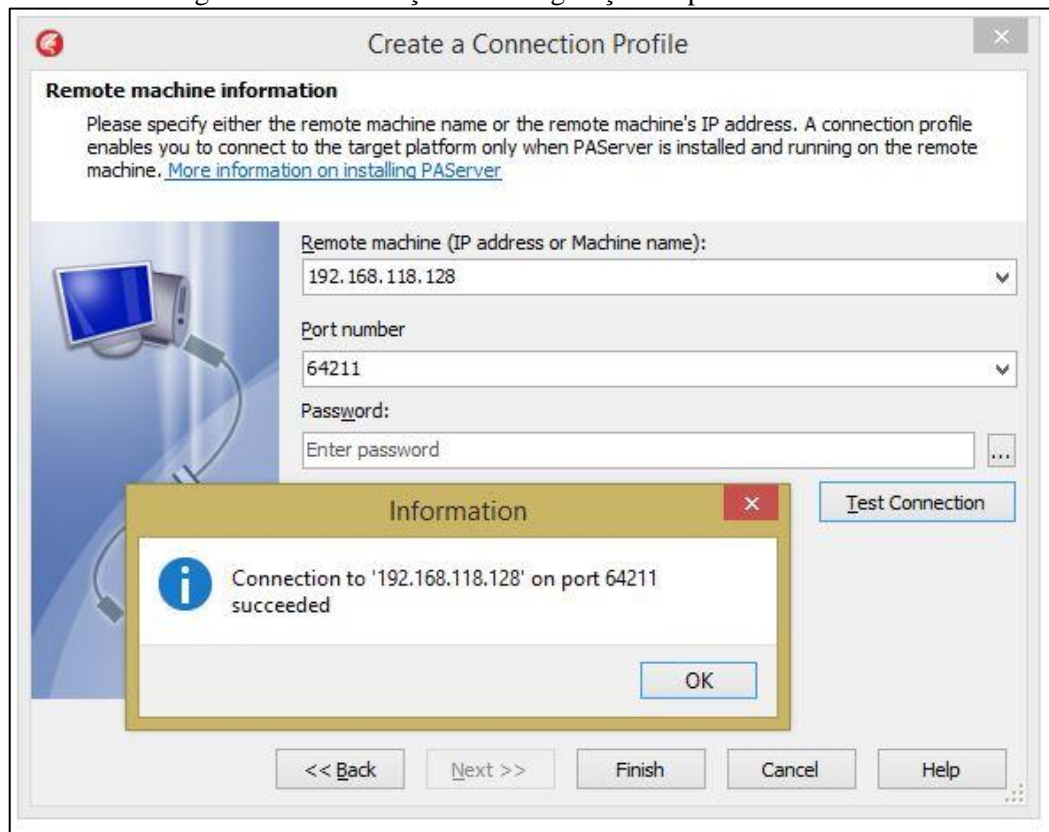


Figura 18 – Conclusão da configuração de nova SDK

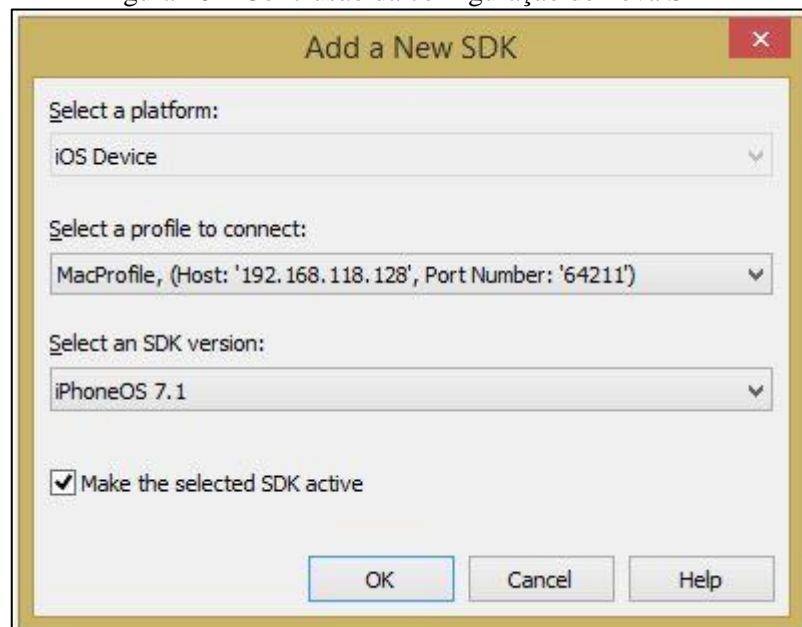
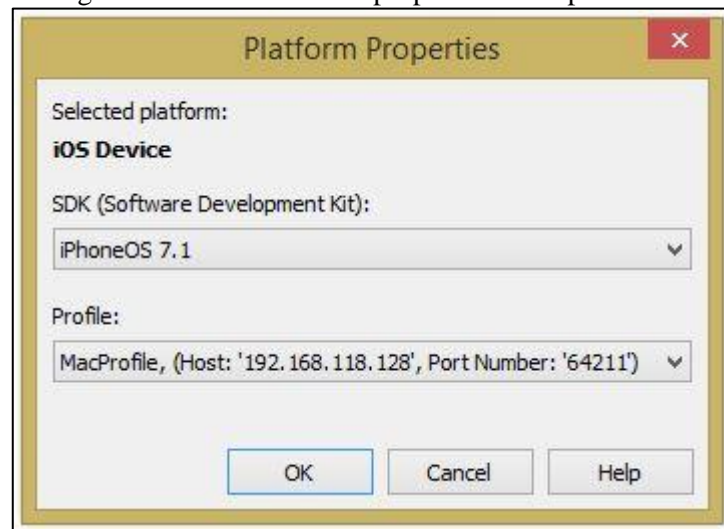
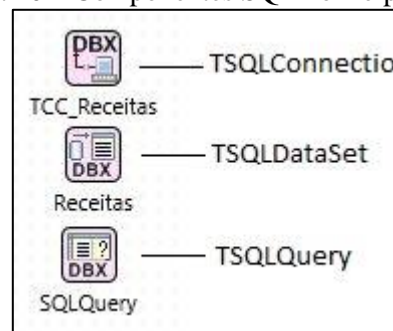


Figura 19 – Conclusão das propriedades da plataforma



A utilização do banco de dados do dispositivo móvel a partir do Delphi XE5 se dá pela configuração do banco de dados utilizado, em `Data Explorer`. Em seguida, é preciso adicionar no projeto ao menos três componentes: `TSQLConnection`, `TSQLQuery` e `TSQLDataSet` (Figura 20). Após todas as configurações feitas e os componentes devidamente adicionados ao projeto, basta executar as linhas de código ilustradas no quadro 5.

Figura 20 – Componentes SQL no Delphi XE5



Quadro 5 – Linhas de código SQL no Delphi XE5

```
Form1.SQLQuery.ParamByName('nome').AsString := txt;
Form1.SQLQuery.ExecSQL();
```

A conexão com um *Web Service* se dá pela configuração do componente `WSDL Importer` (Figura 21), que irá criar uma classe local (Figura 22), onde estarão mapeados os métodos disponibilizados pelo *Web Service*. Em seguida, é preciso adicionar o componente `THTTTPRIO` ao projeto (Figura 23). Efetuadas as configurações, a utilização dos métodos do *Web Service* acontece de forma transparente, como exemplificado no quadro 6.

Figura 21 – Componente WSDL Importer

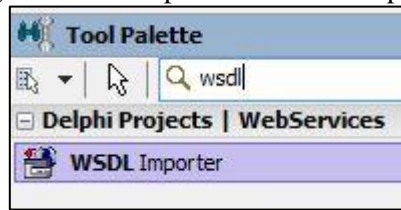
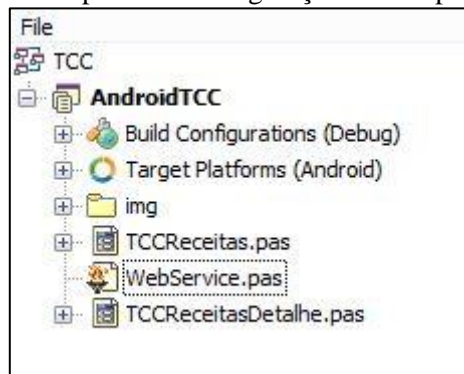


Figura 22 – Classe criada a partir da configuração do componente WSDL Importer



Quadro 6 – Utilização de um método do Web Service na ferramenta Delphi XE5

```
resultado := (HTTPRIOReceitas as WebServiceSoap).GetReceitasResumo(TxtEdit1.Text);
```

O Delphi XE5 permite utilizar o recurso acelerômetro do dispositivo móvel, basta adicionar o componente `TMotionSensor` (Figura 23). Porém para obter os dados é necessário adicionar um componente `TTimer` (Figura 23), assim os dados são obtidos dentro de um intervalo de tempo pré-determinado. No quadro 7 está exemplificado a utilização dos componentes no código fonte.

Figura 23 – TMotionSensor e TTimer



Quadro 7 – Exemplo de código utilizando TMotionSensor

```
for LProp in MotionSensor1.Sensor.AvailableProperties do
begin
  case LProp of
    TCustomMotionSensor.TProperty.AccelerationX:
      begin
        lbAccelerationX.Text := Format('Acceleration X: %6.2f', [MotionSensor1.Sensor.AccelerationX]);
      end;
    TCustomMotionSensor.TProperty.AccelerationY:
      begin
        lbAccelerationY.Text := Format('Acceleration Y: %6.2f', [MotionSensor1.Sensor.AccelerationY]);
      end;
    TCustomMotionSensor.TProperty.AccelerationZ:
      begin
        lbAccelerationZ.Text := Format('Acceleration Z: %6.2f', [MotionSensor1.Sensor.AccelerationZ]);
      end;
  end;
end;
```

O componente `TCameraComponent` do Delphi XE5 (Figura 24), permite a utilização da câmera do dispositivo móvel. Podendo utilizar os recursos referentes a câmera que estão disponíveis no dispositivo: câmera frontal, câmera traseira e configurações de *flash* (Quadro 8).

Figura 24 – Componente `TCameraComponent`



Quadro 8 – Exemplos de utilização do TCameraComponent

```

//Flash automático
if CameraComponent1.HasFlash then
    CameraComponent1.FlashMode := FMX.Media.TFlashMode.fmAutoFlash;
end;

//Flash desligado
if CameraComponent1.HasFlash then
    CameraComponent1.FlashMode := FMX.Media.TFlashMode.fmFlashOff;
end;

//Flash ligado
if CameraComponent1.HasFlash then
    CameraComponent1.FlashMode := FMX.Media.TFlashMode.fmFlashOn;
end;

//Ativa a câmera traseira
CameraComponent1.Active := False;
CameraComponent1.Kind := FMX.Media.TCameraKind.ckBackCamera;
CameraComponent1.Active := True;

//Ativa a câmera frontal
CameraComponent1.Active := False;
CameraComponent1.Kind := FMX.Media.TCameraKind.ckFrontCamera;
CameraComponent1.Active := True;

//Liga a câmera
CameraComponent1.Active := True;

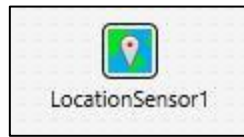
//Delisga a câmera
CameraComponent1.Active := False;

//Captura a imagem e atribui a um componente TImage
procedure TCameraComponentForm.CameraComponent1SampleBufferReady(
Sender: TObject; const ATime: Int64);
begin
    CameraComponent1.SampleBufferToBitmap(imgCameraView.Bitmap, True);
    imgCameraView.Width := imgCameraView.Bitmap.Width;
    imgCameraView.Height := imgCameraView.Bitmap.Height;
end;

```

O sistema de geolocalização do dispositivo pode ser utilizado através do componente `TLocationSensor` do Delphi XE5 (Figura 25). O exemplo contido no quadro 9, além de demonstrar a utilização do componente de geolocalização, também demonstra como obter o endereço referente às coordenadas obtidas pelo componente.

Figura 25 – TLocationSensor



Quadro 9 – Utilização do TLocationSensor

```

procedure TLocationForm.LocationSensor1LocationChanged(Sender: TObject;
const OldLocation, NewLocation: TLocationCoord2D);
const
  LGoogleMapsURL: String = 'https://maps.google.com/maps?q=%s,%s&output=embed';
begin
  lbLatitude.Text := 'Latitude: ' + NewLocation.Latitude.ToString;
  lbLongitude.Text := 'Longitude: ' + NewLocation.Longitude.ToString;
  WebBrowser1.Navigate(Format(LGoogleMapsURL, [NewLocation.Latitude.ToString, NewLocation.Longitude.ToString]));
end;

```

3.4.2 Xamarin

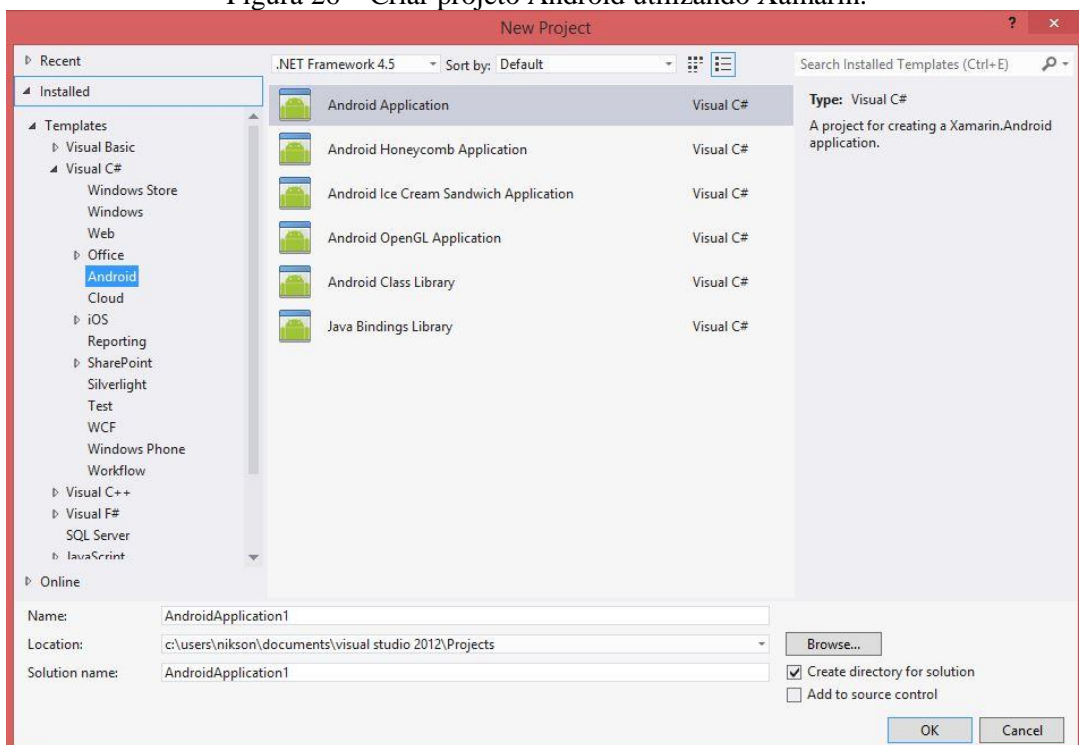
A ferramenta Xamarin também possui instalação. Basta obter o instalador no site da ferramenta (<http://www.xamarin.com/>) e instalar seguindo os passos do instalador.

A ferramenta pode ser utilizada juntamente com o Visual Studio da Microsoft, em plataformas Windows, ou pelo Xamarin Studio, em plataformas Windows e OS X.

Para a análise comparativa a ferramenta foi instalada na plataforma Windows, versão 8.1 e utilizada através do Microsoft Visual Studio 2012.

Para criar uma aplicação Android na ferramenta basta criar um novo projeto e selecionar a opção Android (Figura 26).

Figura 26 – Criar projeto Android utilizando Xamarin.



Para criar os projetos na plataforma iOS (Figura 27) é necessário configurar um equipamento com o sistema operacional OS X para que seja feita a construção das aplicações (Figura 28). A figura 29 ilustra o processo no sistema operacional OS X. As figura 30 e 31 ilustram o processo de configuração na ferramenta Visual Studio 2012.

Figura 27 – Criar projeto iOS utilizando Xamarin.

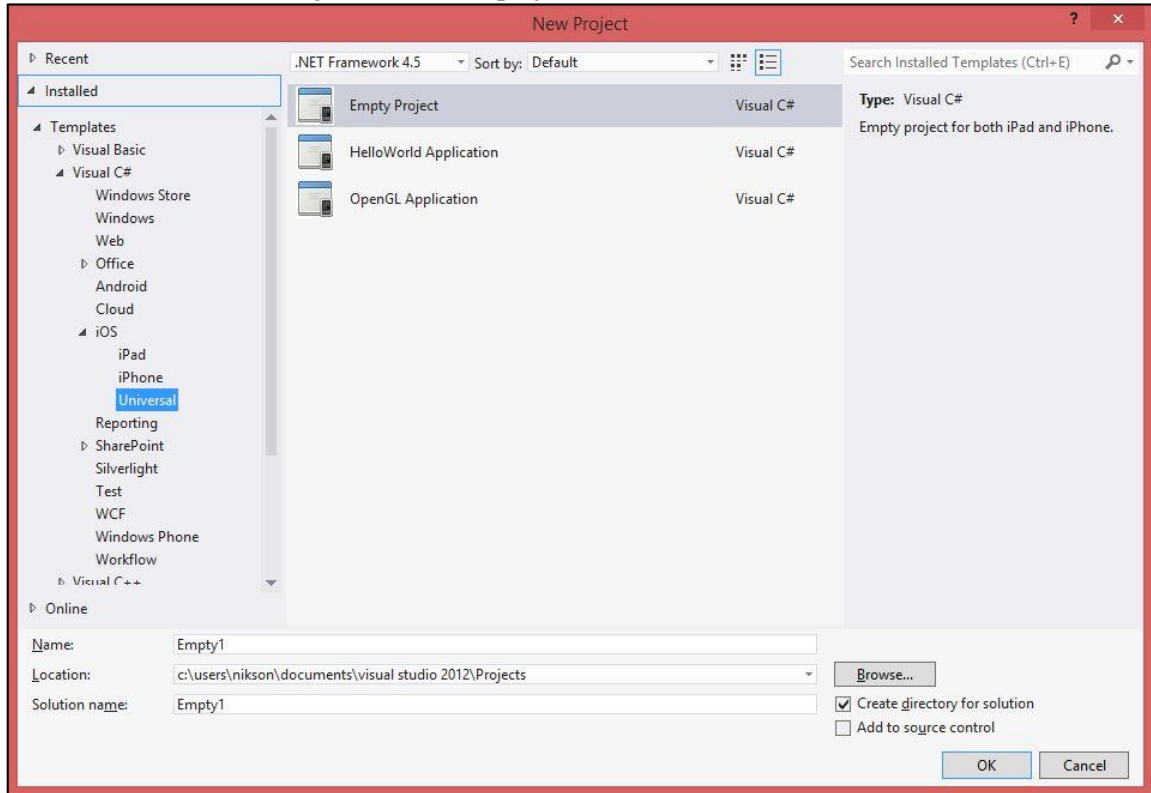


Figura 28– Configurar equipamento com o sistema OS X para construção das aplicações.

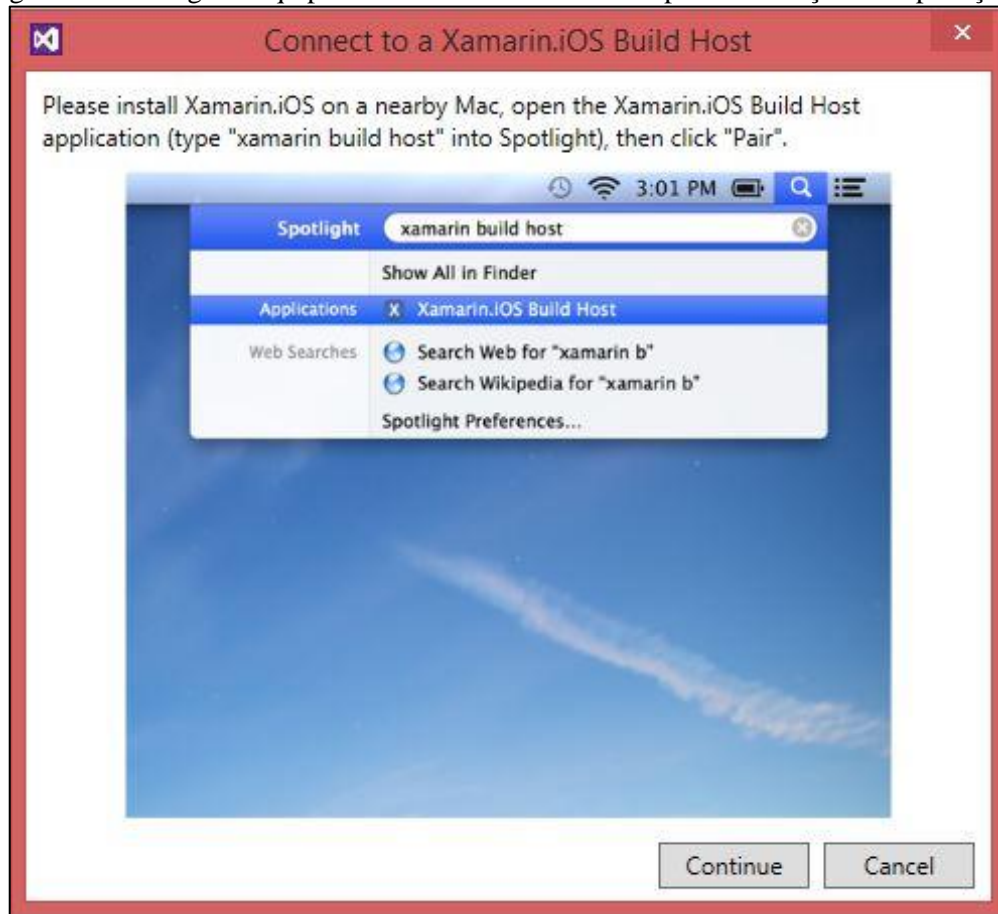


Figura 29 – Processo de configuração no sistema operacional OS X



Figura 30 – Início de configuração na ferramenta Visual Studio 2012

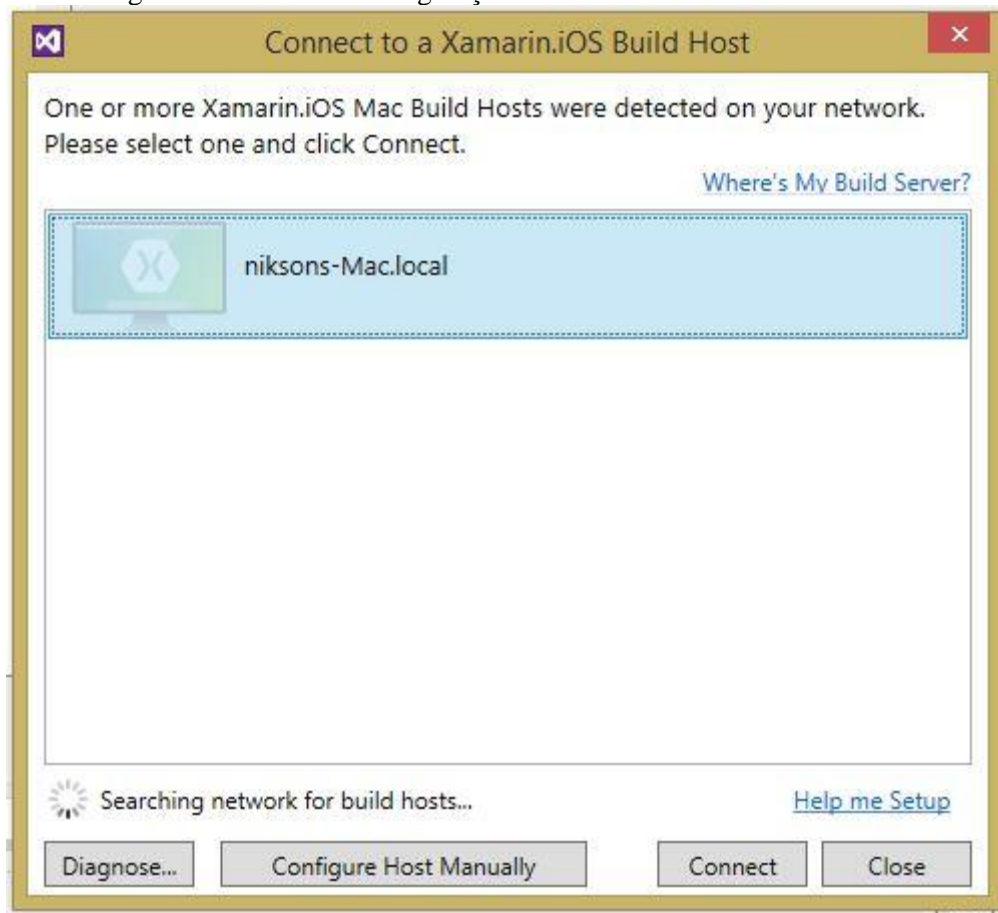


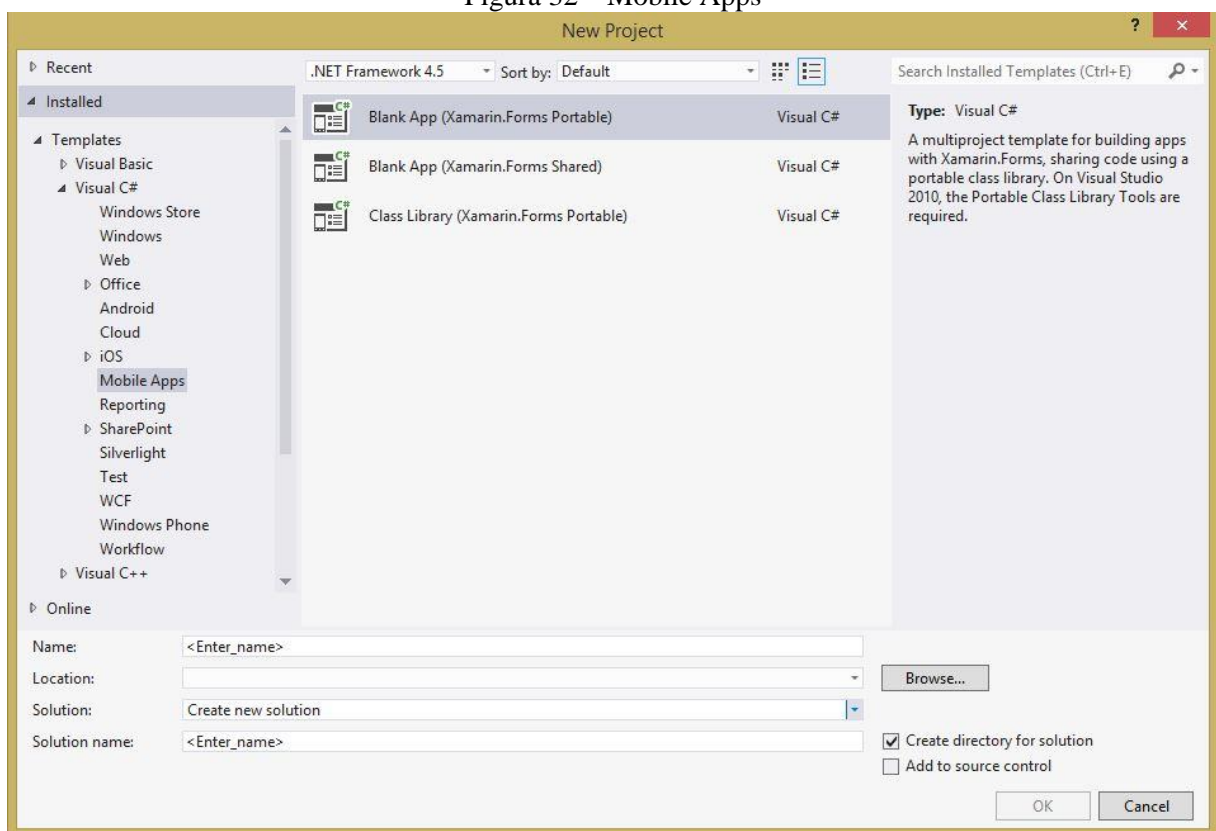
Figura 31 – Conclusão da configuração na ferramenta Visual Studio 2012



A ferramenta Xamarin também disponibiliza os projetos *Mobile Apps* (figura 32), onde a característica multiplataforma está em maior destaque, tendo uma única solução para as

plataformas Android e iOS. Porém, apesar de muito esforço e pesquisa, não foi possível efetuar a criação de um projeto *Blank APP (Xamarin.Forms Portable)* estável utilizando a ferramenta Visual Studio 2012 ou a ferramenta Xamarin Studio para plataforma Windows. Para a ferramenta Xamarin Studio, a opção citada não estava disponível. Por outro lado, utilizando a ferramenta Visual Studio 2012, foi possível criar o projeto. Entretanto, sua criação gerou vários erros de solução desconhecida. O mesmo ocorreu com o projeto *Class Library (Xamarin.Forms Portable)*. O projeto *Blank App (Xamarin.Forms Shared)*, é disponível apenas para a ferramenta Visual Studio a partir de 2013.

Figura 32 – Mobile Apps



O acelerômetro na ferramenta Xamarin é utilizado de maneiras diferentes para as plataformas Android e iOS. O exemplo do quadro 10 refere-se à plataforma iOS. Em seguida está exemplificado no quadro 11, o que é necessário para utilizar o sensor na plataforma Android.

Quadro 10 – Acelerômetro na plataforma iOS

```
CMMotionManager _motionManager = new CMMotionManager ();
_motionManager.StartAccelerometerUpdates (NSOperationQueue.CurrentQueue, (data, error) =>
{
    this.lblX.Text = data.Acceleration.X.ToString ("0.00000000");
    this.lblY.Text = data.Acceleration.Y.ToString ("0.00000000");
    this.lblZ.Text = data.Acceleration.Z.ToString ("0.00000000");
});
```

Quadro 11 – Acelerômetro na plataforma Android

```

protected override void OnCreate(Bundle bundle)
{
    base.OnCreate(bundle);

    SetContentView(Resource.Layout.Main);
    _sensorManager = (SensorManager)GetSystemService(Context.SensorService);
    _sensorTextView = FindViewById<TextView>(Resource.Id.accelerometer_text);
}

public void OnSensorChanged(SensorEvent e)
{
    lock (_syncLock)
    {
        var text = new StringBuilder("x = ")
            .Append(e.Values[0])
            .Append(", y=")
            .Append(e.Values[1])
            .Append(", z=")
            .Append(e.Values[2]);
        _sensorTextView.Text = text.ToString();
    }
}

protected override void OnResume()
{
    base.OnResume();
    _sensorManager.RegisterListener(this, _sensorManager.GetDefaultSensor(SensorType.Accelerometer), SensorDelay.Ui);
}

protected override void OnPause()
{
    base.OnPause();
    _sensorManager.UnregisterListener(this);
}

```

A ferramenta de desenvolvimento móvel multiplataforma Xamarin permite utilizar a câmera do dispositivo. O quadro 12 exemplifica a utilização da câmera na plataforma iOS, enquanto os quadros 13 e 14 exemplificam a utilização na plataforma Android.

Quadro 12 – Utilização da câmera na plataforma iOS

```

TweetStation.Camera.TakePicture (this, (obj) =>{
    var photo = obj.ValueForKey(new NSString("UIImagePickerControllerOriginalImage")) as UIImage;
    var meta = obj.ValueForKey(new NSString("UIImagePickerControllerMediaMetadata")) as NSDictionary;
    ALAssetsLibrary library = new ALAssetsLibrary();
    library.WriteImageToSavedPhotosAlbum (photo.CGImage, meta, (assetUrl, error) =>{
        Console.WriteLine ("assetUrl:"+assetUrl);
    });
});

```


Quadro 13 – Utilização da câmera na plataforma Android início

```

protected override void onCreate(Bundle bundle)
{
    base.onCreate(bundle);
    SetContentView(Resource.Layout.Main);

    if (IsThereAnAppToTakePictures())
    {
        CreateDirectoryForPictures();

        Button button = FindViewById<Button>(Resource.Id.myButton);
        _imageView = FindViewById<ImageView>(Resource.Id.imageView1);

        button.Click += TakeAPicture;
    }
}

private bool IsThereAnAppToTakePictures()
{
    Intent intent = new Intent(MediaStore.ActionImageCapture);
    IList<ResolveInfo> availableActivities = PackageManager.QueryIntentActivities(intent, PackageInfoFlags.MatchDefaultOnly);
    return availableActivities != null && availableActivities.Count > 0;
}

private void CreateDirectoryForPictures()
{
    _dir = new File(Android.OS.Environment.GetExternalStoragePublicDirectory(Android.OS.Environment.DirectoryPictures), "CameraAppDemo");
    if (!_dir.Exists())
    {
        _dir.Mkdirs();
    }
}

```

Quadro 14 – Utilização da Câmera na plataforma Android fim

```

private void TakeAPicture(object sender, EventArgs eventArgs)
{
    Intent intent = new Intent(MediaStore.ActionImageCapture);

    _file = new File(_dir, String.Format("myPhoto_{0}.jpg", Guid.NewGuid()));
    intent.PutExtra(MediaStore.ExtraOutput, Android.Net.Uri.FromFile(_file));

    StartActivityForResult(intent, 0);
}

protected override void OnActivityResult(int requestCode, Result resultCode, Intent data)
{
    base.OnActivityResult(requestCode, resultCode, data);

    Intent mediaScanIntent = new Intent(Intent.ActionMediaScannerScanFile);
    Android.Net.Uri contentUri = Android.Net.Uri.FromFile(_file);
    mediaScanIntent.SetData(contentUri);
    SendBroadcast(mediaScanIntent);

    int height = _imageView.Height;
    int width = Resources.DisplayMetrics.WidthPixels;
    using (Bitmap bitmap = _file.Path.LoadAndResizeBitmap(width, height))
    {
        _imageView.SetImageBitmap(bitmap);
    }
}

```

A ferramenta Xamarin disponibiliza a busca da geolocalização do dispositivo através do GPS. O quadro 15 exemplifica o uso do GPS na plataforma iOS. Os quadros 16, 17 e 18 exemplificam a utilização na plataforma Android.

Quadro 15 – GPS na plataforma iOS

```

var LocMgr = new CLLocationManager();

//Iniciar a localização
if (CLLocationManager.LocationServicesEnabled) {
    LocMgr.StartMonitoringSignificantLocationChanges ();
} else {
    Console.WriteLine ("GPS não está ativo");
}

//Dispara quando feita a localização
LocMgr.LocationsUpdated += (o, e) => Console.WriteLine ("Localização atualizada");

//Para a localização
LocMgr.StopMonitoringSignificantLocationChanges ();

```

Quadro 16 – GPS na plataforma Android

```

protected override void onCreate(Bundle bundle)
{
    base.onCreate(bundle);
    setContentView(Resource.Layout.Main);

    _addressText = FindViewById<TextView>(Resource.Id.address_text);
    _locationText = FindViewById<TextView>(Resource.Id.location_text);
    FindViewById<TextView>(Resource.Id.get_address_button).Click += AddressButton_OnClick;

    InitializeLocationManager();
}

protected override void onResume()
{
    base.onResume();
    _locationManager.RequestLocationUpdates(_locationProvider, 0, 0, this);
}

protected override void onPause()
{
    base.onPause();
    _locationManager.RemoveUpdates(this);
}

```


Quadro 17 – GPS na plataforma Android

```

void InitializeLocationManager()
{
    _locationManager = (LocationManager)GetSystemService(LocationService);
    Criteria criteriaForLocationService = new Criteria
    {
        Accuracy = Accuracy.Fine
    };
    IList<string> acceptableLocationProviders = _locationManager.GetProviders(criteriaForLocationService, true);

    if (acceptableLocationProviders.Any())
    {
        _locationProvider = acceptableLocationProviders.First();
    }
    else
    {
        _locationProvider = String.Empty;
    }
}

public void OnLocationChanged(Location location)
{
    _currentLocation = location;
    if (_currentLocation == null)
    {
        _locationText.Text = "Unable to determine your location.";
    }
    else
    {
        _locationText.Text = String.Format("{0},{1}", _currentLocation.Latitude, _currentLocation.Longitude);
    }
}
}

```

Quadro 18 – GPS na plataforma Android

```

public void OnProviderDisabled(string provider) {}

public void OnProviderEnabled(string provider) {}

public void OnStatusChanged(string provider, Availability status, Bundle extras) {}

async void AddressButton_OnClick(object sender, EventArgs eventArgs)
{
    if (_currentLocation == null)
    {
        _addressText.Text = "Can't determine the current address.";
        return;
    }

    Geocoder geocoder = new Geocoder(this);
    IList<Address> addressList = await geocoder.GetFromLocationAsync(_currentLocation.Latitude, _currentLocation.Longitude, 10);

    Address address = addressList.FirstOrDefault();
    if (address != null)
    {
        StringBuilder deviceAddress = new StringBuilder();
        for (int i = 0; i < address.MaxAddressLineIndex; i++)
        {
            deviceAddress.Append(address.GetAddressLine(i))
                .AppendLine(",");
        }
        _addressText.Text = deviceAddress.ToString();
    }
    else
    {
        _addressText.Text = "Unable to determine the address.";
    }
}
}

```

3.4.3 PhoneGap

Para utilizar o PhoneGap é necessário obter os arquivos de instalação no site da ferramenta (<http://phonegap.com/>).

A ferramenta funciona de forma semelhante nas plataformas Windows e OS X, sendo que o projeto final executará igualmente para as plataformas Android e iOS.

Caso a plataforma de desenvolvimento escolhida seja OS X, será possível executar o projeto nos dispositivos ou utilizar os simuladores das plataformas Android e iOS. Para a plataforma de desenvolvimento Windows, só é possível executar o projeto em dispositivos ou emuladores Android. No caso do iOS é necessário exportar o projeto para aquela plataforma.

A pasta com o nome “www” é o local onde será desenvolvido o projeto, utilizando HTML5, Javascript e CSS. Isso faz com que o projeto funcione com qualquer ferramenta de desenvolvimento, nesse caso Eclipse e Xcode.

A figura 33 ilustra o aplicativo 1 nas plataformas Android e iOS. Nota-se que mesmo as funcionalidades executando de forma semelhante, é preciso efetuar ajustes em relação à interface. No exemplo ilustrado na figura 33, a interface foi desenvolvida para o emulador de Android com tela de 7 polegadas.

Figura 33 – Simulação do aplicativo 1 PhoneGap



Para a plataforma Windows versão 8.1 é necessário obter o Apache ant, localizado em seu próprio site. Em seguida, adicionar algumas configurações nas variáveis de ambiente, iniciando pelo Path, adicionar os endereços referentes a Android SDK, Apache ant e o Java JDK (Quadro 19).

Quadro 19 – Configuração do Path

```
;C:\Android\sdk\platform-tools
;C:\Android\sdk\tools
;C:\Android\apache-ant-1.9.3\bin
;C:\Program Files\Java\jdk1.7.0_25\bin
```

O próximo passo é adicionar duas novas variáveis, a primeira denominada JAVA_HOME, e a segunda como ANT_HOME, com os valores: “C:\ProgramFiles\Java\jdk1.7.0_25\bin” e “C:\Android\apache-ant-1.9.3\bin”, respectivamente. Na plataforma OS X essas configuração não são necessárias.

Após o ambiente estar configurado, o próximo passo é criar o projeto que utilizará o Phonegap. Para isso é necessário utilizar o Prompt de Comandos em modo administrador para a plataforma Windows, e o Terminal para a plataforma OS X. Para ambas plataformas o comando a ser executado é o mesmo (Quadro 20). O comando ficará conforme as figuras 34 e 35.

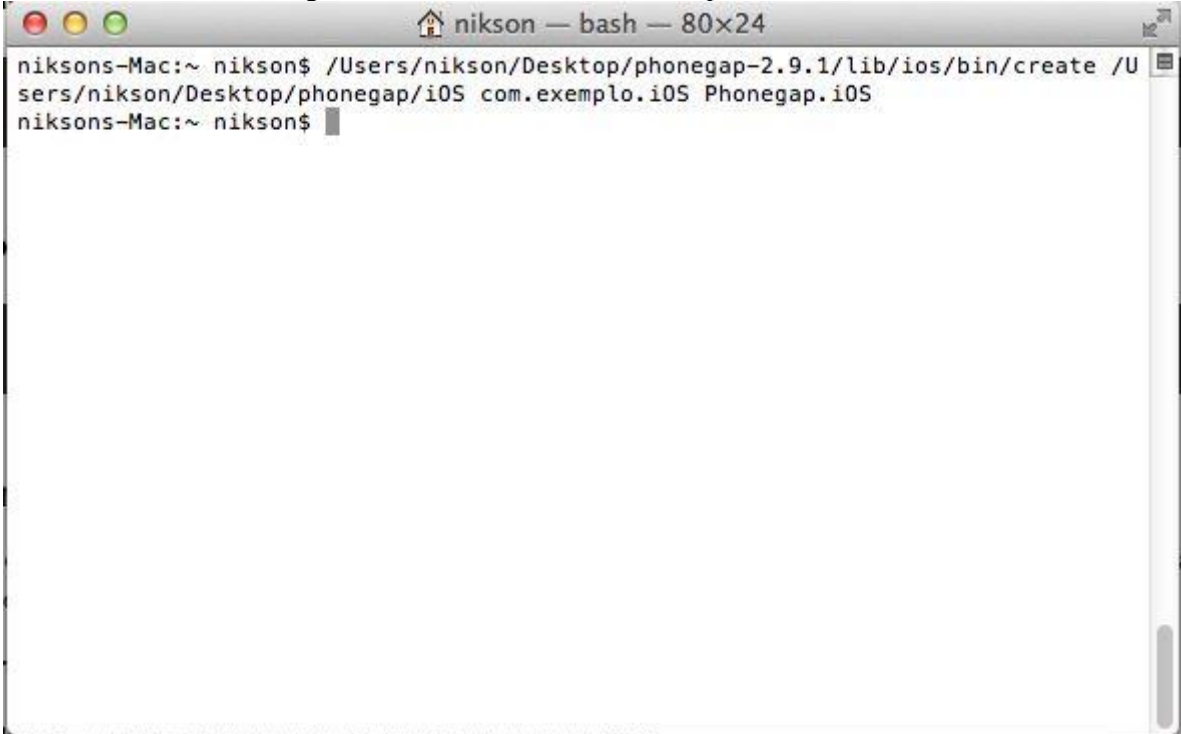
Quadro 20 – Comandos e descrições.

COMANDO	DESCRIÇÃO
C:\Android\phonegap\lib\android\bin\create.bat	Arquivos de instalação Phonegap
C:\PhoneGap\Android	Nome do software
com.exemplo.Android	Estrutura do projeto
Phonegap.Android	Nome da atividade principal

Figura 34 – Comando executado na plataforma Windows

```
Administrator: Prompt de Comando
C:\WINDOWS\system32>C:\Android\phonegap\lib\android\bin\create.bat C:\PhoneGap\A
ndroid com.exemplo.Android Phonegap.Android
Creating new android project...
Copying template files...
Copying js, jar & config.xml files...
Copying cordova command tools...
Updating AndroidManifest.xml and Main Activity...
C:\WINDOWS\system32>
```

Figura 35 – Comando executado na plataforma OS X

A screenshot of a macOS terminal window. The title bar shows the window name as 'nikson' and the size as '80x24'. The terminal content shows a user named 'nikson' at a 'niksons-Mac' prompt. The user enters the command: `/Users/nikson/Desktop/phonegap-2.9.1/lib/ios/bin/create /Users/nikson/Desktop/phonegap/iOS com.exemplo.iOS Phonegap.iOS`. The command is executed, and the prompt returns to `niksons-Mac:~ nikson$`.

```
niksons-Mac:~ nikson$ /Users/nikson/Desktop/phonegap-2.9.1/lib/ios/bin/create /Users/nikson/Desktop/phonegap/iOS com.exemplo.iOS Phonegap.iOS
niksons-Mac:~ nikson$
```

Na plataforma OS X, o processo de criação de projeto está concluído, basta abrir o arquivo “Nome da atividade principal.xcodeproj” e iniciar o desenvolvimento utilizando a ferramenta Xcode.

Utilizando a plataforma Windows, utiliza-se a ferramenta Eclipse, com o complemento Android SDK, que está disponível no site oficial do Android. Utilizando o Eclipse, basta criar um novo projeto e optar por criar a partir de um código existente (figura 36), selecionar a pasta criada pelo comando anterior e finalizar o processo (figura 37).

Figura 36 – Criar projeto PhoneGap a partir de um código existente

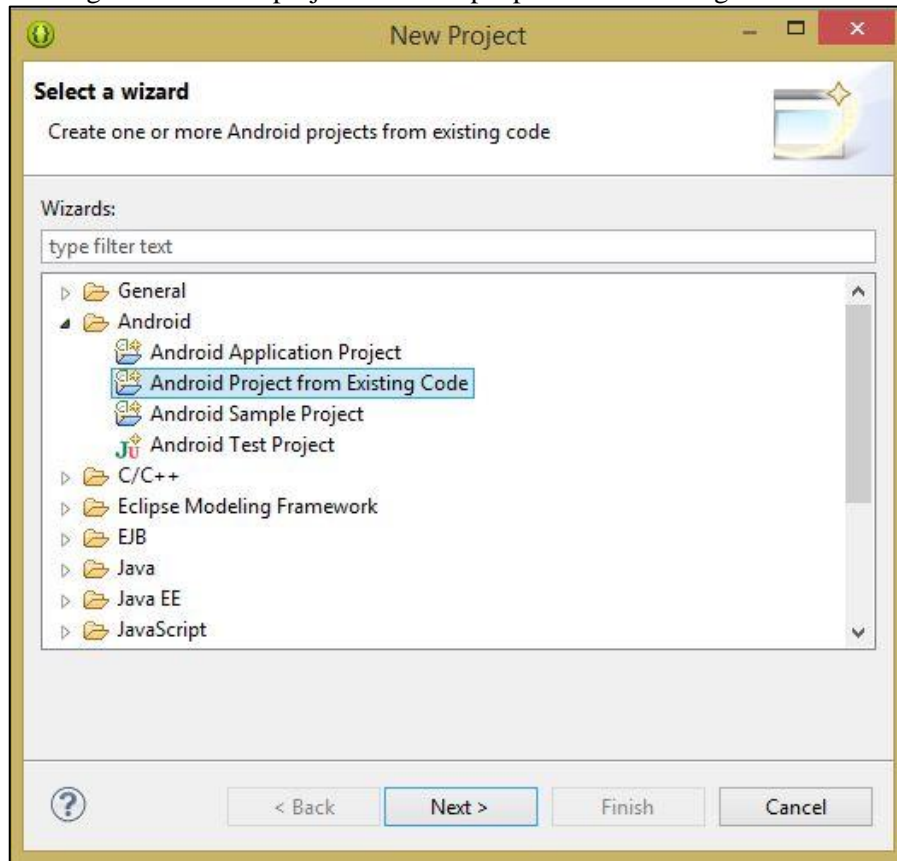
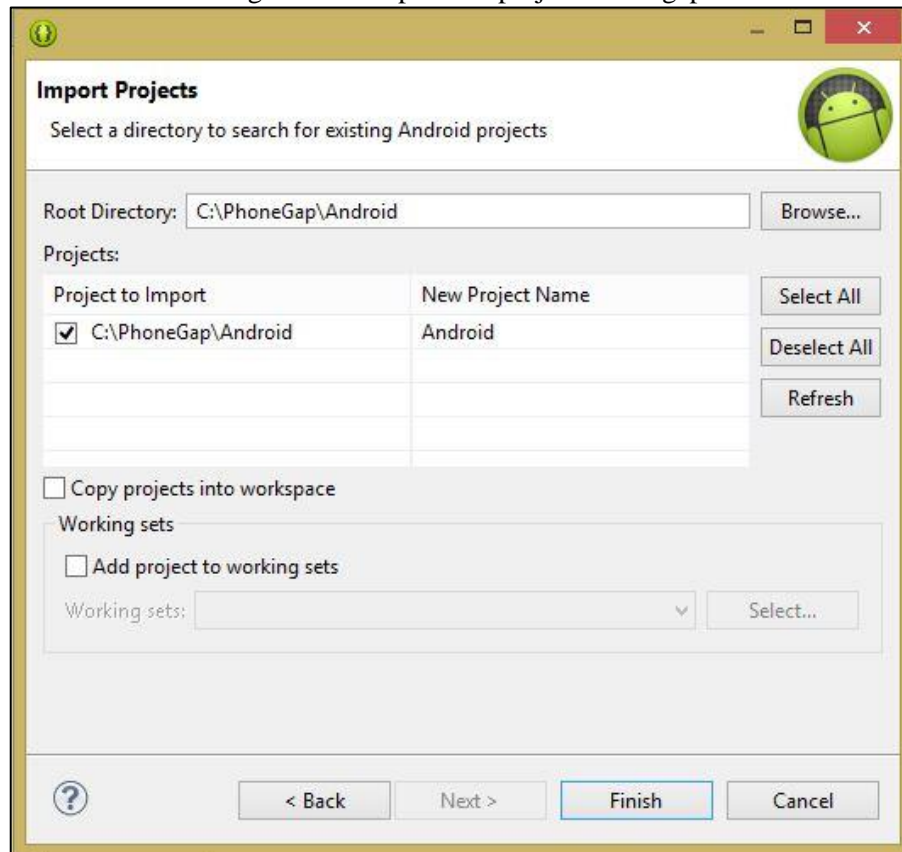


Figura 37 – Importar o projeto Phonegap



O uso do banco de dados do dispositivo móvel (Quadro 21), assim como a busca de informações de um *Web Service* (Quadro 22), ocorre de forma simples e igual para as plataformas Android e iOS, como esperado de uma ferramenta de desenvolvimento móvel multiplataforma.

Quadro 21 – Uso do banco de dados com a ferramenta PhoneGap

```
function criarTabelas(tx) {
    tx.executeSql('CREATE TABLE IF NOT EXISTS RECEITAS (id unique, nome, imagem)');
    tx.executeSql('CREATE TABLE IF NOT EXISTS INGREDIENTES (id unique, id_receita, descricao)');
    tx.executeSql('CREATE TABLE IF NOT EXISTS PREPARO (id unique, id_receita, descricao)');
}

function buscarReceitas(tx) {
    tx.executeSql('SELECT * FROM RECEITAS WHERE NOME LIKE \'%\' + $('#TxtBuscar').val() + \'%\',
        [], buscaSucesso, error);
}
```

Quadro 22 – Busca de dados de um Web Service com a ferramenta Phonegap

```
function BuscaReceitasWeb(ids) {
    var response = "";
    $.ajax({
        type: "GET",
        url: "http://192.168.0.100:80/WebService.asmx/GetReceitasResumoJQuery",
        data: { "param": $('#TxtBuscar').val() },
        contentType: "text/plain; charset=utf-8",
        dataType: "text",
        success: function (response) {
            var json_obj = $.parseJSON(response);
            var output = "";
            for (var i in json_obj) {
                var x = ids.indexOf(json_obj[i].Id);
                if (x == -1) {
                    output += "<li><a href=\\javascript:goDetail(" + json_obj[i].Id + ",0)>"
                        + json_obj[i].Nome + "</a></li>";
                }
            }
            $('#details').append(output);
        },
        error: function (response) {
            alert(response);
        }
    });
    return true;
}
```

Para obter os dados do recurso acelerômetro do dispositivo móvel, é utilizado o método `getCurrentAcceleration`, que é compatível para ambas plataformas, Android e iOS (Quadro 23). Porém a plataforma iOS tem algumas peculiaridades: para obter os dados de forma correta, é necessário utilizar o método `watchAccelerometer`, para capturar os dados em intervalos de tempo pré-determinados (Quadro 24).

Quadro 23 – PhoneGap – getCurrentAcceleration

```

document.addEventListener("deviceready", onDeviceReady, false);

function onDeviceReady() {
    navigator.accelerometer.getCurrentAcceleration(onSuccess, onError);
}

function onSuccess(acceleration) {
    alert('Acceleration X: ' + acceleration.x + '\n' +
        'Acceleration Y: ' + acceleration.y + '\n' +
        'Acceleration Z: ' + acceleration.z + '\n' +
        'Timestamp: '      + acceleration.timestamp + '\n');
}

function onError() {
    alert('onError!');
}

```

Quadro 24 – Phonegap – watchAccelerometer

```

var watchID = null;

document.addEventListener("deviceready", onDeviceReady, false);

function onDeviceReady() {
    startWatch();
}

function startWatch() {
    var options = { frequency: 3000 };
    watchID = navigator.accelerometer.watchAcceleration(onSuccess, onError, options);
}

function stopWatch() {
    if (watchID) {
        navigator.accelerometer.clearWatch(watchID);
        watchID = null;
    }
}

function onSuccess(acceleration) {
    var element = document.getElementById('accelerometer');
    element.innerHTML = 'Acceleration X: ' + acceleration.x + '<br />' +
        'Acceleration Y: ' + acceleration.y      + '<br />' +
        'Acceleration Z: ' + acceleration.z      + '<br />' +
        'Timestamp: '      + acceleration.timestamp + '<br />';
}

function onError() {
    alert('onError!');
}

```

A ferramenta Phonegap também torna possível utilizar a câmera dos dispositivos móveis, assim como capturar imagens que estão em suas galerias ou bibliotecas (Quadro 25).

No exemplo mostrado no quadro 25, tem-se algumas peculiaridades em relação à plataforma Android: a plataforma ignora o parâmetro `AllowEdit` e os parâmetros `pictureSource.PHOTOALBUM` e `pictureSource.SAVEDPHOTOALBUM` e mostra somente as fotos da galeria do dispositivo. A plataforma iOS também contém peculiaridades: é necessário definir a qualidade da imagem abaixo de 50, para evitar erros de memória em alguns

dispositivos. Quando utilizado o `destinationType.FILE_URI`, as fotos são salvas no diretório temporário do aplicativo, que pode ser apagado utilizando o `navigator.fileMgr`.

Quadro 25 – Utilização da câmera com a ferramenta PhoneGap

```

var pictureSource;
var destinationType;

document.addEventListener("deviceready",onDeviceReady,false);

function onDeviceReady() {
    pictureSource=navigator.camera.PictureSourceType;
    destinationType=navigator.camera.DestinationType;
}

function onPhotoDataSuccess(imageData) {
    var smallImage = document.getElementById('smallImage');
    smallImage.style.display = 'block';

    smallImage.src = "data:image/jpeg;base64," + imageData;
}

function onPhotoURISuccess(imageURI) {
    var largeImage = document.getElementById('largeImage');

    largeImage.style.display = 'block';
    largeImage.src = imageURI;
}

function capturePhoto() {
    navigator.camera.getPicture(onPhotoDataSuccess, onFail, { quality: 50,
        destinationType: destinationType.DATA_URL });
}

function capturePhotoEdit() {
    navigator.camera.getPicture(onPhotoDataSuccess, onFail, { quality: 20, allowEdit: true,
        destinationType: destinationType.DATA_URL });
}

function getPhoto(source) {
    navigator.camera.getPicture(onPhotoURISuccess, onFail, { quality: 50,
        destinationType: destinationType.FILE_URI,
        sourceType: source });
}

function onFail(message) {
    alert('Failed because: ' + message);
}

```

A utilização do GPS não contém peculiaridades para a plataforma iOS, apenas para a plataforma Android, onde o método `altitudeAccuracy` não é suportado, retornando sempre o valor nulo.

O exemplo da utilização do GPS através da ferramenta PhoneGap pode ser visualizado no quadro 26.

Quadro 26 – Utilização do GPS com a ferramenta PhoneGap

```

document.addEventListener("deviceready", onDeviceReady, false);

function onDeviceReady() {
    navigator.geolocation.getCurrentPosition(onSuccess, onError);
}

function onSuccess(position) {
    var element = document.getElementById('geolocation');
    element.innerHTML = 'Latitude: ' + position.coords.latitude + '<br />' +
        'Longitude: ' + position.coords.longitude + '<br />' +
        'Altitude: ' + position.coords.altitude + '<br />' +
        'Accuracy: ' + position.coords.accuracy + '<br />' +
        'Altitude Accuracy: ' + position.coords.altitudeAccuracy + '<br />' +
        'Heading: ' + position.coords.heading + '<br />' +
        'Speed: ' + position.coords.speed + '<br />' +
        'Timestamp: ' + position.timestamp + '<br />';
}

function onError(error) {
    alert('code: ' + error.code + '\n' +
        'message: ' + error.message + '\n');
}

```

3.5 RESULTADOS E DISCUSSÃO

Neste capítulo são apresentados os resultados obtidos ao término do desenvolvimento dos aplicativos e da análise das ferramentas de desenvolvimento móvel multiplataforma.

3.5.1 Resultado do questionário de avaliação

Para realizar a análise comparativa das ferramentas de desenvolvimento móvel multiplataforma, foi respondido o questionário apresentado no quadro 2, com o intuito de evidenciar as diferenças entre as características não classificadas como subjetivas das ferramentas. O questionário foi respondido pelo autor, baseado nas pesquisas realizadas no desenvolvimento da análise comparativa, ressaltando que no início das pesquisas o autor não era conhecedor das ferramentas. Os resultados estão apresentados no quadro 27.

Para responder as questões referentes as subcategorias “Comportamento em relação ao tempo” e “Utilização de recursos”, foram realizados alguns testes com os aplicativos desenvolvidos. Esses testes estão ilustrados nas tabelas 1, 2, 3, 4 e 5.

Tabela 1 – Tempo em min de instalação no dispositivo Android

	Delphi XE5	Xamarin	PhoneGap
Aplicativo 1	01:40:508	00:13:990	00:05:844
Aplicativo 2	01:56:259	00:14:281	00:06:327

Tabela 2 – Tempo em min de instalação no emulador Android

	Delphi XE5	Xamarin	PhoneGap
Aplicativo 1	02:45:740	02:30:771	00:13:716
Aplicativo 2	03:20:165	03:11:005	00:16:846

Tabela 3 – Tempo em min de instalação no emulador iOS

	Delphi XE5	Xamarin	PhoneGap
Aplicativo 1	01:41:249	00:28:108	00:04:293
Aplicativo 2	02:02:752	00:35:468	00:05:346

Tabela 4 – Tempo em min de compilação dos projetos

	Delphi XE5	Xamarin	PhoneGap
Aplicativo 1	01:02:473	00:05:532	00:02:574
Aplicativo 2	01:12:165	00:03:372	00:03:413

Tabela 5 – Recurso utilizando no dispositivo Android

	Delphi XE5	Xamarin	PhoneGap
Aplicativo 1	RAM: 38,52mb, CPU 1,08%	RAM: 14,97mb, CPU 0,01%	RAM: 27,52mb, CPU 1,00%
Aplicativo 2	RAM: 45,30mb, CPU 1,51%	RAM: 18,01mb, CPU 0,01%	RAM: 39,41mb, CPU 1,33%

O cenário utilizado para os testes é o dispositivo Samsung Note 2, que possui processador *Quad-core* 1.6, 2 GB de memória RAM e 16 GB de armazenamento interno. O computador usado possui o processador Intel i5 2500k com quatro *cores* de 3.7Ghz, 8GB de memória RAM de velocidade 1600mhz e disco rígido com capacidade 2TB e 7200rpm.

Ao realizar os testes de instalação nos dispositivos, foi considerado o tempo desde o início da execução do projeto na ferramenta de desenvolvimento, até o projeto aparecer executando no dispositivo. O mesmo é considerado para os testes de instalação nos emuladores, que já estão iniciados no início dos testes.

A tabela 6 contém o uso dos recursos do computador usados pelas ferramentas com os projetos abertos.

A simulação do sistema operacional OS X 10.9.3 foi realizada com o auxílio da ferramenta VMware Workstation versão 10.0.2.

Tabela 6 – Recurso utilizando pelas ferramenta no computador

Delphi XE5	Xamarin - Visual Studio 2012	PhoneGap - Eclipse
RAM: 352,3mb, CPU 3,5%	RAM: 191,6mb, CPU 0,05%	RAM: 345,7mb, CPU 0,01%

Quadro 27 – Questionário de avaliação

		Delphi XE5			Xamarin			PhoneGap		
		0	5	10	0	5	10	0	5	10
Adequação funcional	Compila o mesmo código para as plataformas Android e iOS?			X	X					X
	Compila no emulador?			X			X			X
	Compila no dispositivo?			X			X			X
	Executa a construção rapidamente?		X				X			X
	Possui suporte e documentação?			X		X			X	
	A interface gráfica corresponde ao apresentado no dispositivo?			X			X		X	
	O código funciona corretamente?			X			X			X
	O software possui IntelliSense ¹ ?			X			X	X		
	O software possui uma interface de modelagem?			X			X	X		
			85			75			60	
Confiabilidade	Funciona sem problemas durante a codificação?		X				X			X
	Quando necessário, o software está operacional?			X			X			X
	Se algo inesperado ocorrer durante a instalação no dispositivo, o software continua funcionando?		X			X				X
	Se algo inesperado ocorrer durante a instalação no emulador, o software continua funcionando?		X			X				X
	Quando ocorre um defeito, o software consegue recuperar os dados que estavam sendo trabalhados?			X			X			X
		35			40			50		
Usabilidade	O software é apropriado para o desenvolvimento dos aplicativos especificados?			X		X				X
	Os conceitos-chave do software são de fácil aprendizagem?			X			X			X
	É fácil instalar o aplicativo no emulador?			X			X			X
	É fácil instalar o aplicativo no dispositivo?			X			X			X
	É intuitiva a criação de projetos?			X			X	X		
	O software auxilia o usuário a não cometer erros?			X			X	X		
	A interface do software proporciona prazer e interação satisfatórios?			X			X		X	
	O software foi projetado para atender usuários com necessidades especiais?	X				X			X	
		63			58			42.5		
Eficiência de desempenho	Tempo necessário para compilar o projeto	X				X				X
	Tempo necessário para instalar no	X				X				X

¹Recurso do Visual Studio projetado para ajudar o desenvolvedor. Fornece elementos de código lógico.

	dispositivo								
	Tempo necessário pra instalar no emulador	X			X				X
	Recursos utilizados no computador	X				X		X	
	Recursos utilizados no dispositivo	X				X		X	
	O software atendeu todos os requisitos especificados nos aplicativos?			X		X			X
				10		40			50
Compatibilidade	O software é capaz de coexistir com outros softwares do computador?			X			X		X
	O software é capaz de coexistir com outros softwares do dispositivo?			X			X		X
	O software é capaz de interagir com o Mac OS? (Necessário para compilar para iOS)			X			X	X	
				30		30			20
Portabilidade	O software funciona na plataforma Windows?			X			X		X
	O software funciona na plataforma Mac OS?	X					X		X
	O software é facilmente instalado?			X			X	X	
	O software pode substituir outro?			X			X		X
				30		40			30
Efetividade	O software permite o desenvolvimento dos aplicativos especificados por completo?			X		X			X
				10		5			10
Eficiência	A compilação é rápida? Ou faz o usuário esperar?		X				X		X
				5		10			10
Satisfação	O usuário possui satisfação ao utilizar o software?		X			X			X
	O usuário sente prazer em usar o software?		X			X			X
	O uso de funções e componentes simples é de fácil acesso?		X				X	X	
				7.5		10			5
Cobertura de contexto	O software pode ser usado com efetividade, eficiência, sem riscos e com satisfação em todos os contextos especificados?		X				X		X
	O software pode ser usado com efetividade, eficiência, sem riscos e com satisfação em todos os contextos especificados?			X			X		X
				15		20			20
Total				290.5		353			297.5

Legenda: 0=Não; 5=Parcial; 10=Sim.

Parte considerável das respostas obtidas tiveram respostas iguais, isso ocorre devido ao fato que Delphi XE5, Xamarin e PhoneGap são ferramentas de desenvolvimento móvel

multiplataformas. As questões que obtiveram respostas iguais foram destacadas para facilitar a análise.

Baseado no questionário, foi possível concluir que algumas características se destacam mais em determinada ferramenta (quadro 28).

Quadro 28 – Comparativo de características

	Delphi XE5	Xamarin	PhoneGap
Adequação funcional	X		
Confiabilidade			X
Usabilidade	X		
Eficiência de desempenho			X
Compatibilidade	X	X	
Portabilidade		X	
Efetividade	X		X
Eficiência		X	X
Satisfação		X	
Cobertura de contexto	X	X	X

Ao responder o questionário, notou-se grande semelhança entre as ferramentas, porém no quadro 28 pode-se observar que o Delphi XE5 é uma ferramenta que tem vantagens relacionadas a adequação funcional, que demonstra preparo para desenvolver aplicativos multiplataformas com necessidades diversas. Em contrapartida, a ferramenta PhoneGap se mostrou de grande eficiência e efetividade. Não menos importante, a ferramenta Xamarin tem alto índice de compatibilidade e portabilidade. Uma de suas vantagens é ter uma ferramenta de desenvolvimento compatível com a plataforma Windows e OS X e possuir compatibilidade com o Visual Studio da Microsoft.

4 CONCLUSÕES

Quando se inicia um projeto móvel e multiplataforma, uma decisão precisa ser tomada: desenvolver aplicativos distintos para cada plataforma, ou utilizar uma ferramenta de desenvolvimento móvel multiplataforma. Se a última for a opção, surge outra questão: com qual ferramenta será possível atingir os objetivos em relação ao projeto? A decisão de qual ferramenta utilizar irá influenciar muitos fatores, dentre eles, a produtividade no desenvolvimento e a eficiência do projeto desenvolvido.

Com o intuito de auxiliar a escolha de uma ferramenta de desenvolvimento móvel multiplataforma, foi desenvolvida uma análise comparativa das ferramentas, iniciando pela definição dos critérios de avaliação baseados na norma NBR ISO/IEC 25000 SQuERE, que estabelece os itens a serem considerados para a qualidade de software.

Estabelecidos os critérios da avaliação, estes foram correlacionados com as principais funcionalidades de 3 ferramentas de desenvolvimento, visando qualificá-las e compará-las. Então, notou-se a necessidade da divisão das subcaracterísticas, visando identificar as subjetivas, onde as menos subjetivas foram avaliadas através de um questionário produzido pelo autor e as subjetivas pela avaliação pessoal do autor.

Ao término do desenvolvimento dos aplicativos desenvolvidos com o intuito de avaliar as ferramentas, foi possível concluir que:

- a) Delphi XE5 é adequado para o desenvolvimento multiplataforma; tem grande usabilidade; é efetivo e pode cobrir as diversas necessidades dos aplicativos. Em contrapartida, a ferramenta é compatível apenas para ser instalada na plataforma Windows e seu desempenho é menor que as demais ferramentas analisadas;
- b) Xamarin proporciona satisfação ao usuário em relação ao uso e a facilidade de acesso a funções e componentes, principalmente por ter a possibilidade de ser utilizado através da ferramenta Visual Studio da Microsoft. Mostra-se completo em relação ao desenvolvimento móvel. Também possui grande compatibilidade, pois disponibiliza uma ferramenta de desenvolvimento compatível com as plataformas Windows e OS X. Para a versão Visual Studio 2012, não foi possível desenvolver um único código para Android e iOS;
- c) PhoneGap não proporciona facilidade em sua instalação e criação de projetos, assim como o desenvolvimento e acesso a funções não é facilitado. Entretanto, se mostrou confiável e eficiente em relação ao que se propõem a fazer, o que resulta em ganho de confiança do usuário.

Tendo os resultados da análise comparativa, foi concluído que as ferramentas apresentam semelhanças em relação à qualidade de software. Portanto resta ao usuário optar pela ferramenta que está mais preparada para atingir seus objetivos. Uma ferramenta efetiva, com grande usabilidade (Delphi XE 5), uma ferramenta que pode ser utilizada através do Visual Studio e possui grande compatibilidade e portabilidade (Xamarin) ou uma ferramenta que tem com principal destaque a eficiência e efetividade (PhoneGap).

4.1 EXTENSÕES

Sugere-se as seguintes extensões deste trabalho:

- a) realizar a análise comparativa sobre o recurso Bluetooth, utilizando as ferramentas Delphi XE5, Xamarin e PhoneGap;
- b) realizar a análise comparativa sobre os componentes de interface com usuário e sua exibição e funcionalidade nas diferentes plataformas, utilizando as ferramentas Delphi XE5, Xamarin e PhoneGap.

REFERÊNCIAS

- ADOBE SYSTEMS INC. PhoneGap, 2014. Disponível em: <<http://phonegap.com/>>. Acesso: 16 abr. 2014
- CYGNIS MEDIA. **Benefits (and disadvantages) of developing cross-platform mobile apps**. 2013. Disponível em: <<http://www.cygnismedia.com/blog/developing-cross-platform-mobile-apps>>. Acesso em: 29 out. 2013.
- EMBARCADERO TECHNOLOGIES INC. **Delphi XE5**, 2014. Disponível em: <<http://www.embarcadero.com/br/products/delphi/>>. Acessado em: 17 abr. 2014
- FERREIRA, Flavio D. V. M. **Projeto de melhoria da qualidade do desenvolvimento de um software multiplataforma**. 2012. Disponível em: <<http://www.slideshare.net/flaviofabricio/projeto-de-melhoria-da-qualidade-do-desenvolvimento-de-um-software-multiplataforma>>. Acesso em: 29 out. 2013.
- GUINTEHER, Marcel. **Desenvolvimento mobile multiplataforma com Phonegap**. 2011. Disponível em: <<http://www.mobiltec.com.br/blog/index.php/desenvolvimento-mobile-multiplataforma-com-phonegap>>. Acesso em: 23 abr. 2014.
- JÚNIOR, Gesmar. **Desenvolvimento multiplataforma mobile com HTML5 – Parte 1**. 2012. Disponível em: <<http://gesmarjunior.wordpress.com/2012/04/01/desenvolvimento-multiplataforma-mobile-com-html5-parte-1>>. Acesso em: 25 abr. 2014.
- NUNES, Flávio. **Desenvolvendo aplicativos móveis multiplataforma**. São Paulo, 2013. Disponível em: <<http://imasters.com.br/desenvolvimento/desenvolvendo-aplicativos-moveis-multiplataforma>>. Acesso em: 24 abr. 2014.
- REZENDE, Vandir F. **Análise comparativa entre Groovy e Java, aplicado no desenvolvimento web**. 2011. 76 f. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) – Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.
- RIBEIRO, Iury. **Introdução ao desenvolvimento móvel**. Recife, 2012. Disponível em: <<http://www.slideshare.net/iurylira/desenvolvimento-mvel-12688321>>. Acesso em: 29 out. 2013.
- SILVA, Jomar. **Desenvolvimento de apps multiplataforma para dispositivos móveis com HTML5**. 2013. Disponível em: <<http://www.slideshare.net/IntelSoftwareBR/desenvolvimento-de-apps-multiplataforma-para-dispositivos-mveis-com-html5>>. Acesso em: 01 maio. 2014.
- SOUZA, Carlos E. de. **Desenvolvimento de aplicativo móvel multiplataforma integrado ao sistema de alerta de cheias da bacia do Itajaí**. 2012. 97 f. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) – Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.
- WAZLAWICK, Raul S.. **Engenharia de software: conceitos e práticas**. Rio de Janeiro: Elsevier, 2011.
- XAMARIN INC. **Xamarin**, 2014. Disponível em: <<http://xamarin.com/>>. Acesso: 15 abr. 2014