CENTRO UNIVERSITÁRIO DE ANÁPOLIS – UNIEVANGÉLICA
BACHARELADO EM ENGENHARIA DE COMPUTAÇÃO

MATHEUS DE OLIVEIRA MARQUES FIGUEREDO

DESENVOLVIMENTO DE APLICAÇÕES MÓVEIS: AVALIAÇÃO DO ESFORÇO NECESSÁRIO PARA AS PLATAFORMAS NATIVA E MULTIPLATAFORMA

ANÁPOLIS - GO
2017/02

MATHEUS DE OLIVEIRA MARQUES FIGUEREDO

DESENVOLVIMENTO DE APLICAÇÕES MÓVEIS: AVALIAÇÃO DO ESFORÇO NECESSÁRIO PARA AS PLATAFORMAS NATIVA E MULTIPLATAFORMA

Monografia apresentado ao Curso de Bacharelado em Engenharia de Computação do Centro Universitário de Anápolis – UniEVANGÉLICA, como requisito parcial à aprovação na disciplina Trabalho de Conclusão de Curso II, sob orientação da Profª Ma. Renata Dutra Braga.

ANÁPOLIS - GO
2017/02

MATHEUS DE OLIVEIRA MARQUES FIGUEREDO

DESENVOLVIMENTO DE APLICAÇÕES MÓVEIS: AVALIAÇÃO DO ESFORÇO NECESSÁRIO PARA AS PLATAFORMAS NATIVA E MULTIPLATAFORMA

Monografia apresentado ao Curso de Bacharelado em Engenharia de Computação do Centro Universitário de Anápolis - UNIEVANGÉLICA, como requisito parcial à aprovação na disciplina Trabalho de Conclusão de Curso II, sob orientação da Profª Ma. Renata Dutra Braga e avaliado pela seguinte banca examinadora:

_____
Profª Natasha Sophie Pereia

_____
Professor Millys Fabrielle Araújo Carvalhaes

_____
Profª Ma. Renata Dutra Braga - UniEvangelica

ANÁPOLIS - GO
2017/02

AGRADECIMENTOS

Primeiramente agradeço a Deus que permitiu que este momento fosse vivido por mim, trazendo alegria aos meus pais e a todos que contribuíram para a realização deste trabalho.

A esta instituição pelo excelente ambiente oferecido aos seus alunos e os profissionais qualificados que disponibiliza para nos ensinar.

À Profª Ma. Renata Dutra Braga por toda sua atenção, dedicação e esforço para que eu pudesse ter confiança e segurança na realização deste trabalho.

Agradeço aos meus pais, pelo amor, carinho, paciência e seus ensinamentos.

RESUMO

Introdução: Com o aumento significativo da utilização de aplicações móveis, tem gerado a necessidade de desenvolvimento destas aplicações de forma eficiente e com qualidade, visando atender a demanda e as expectativas de clientes e usuários. **Objetivo:** Avaliar o esforço para o desenvolvimento de um mesmo aplicativo em duas plataformas (nativa e multiplataforma), destacar as potencialidades e fragilidades de cada uma delas. **Metodologia:** Um estudo será estabelecido, sendo que para o desenvolvimento nativo será usado a ferramenta Android Studio e para a multiplataforma será usado o framework PhoneGap. Para medição do esforço, a métrica Pontos por Casos de Uso será adotada para estimar o custo e tempo de ambos tipos de plataforma. **Resultado:** Um planejamento do esforço de tempo foi realizado usando o método GQM (Objetivo, Questão e Métrica). A partir disto, o desenvolvimento em ambas as plataformas foi realizado e medido. Durante o processo, potencialidades e fragilidades foram encontradas de acordo com os dois tipos de plataformas utilizadas para o desenvolvimento de um mesmo produto. Foi necessário um maior esforço para o desenvolvimento na multiplataforma do que o nativo. **Conclusão:** Destaca-se a necessidade de entender, de forma clara, as funcionalidades do aplicativo a ser desenvolvido e qual mercado deseja atingir. Além disto, devem-se levar em consideração as habilidades, experiência e tempo de prática da equipe em cada tipo de desenvolvimento para determinar a escolha de qual plataforma utilizar. Esses aspectos tem forte influência no processo de medição e estimativa.

Palavras chaves: Desenvolvimento mobile. *Android Studio*. *Framework PhoneGap*. Métrica de Software. Pontos por Caso de Uso.

ABSTRACT

Introduction: With the significant increase in the use of mobile applications, it has generated the need to develop these applications efficiently and with quality, aiming to meet the demand and expectations of customers and users. **Objective:** To evaluate the effort to develop the same application in two platforms (native and multiplatform), highlighting the potentialities and weaknesses of each one of them. **Methodology:** A study will be established, and for the native development will be used the Android Studio tool and for the multiplatform will be used the framework PhoneGap. For effort measurement, the Points of Use metric will be used to estimate the cost and time of both platform types. **Result:** A time effort planning was performed using the GQM method (Objective, Question and Metrics). From this, the development on both platforms was realized and measured. During the process, potentialities and weaknesses were found according to the two types of platforms used for the development of the same product. It took a greater effort to develop cross-platform than native. **Conclusion:** The need to clearly understand the features of the application to be developed and which market to achieve is highlighted. In addition, the team's skills, experience and time in each type of development should be taken into account in determining which platform to use. These aspects have a strong influence on the measurement and estimation process.

Keywords: Mobile development. Android Studio. Framework PhoneGap. Software Metrics. Points for Use Case.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE SIGLAS E ABREVIATURAS

APIS	*Application Programming Interface*
Apps	Aplicativos
CSS	*Cascading Style Sheets*
GPS	Global Position System
HTML5	*Hypertext Markup Language*, versão 5
IOS	*Ios Operating System*
POO	Programação Orientada a Objeto
XML	*Extensible Markup Language*
XSL	*Extensible Stylesheet Language for Transformation*
GQM	*Goal/Question/Metric*
PCU	Pontos por Casos de Uso
UC	Caso de uso
HTML	HyperText Markup Language
IDE	Ambiente de Desenvolvimento Integrado

SUMÁRIO

1. INTRODUÇÃO

Com o aumento considerável da utilização de aplicações *mobile*, os dispositivos estão se tornando mais portáteis e ubíquos, para os usuários. A cada momento as aplicações possuem novas capacidades e oferecem informações, entretenimento e comunicação para os usuários. “A sociedade encontra-se na era da mobilidade, a possibilidade de permanecer conectado sob as formas tecnológicas atuais de comunicação sem fio representa novas práticas e comportamentos de comunicação [...]” (SILVA;GOMES, 2015).

À medida que crescem esses fatores (aumento da utilização de aplicações mobile e diversidade de sistemas operacionais móveis), surge a necessidade de desenvolver aplicações (*apps*) que sejam robustas, amigáveis e eficientes para os usuários. Para o desenvolvimento dessas aplicações é necessário ter agilidade e qualidade para atender à demanda e expectativas de usuários e clientes.

> Existem três tipos de plataformas para o desenvolvimento de aplicativos móveis, os apps nativos, os apps *web* e os apps multiplataformas, com essa diversidade torna o desenvolvimento de aplicações mobile complexo, pois essas plataformas possuem diferentes funcionalidades, restrições e especialidades (TAURION, 2016, p.1).

O desenvolvimento *mobile* é um processo árduo e complexo, pois as plataformas de desenvolvimento nativas e multiplataforma possuem diferentes processos e requisitos para a sua implementação, esses fatos demostram o problema de interoperabilidade e divergências entre as plataformas existentes. Diante dessa situação, a seguinte pergunta de pesquisa foi estabelecida: quais as fragilidades e potencialidades da plataforma nativa e multiplataforma para o desenvolvimento de aplicativos móveis?

2. OBJETIVOS DA PESQUISA

2.1 OBJETIVO GERAL

Avaliar o esforço necessário para o desenvolvimento de um mesmo aplicativo em duas plataformas (nativa e multiplataforma), destacando as potencialidades e fragilidades identificadas para cada uma delas.

2.2 OBJETIVOS ESPECÍFICOS

- Definir o escopo do aplicativo e desenvolver os artefatos (Diagrama de Casos de Uso, Diagrama de Atividade);
- Definir métricas e técnica de estimativa para o desenvolvimento do mesmo aplicativo nas diferentes plataformas;
- Desenvolver um aplicativo na plataforma nativa utilizando a ferramenta Android Studio;
- Desenvolver um aplicativo na multiplataforma utilizando o framework PhoneGap;
- Reportar a medição durante o desenvolvimento do mesmo aplicativo nas diferentes plataformas;
- Identificar as potencialidades e fragilidades encontradas durante o desenvolvimento do aplicativo em ambas plataformas.

3. METODOLOGIA

Trata-se de uma pesquisa qualitativa, “onde o foco é a maior compreensão, levando em consideração verdades e produzindo informações aos interessados no estudo” (GERHARDT;SILVEIRA, 2009, p.1). A metodologia para desenvolvimento deste estudo foi assim organizada e estruturada nos itens que se seguem.

3.1 DESENVOLVIMENTO DO APLICATIVO NA PLATAFORMA NATIVA

Para desenvolver o aplicativo escolhido de forma nativa, será adotado o sistema Android, que é o sistema operacional móvel do Google. Criado em 2008, todas suas versões recebem nome de doces e está sempre em constante aperfeiçoamento. Desde sua criação já foram lançadas doze versões e as mais recentes são: Android M (Marshmallow) e Android 7.0 (Nougat). (ANDROID, 2017)

Para o desenvolvimento será utilizada a linguagem de programação Java e a IDE Android Studio 2.3.2 .

> O Android Studio é o ambiente de desenvolvimento integrado (IDE) oficial para o desenvolvimento de aplicativos Android e é baseado no IntelliJ IDEA. Além do editor de código-fonte e das ferramentas de desenvolvedor avançados do IntelliJ, o Android Studio oferece ainda mais recursos para aumentar sua produtividade na criação de aplicativos Android. (ANDROID, 2016, p.1)

3.2 DESENVOLVIMENTO DO APLICATIVO NA MULTIPLATAFORMA

No desenvolvimento mutiplataforma serão utilizados o *Cordova* e PhoneGap. O papel do *Cordova* será criar uma comunicação entre o *HTML*, *CSS* e *JavaScript* e a parte nativa do dispositivo. “O papel do *Cordova* é apenas criar essa janela de navegador para nós, e fazer a comunicação das nossas chamadas de código para chamadas nativas quando necessário. ” (LOPES, 2016).

O PhoneGap foi criado em 2009 pela empresa Nitobi e comprado em 2011 pela Adobe que doou o código para o projeto Apache (LOPES, 2016).

3.3 IDENTIFICAÇÃO DAS POTENCIALIDADES E FRAGILIDADES

A identificação das potencialidades e fragilidades, para as diferentes plataformas utilizadas nesse estudo, será realizada de maneira descritiva de acordo com as métricas encontradas ao utilizar o método GQM. Serão listadas, em uma tabela, organizadas em três colunas: fragilidades, potencialidade e o tipo de plataforma.

3.4 ESTIMATIVA DO ESFORÇO NECESSÁRIO PARA O DESENVOLVIMENTO DO APLICATIVO

Para planejar a medição e estimativa do esforço necessário para o desenvolvimneto do mesmo aplicativo, desenvolvido em duas plataformas, será adotado o método *Goal/Question/Metric* (GQM).

A medição do esforço preciso para o desenvolvimento do mesmo aplicativo utilizando as plataformas nativas e multiplataformas, é necessário para agregar mais informações no resultado final. Ter noções do tamanho do esforço que é preciso em cada modo de desenvolvimento, ajudará no momento da escolha de qual plataforma utilizar.

As métricas de *software* proporcionam uma maneira quantitativa de avaliar a qualidade dos atributos internos do produto, possibilitando avaliar a qualidade antes que ele seja criado. As métricas proporcionam as informações necessárias para criar requisitos eficazes e modelos de projeto, código sólido e testes completos (PRESSMAN, 2016).

4. JUSTIFICATIVA

A utilização dos aplicativos mobile tem crescido e o seu uso é um hábito para os usuários que os utilizam com o propósito de obter informações, entretenimento e benefícios através dos recursos que as aplicações oferecem. Com o crescimento da utilização dos aplicativos o mercado de desenvolvimento mobile cresce, conforme Lourenço (2016. p.1) aponta:

> Para os desenvolvedores de software surge uma nova oportunidade de negócio, pois, a procura por aplicativos e jogos que atendam este mercado que está crescendo na mesma proporção que cresce a criação de novos dispositivos, é muito grande [...].

O desenvolvimento de aplicativos *mobile* está subdivido em três plataformas: nativas, multiplataformas e web apps (SILVA;GOMES, 2015). Na plataforma nativa os aplicativos são desenvolvidos para sistemas operacionais específicos, na multiplataforma são desenvolvidos aplicativos híbridos através de um *framework* utilizando um modelo padrão nativo (GARAY, 2014), e nas aplicações *web* apps, são aplicativos habilitados para a navegadores *web* que podem ser executados em diferentes dispositivos móveis (SILVA;GOMES, 2015),

As diferenças entre as plataformas, proporcionam dúvidas nos profissionais, no momento de optar por qual melhor atende as suas expectativas, pois cada plataforma possui suas próprias características e peculiaridades diferentes. Esse estudo visa contribuir com a exposição das fragilidades, potencialidades, semelhanças e divergências existentes em cada plataforma.

5. REFERENCIAL TEÓRICO

Serão apresentados conceitos referentes às plataformas mobile existentes e suas características, os sistemas operacionais mobile utilizados no mercado, o *framework PhoneGap* usado para o desenvolvimento na multiplataforma e o paradigma de linguagem de programação orientado a objetos. Além de uma técnica para estimativa e medição do esforço necessário para o desenvolvimento em ambas plataformas.

5.1 PROGRAMAÇÃO ORIENTADA A OBJETOS (POO)

Programação Orientada a Objetos ou, POO, “é um paradigma de programação de computadores onde se usam classes e objetos para representar e processar dados”. (SANTOS, 2003, p.1)

A programação orientada a objetos utiliza do encapsulamento. De acordo com SANTOS (2003) “ocultar dados permitindo apenas operações especializadas que manipulem essas informações, chama-se encapsulamento, e é um dos benefícios mais tangível da programação orientada a objetos. ”

A herança em programação orientada a objetos, é o que possibilita criar uma nova classe, a partir de classes existentes, “é uma forma de reutilização de software em que uma classe é criada, pois absorve membros de uma classe existente, e aprimora as novas classes com capacidades novas ou modificadas “ (DEITEL, 2015,p.279).

Abstração em *POO* é onde são criado as classes. Para Santos (2003, p.4), “o modelo é uma simplificação do mundo real, os dados contidos no modelo são somente os relevantes à abstração do mundo real sendo feita”.

O Site Impacta (2016, p.1) explanou que:

> As quatro maiores plataformas de dispositivos móveis – iPhone, Android, Windows e BlackBerry – usam programação baseada em objetos [...]. Como podemos ver, praticamente todas as plataformas de desenvolvimento de aplicativos mobile são orientadas a objetos, deixando a importância da mesma bem clara.

Portanto, a programação orientada a objetos possui características que são essenciais para o desenvolvimento móvel, além da maioria dos sistemas dispositivos fazerem uso dessa técnica para os seus próprios desenvolvimentos.

5.2 PLATAFORMAS MOBILE

O desenvolvimento de aplicativos mobile é dividido em três plataformas: nativa, multiplataforma e *Web App*. Nas plataformas nativas, os aplicativos são desenvolvidos para sistemas operacionais próprios, na multiplataforma os aplicativos necessitam de um *framework*[1], para a sua implementação, e nos aplicativos *Web App* utilizam recursos da *Web*, são usados através de um navegador dos dispositivos móveis.

> Deve-se salientar que dentro da camada de aplicação pertencente ao ecossistema móvel há uma subdivisão no que diz respeito ao processo de desenvolvimento de softwares. Compõem essa subdivisão as chamadas aplicações nativas, aplicações web e as aplicações híbridas ou multiplataformas (Cross Platform). (NABUCO; FAÇANHA; ARAÚJO, 2012, p.4).

As aplicações nativas requerem uma linguagem de programação específica como a *Objective* – C na plataforma *IOS* (APPLE, 2016), e a linguagem Java na plataforma *Android* (ANDROID, 2016). A aplicação híbrida é uma junção das funcionalidades das apps nativa da aplicação Web, que combina recursos das duas aplicações e as aplicações Web utilizam uma *Web Browser*[2] para o seu funcionamento (NABUCO et al., 2012, p.1).

5.2.1 PLATAFORMA NATIVA

Na plataforma nativa os aplicativos são desenvolvidos para sistemas operacionais específicos e possuem características particulares, por isso precisam

---

[1] Um *framework* é uma abstração que une códigos semelhantes provendo uma funcionalidade genérica (FERREIRA, 2015, p.1).

[2] Web Browser é um programa de computador que habilita seus usuários a interagirem com documentos virtuais da Internet, também conhecidos como páginas da web (DALMA, 2015, p.1).

atender especificações do ambiente nativo dos dispositivos móveis. As aplicações nativas podem acessar todos os recursos dos dispositivos móveis, como: GPS, lista de contatos, câmera e acelerômetro.

“Aplicações nativas podem-se observar características exclusivas, que está intrinsecamente ligado ao sistema operacional de um determinado dispositivo, tornando o desenvolvimento destas aplicações dependente dessas características. ” (NABUCO et al., 2012, p.1)

Os *apps* estão ligados ao ambiente nativo, executando apenas em plataformas particulares, esses apps seguem padrões técnicos determinados pelas plataformas.

> As aplicações nativas seguem-se os padrões técnicos, da interface e de experiência de usuário promovendo uma interação entre a aplicação nativa e o usuário. As aplicações nativas têm acesso por meio de APIS, os recursos dos dispositivos móveis, como sensores, câmera, GPS, contatos e e-mail. (EL-KASSAS et al., 2015, p.1).

5.2.2 PLATAFORMA HÍBRIDA

A plataforma híbrida representa uma categoria de aplicações *web*. Nela a aplicação é executada dentro do ambiente nativo dos dispositivos móveis, sendo que os aplicativos híbridos têm o seu código fonte fundamentado em *HTML5*. Segundo GARAY (2014, p.1) a aplicação híbrida é “uma aplicação que roda em uma visão *web* de um aplicativo nativo. Dito a grosso modo, uma aplicação híbrida usa um aplicativo nativo como seu contêiner para torná-la como um aplicativo nativo”.

Aplicações híbridas utilizam *o HTML5, CSS* e *Java Script* e um *rendering engine* [3] nativo para acessar os recursos dos sistemas operacionais *mobile*. De acordo com GARAY (2014) os aplicativos híbridos possuem dois elementos: componente *web* baseado em *HTML5*, e um padrão de projeto nativo que permitem acessar os recursos peculiares das plataformas e dos dispositivos.

Portanto, os aplicativos híbridos dizem respeito ao ato de combinar elementos de aplicações nativas e recursos da *web*, realizando uma junção desses elementos. A

[3] É um software que transforma conteúdo em linguagem de marcação (como HTML, XML, etc.) e informações de formatação (como CSS, XSL, etc.) em um conteúdo formatado para ser exibido em uma tela (EL-KASSAS et al, 2015, p.1).

utilização de linguagens de programação *web* como *HTML5* e o empacotamento das funcionalidades da plataforma nativa resulta em um aplicativo que executará em diferentes plataformas (*Android, IOs,* entre outras).

5.2.3 PLATAFORMA WEB APPS

Um aplicativo *web* móvel é uma aplicação que é acessada através do navegador *web* do dispositivo móvel. É construída com as principais tecnologias*: HTML*, *CSS* e *Java Script*. Na plataforma *Web Apps*, os aplicativos são executados em um container[4], que simula a interface de um aplicativo, mas funciona diretamente no navegador de algum dispositivo móvel.

> O desenvolvimento de aplicações web, para efeitos de definição, caracteriza-se [...] como sendo uma ou um conjunto de páginas web entregues via protocolo HTTP (Protocolo de Transferência de Hipertexto) usando requisições do lado do servidor ou do lado do cliente. (NABUCO; FAÇANHA; ARAÚJO, 2012, p.5).

Portanto as aplicações *web Apps* são aplicações habilitadas para a internet, utilizando tecnologias *web*, e leva em conta aspectos próprio dos dispositivos móveis, como tamanho de tela e modo de uso, segundo WHITE (2013) classifica *web Apps* como *sites* otimizados para telas menores, o sistema operacional do dispositivo não tem domínio sobre seu conteúdo e suas funcionalidades.

5.3 SISTEMAS OPERACIONAIS MÓVEIS

Sistemas operacionais móveis são desenvolvidos para dispositivos móveis, os quais possibilita executar os aplicativos.

> Um sistema operacional móvel é um software que permite dispositivos móveis executar programas, ele é responsável por apresentar as informações e acesso aos aplicativos, também gerenciam a conexão Wi-Fi dos celulares e também acessam os recursos dos aparelhos móveis (ROUSE, 2011, p.1).

---

[4] Componentes que tem a capacidade de abrigar outros componentes (WHITE, 2013, p.1).

O mercado de aparelhos móveis é dividido por diferentes fabricantes, a existência de múltiplas plataformas cria uma variedade de aplicativos, e cada plataforma é codificada para serem executadas em sistemas operacionais específicos. A Tabela 01 apresenta os principais sistemas operacionais utilizados no mercado, bem como sua quantidade de usuários:

Tabela 1 - *Ranking* de Sistemas Operacionais *Mobile.*

Sistema Operacional	2016 (milhões)
Android	296,9
Ios	44,3
Windows	1,9
Blackberry	0,4
Outros	0,680
Total	344,359.7

Fonte: Adaptado de Gartner (2016).

De acordo com a Tabela 01, o *Android* destaca-se como o sistema operacional mais utilizado no mundo. Ele é um sistema de código aberto, compatível com as multimídias disponível em aplicativos móveis. (ANDROID, 2016). O *Iphone Operating System* (*IOs*) que aparece em segundo lugar. É um sistema operacional móvel da *Apple* Inc, desenvolvido originalmente para o *Iphone*. (APPLE, 2016). Em terceiro lugar está o *Windows Phone,* que utiliza o padrão de aplicativos usados para o *Windows* na versão de computadores. (MICROSOFT, 2016). Em quarto lugar é o *Blackberry – Rim*, conhecido por direcionar os seus dispositivos móveis para o mercado corporativo, permitindo a produção de conteúdo profissional (BLACKBERRY, 2016).

5.4 PHONEGAP

PhoneGap é um *framework* que facilita a implementação de aplicativos móveis multiplataforma. É uma estrutura móvel de código aberto que permite a criação de aplicações móveis e acesso do *hardware* nativo do dispositivo móvel, através de uma

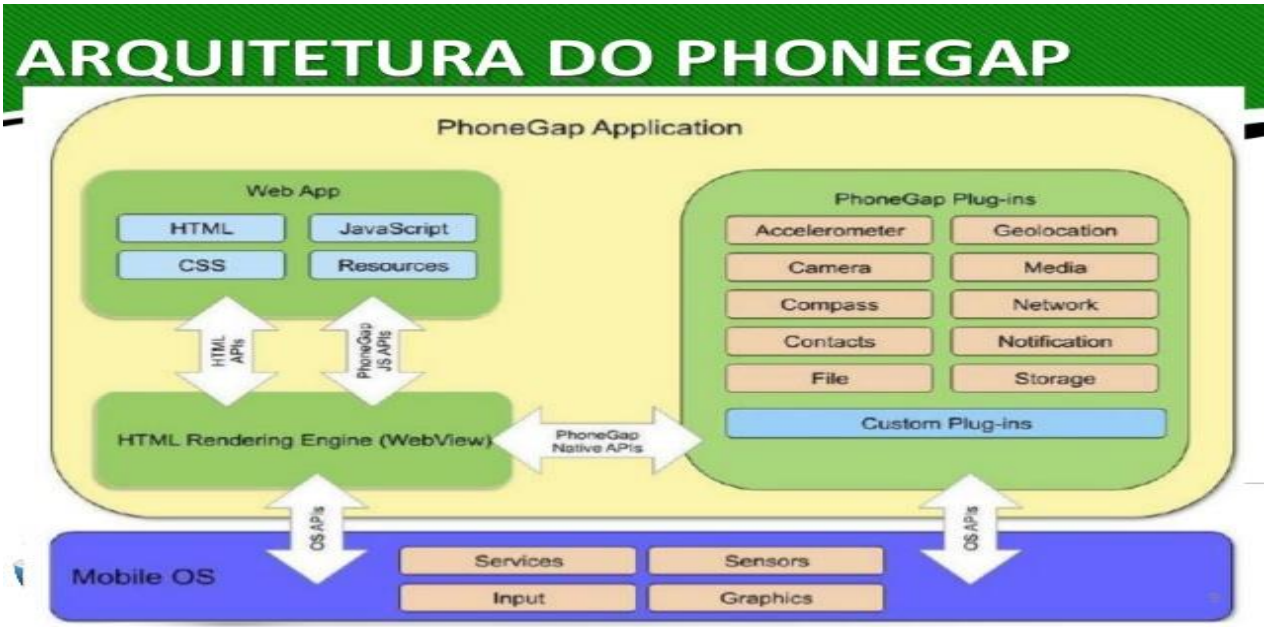
API *Java script*, independente de qual plataforma esteja utilizando o *Phonegap*. (ADOBE PHONEGAP BUILD, 2016).

Carvalho (2015, p.1) afirma que:

> O Phonegap é um *framework* multiplataforma que possibilita a criação de aplicativos móveis utilizando tecnologia web [...] para serem executados em diversas plataformas como *Android, iOS, Windows Phone, BlackBerry, Firefox OS,* entre outros. Ele funciona como um empacotador de código que através de uma *WebView* permite utilização de código *HTML5* no desenvolvimento do app e permite acesso as *APIS* nativas de cada plataforma.

O *PhoneGap* utiliza tecnologias *web* como o: *HTML5*, *CSS3* e *Java Script*, e os aplicativos são executados em pacotes direcionados para cada plataforma que consistem em padrões compatíveis com a API nativas para acessar os recursos dos dispositivos (ADOBE PHONEGAP BUILD, 2016).

Figura 1: Arquitetura Phonegap.



O *PhoneGap* fornece uma API que permite acessar as funcionalidades do sistema operacional nativo usando *Java Script*, o desenvolvedor implementa a lógica do aplicativo usando o *Java Script* e a API *PhoneGap* que é responsável por trabalhar com a comunicação do sistema operacional nativo.

5.5 PONTOS POR CASO DE USO

Em 1993, Gustav Karner criou os Pontos por Casos de Uso (PCU) com o propósito de estimar recursos para projetos de *software* orientados a objetos desenvolvidos pela empresa Objectory AB. (BELGAMO;FABBRI, 2004, p.1).

Para utilização da métrica PCU, é necessário seguir alguns passos para obtenção de um resultado eficiente. O processo de contagem constitui-se dos seguintes passos (MEDEIROS, 2004):

1. Contar os atores e identificar suas complexidades;
2. Contar os casos de uso e identificar suas complexidades;
3. Calcular os PCUs não ajustados;
4. Determinar o fator de complexidade técnica;
5. Determinar o fator de complexidade ambiental;
6. Calcular os PCUs ajustados;

Após isto, será obtido a quantidade de pontos totais para desenvolvimento de um determinado projeto, podendo estes serem convertidos em tempo e/ou custo.

A planilha do PCU, é constituída de 5 seções. Começando pela seção da **Error! Reference source not found.** 2.

Figura 2: Mensurando Complexidade dos Atores

1.0. Mensurando Complexidade dos Atores

Ator	Interface	Peso		Qtd. Atores	Valor
Simples	Interface de programa (API)	1		0	0
Médio	Protocolo (Ex.:TCP/IP) ou interface em modo texto	2		0	0
Complexo	Interface gráfica	3		0	0
			Total	0	0

Fonte: O autor.

Nessa seção, é possível o “Ator” possui três classificações, sendo a simples onde outro sistema é acessado através de uma API de programação. Médio quando outro sistema é acessado através da rede. E por último, o complexo, onde um usuário interage através de uma interface gráfica. Na Figura 3, é mostrado a próxima seção da planilha.

Figura 3:Mensurando Complexidade dos Casos de Uso

2.0. Mensurando Complexidade dos Use Cases

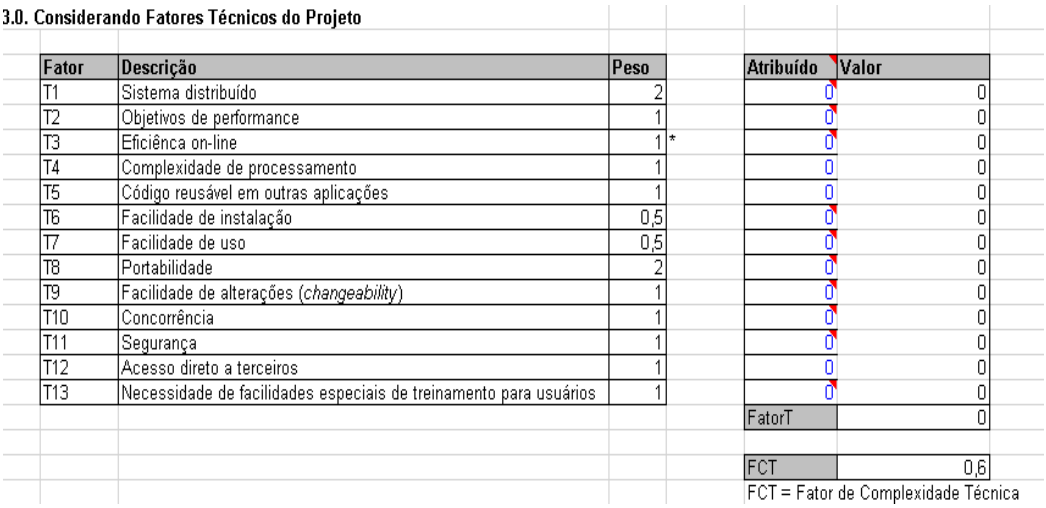
Caso de L	Descrição	Peso		Qtd. Casos de Uso	Valor
Simples	< 3 transações ou < 5 classes de análise	5		0	0
Médio	4-7 transações ou 5 a 10 classes de análise	10		0	0
Complexo	> 7 transações ou > 10 classes de análise	15		0	0
			Total	0	0
			PCUNA	0	

PCUNA = Pontos de Casos de Uso Não Ajustados

Fonte: O autor.

Na segunda seção é tratado a complexidade dos casos de uso, que foram elaborados durante o processo de definição do escopo e funcionalidades do sistema. São classificados em simples, médio e complexo. Variando de acordo com a quantidade de transações. Na Figura 4 pode-se ver a seção 3.

Figura 4:Fatores Técnicos



3.0. Considerando Fatores Técnicos do Projeto

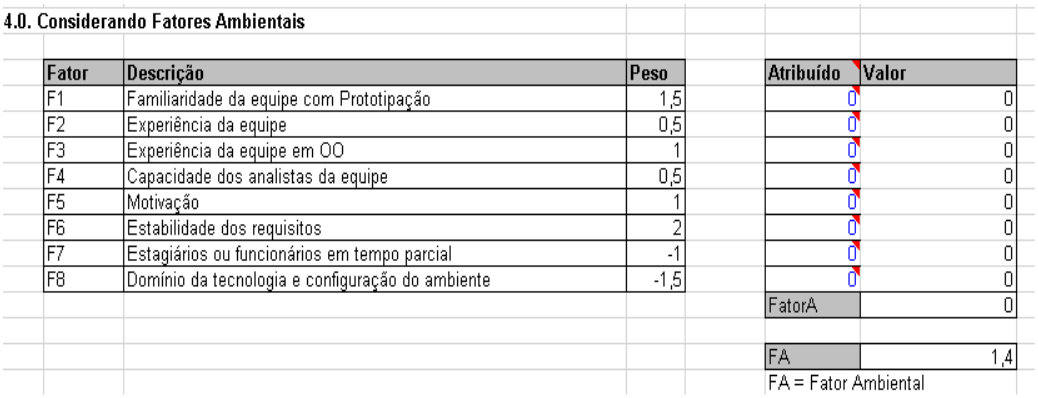
Fator	Descrição	Peso		Atribuído	Valor
T1	Sistema distribuído	2		0	0
T2	Objetivos de performance	1		0	0
T3	Eficiênca on-line	1	*	0	0
T4	Complexidade de processamento	1		0	0
T5	Código reusável em outras aplicações	1		0	0
T6	Facilidade de instalação	0,5		0	0
T7	Facilidade de uso	0,5		0	0
T8	Portabilidade	2		0	0
T9	Facilidade de alterações (*changeability*)	1		0	0
T10	Concorrência	1		0	0
T11	Segurança	1		0	0
T12	Acesso direto a terceiros	1		0	0
T13	Necessidade de facilidades especiais de treinamento para usuários	1		0	0
				FatorT	0
				FCT	0,6

FCT = Fator de Complexidade Técnica

Fonte: O autor.

Na seção 3, é analisado os fatores técnicos do projeto. Para cada fator da tabela deve ser atribuído um valor que mostra a influência do fator no sistema. A Figura 5 mostra a seção de número 4.

Figura 5:Fatores Ambientais



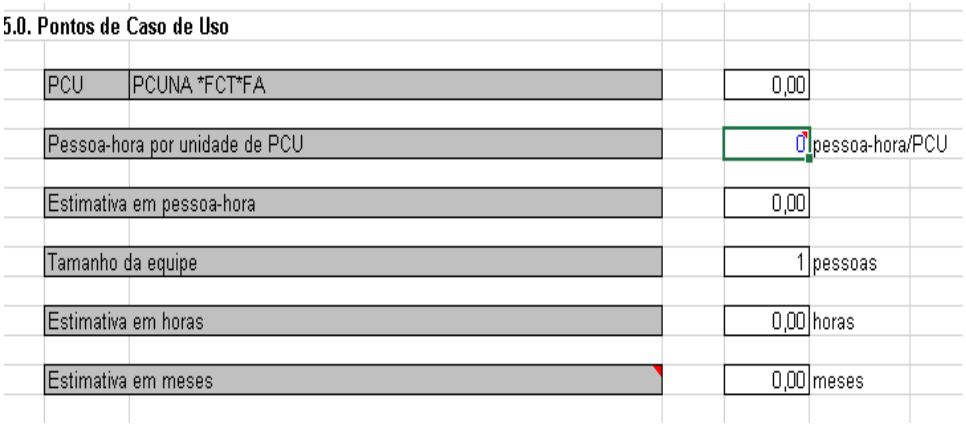
4.0. Considerando Fatores Ambientais

Fator	Descrição	Peso	Atribuído	Valor
F1	Familiaridade da equipe com Prototipação	1,5	0	0
F2	Experiência da equipe	0,5	0	0
F3	Experiência da equipe em OO	1	0	0
F4	Capacidade dos analistas da equipe	0,5	0	0
F5	Motivação	1	0	0
F6	Estabilidade dos requisitos	2	0	0
F7	Estagiários ou funcionários em tempo parcial	-1	0	0
F8	Domínio da tecnologia e configuração do ambiente	-1,5	0	0
			FatorA	0
			FA	1,4
			FA = Fator Ambiental	

Fonte: O autor.

Nessa seção é considerado os fatores ambientais do projeto. Da mesma forma que os fatores técnicos da seção 3 são analisados, aqui também é determinado um valor para cada fator com base na influência que esse fator tem sobre o projeto. Na Figura 6 nota-se a última seção da planilha de estimativa.

Figura 6:Pontos de Caso de Uso



5.0. Pontos de Caso de Uso

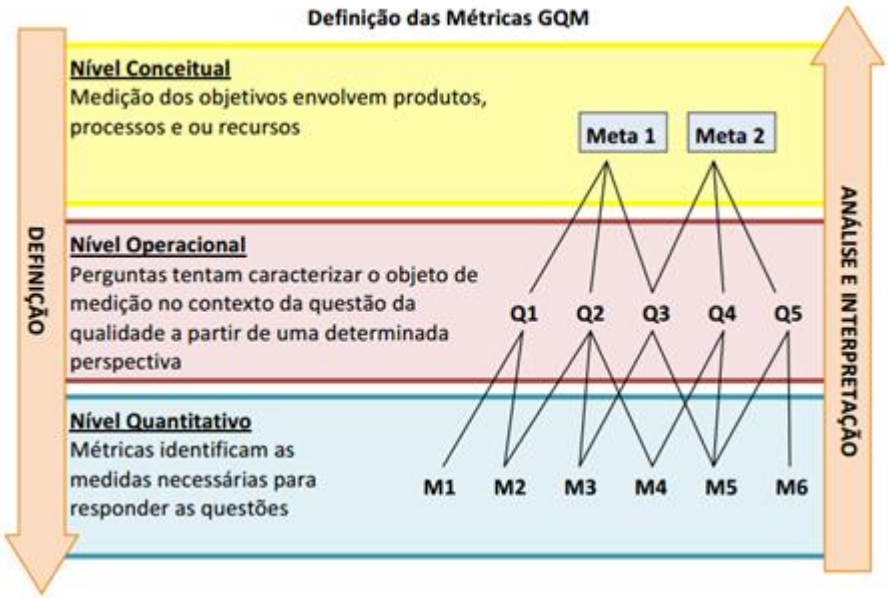
PCU	PCUNA *FCT*FA	0,00	
Pessoa-hora por unidade de PCU		0	pessoa-hora/PCU
Estimativa em pessoa-hora		0,00	
Tamanho da equipe		1	pessoas
Estimativa em horas		0,00	horas
Estimativa em meses		0,00	meses

Fonte: O Autor.

Na última seção, é realizado as contas com base nos valores atribuídos em todas as seções anteriores, resultando assim

5.6 MÉTODO GQM

Figura 7:Método GQM.



Fonte: SILVA, C. et al, 2009.

Como existem várias medidas diferentes para um software, é necessário saber escolher de forma correta quais serão adotadas para o projeto. “Essa escolha deve considerar características como o custo de aplicação de uma medida, o que é obvio, considerar os objetivos do projeto” (KOSCIANSKI, 2007, p.224).

Uma das formas de tratar o planejamento da medição de um projeto é o método GQM (*Goal-Question-Metric)*. De acordo com Dal’Osto (2003), o método GQM possui três níveis:

- Nível Conceitual: Uma meta é definida para um objeto de acordo com uma variedade de razões;
- Nível Operacional: São utilizadas questões para definir a melhoria e garantir a meta selecionada;
- Nível Quantitativo: É associado um conjunto de dados às questões, afim de respondê-las de forma quantitativa.

6. RESULTADOS ALCANÇADOS

6.1 ESCOPO DO APLICATIVO

Trata-se de um aplicativo móvel para plataforma *Android*, onde será possível o usuário inserir a idade da criança (em meses), respondendo posteriormente questões, de acordo com a caderneta disponibilizada pelo Ministério da Saúde em postos de saúde e hospitais do brasil todo, que irão avaliar o crescimento e desenvolvimento de crianças de 1 a 24 meses de idade.

A seguir um exemplo de caderneta com os marcos de crescimento e desenvolvimento da criança.

Figura 8:Caderneta de Acompanhamento do Desenvolvimento.

Ficha de acompanhamento do desenvolvimento

Registro: Nome:

Data de nascimento ___/___/___	Marcos do desenvolvimento (resposta esperada)	Idade (meses)												
		1	2	3	4	5	6	7	8	9	10	11	12	13	15
	Abre e fecha os braços em resposta à estimulação *(Reflexo de Moro)*														
	Postura: barriga para cima, pernas e braços fletidos, cabeça lateralizada														
	Olha para a pessoa que a observa														
	Dá mostras de prazer e desconforto														
	Fixa e acompanha objetos em seu campo visual														
	Colocada de bruços, levanta a cabeça momentaneamente														
	Arrulha e sorri espontaneamente														
	Começa a diferenciar dia/noite														
	Postura: passa da posição lateral para linha média														
	Colocada de bruços, levanta e sustenta a cabeça apoiando-se no antebraço														
	Emite sons - Balbucia														
	Conta com a ajuda de outra pessoa mas não fica passiva														
	Rola da posição supina para prona														
	Levantada pelos braços, ajuda com o corpo														
	Vira a cabeça na direção de uma voz ou objeto sonoro														
	Reconhece quando se dirigem a ela														
	Senta-se sem apoio														
	Segura e transfere objetos de uma mão para a outra														
	Responde diferentemente a pessoas familiares e ou estranhos														
	Imita pequenos gestos ou brincadeiras														
	Arrasta-se ou engatinha														
	Pega objetos usando o polegar e o indicador														
	Emprega pelo menos uma palavra com sentido														
	Faz gestos com a mão e a cabeça (tchau, não, bate palmas, etc.)														

Fonte: Ministério da Saúde, 2002.

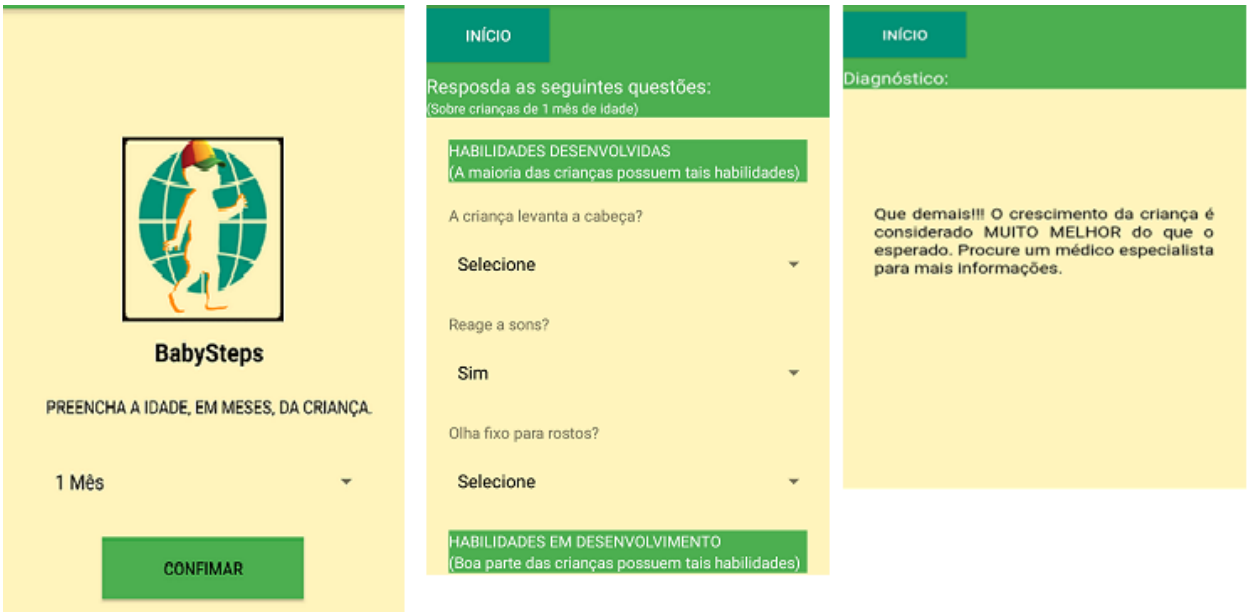
6.1.1 PROTÓTIPOS DE INTERFACE

Protótipos de interface foram elaborados utilizando a ferramenta PhotoScape, sendo elas: tela inicial, tela de questionário e tela de resposta.

A **Error! Reference source not found.**, trata-se do protótipo de interface da tela inicial do aplicativo, contendo o logo, o campo de inserção de idade (em meses) e botão de confirmação. Também do protótipo de interface da tela de questionário, onde o usuário irá responder questões que são marcos do crescimento e desenvolvimento de crianças de 1 a 24 meses de idade.

Ainda na **Error! Reference source not found.** está o protótipo da tela de resposta que será exibida ao usuário após responder todo o questionário de acordo com a idade (em meses) selecionada na tela principal.

Figura 9: Protótipos – PhotoScape.

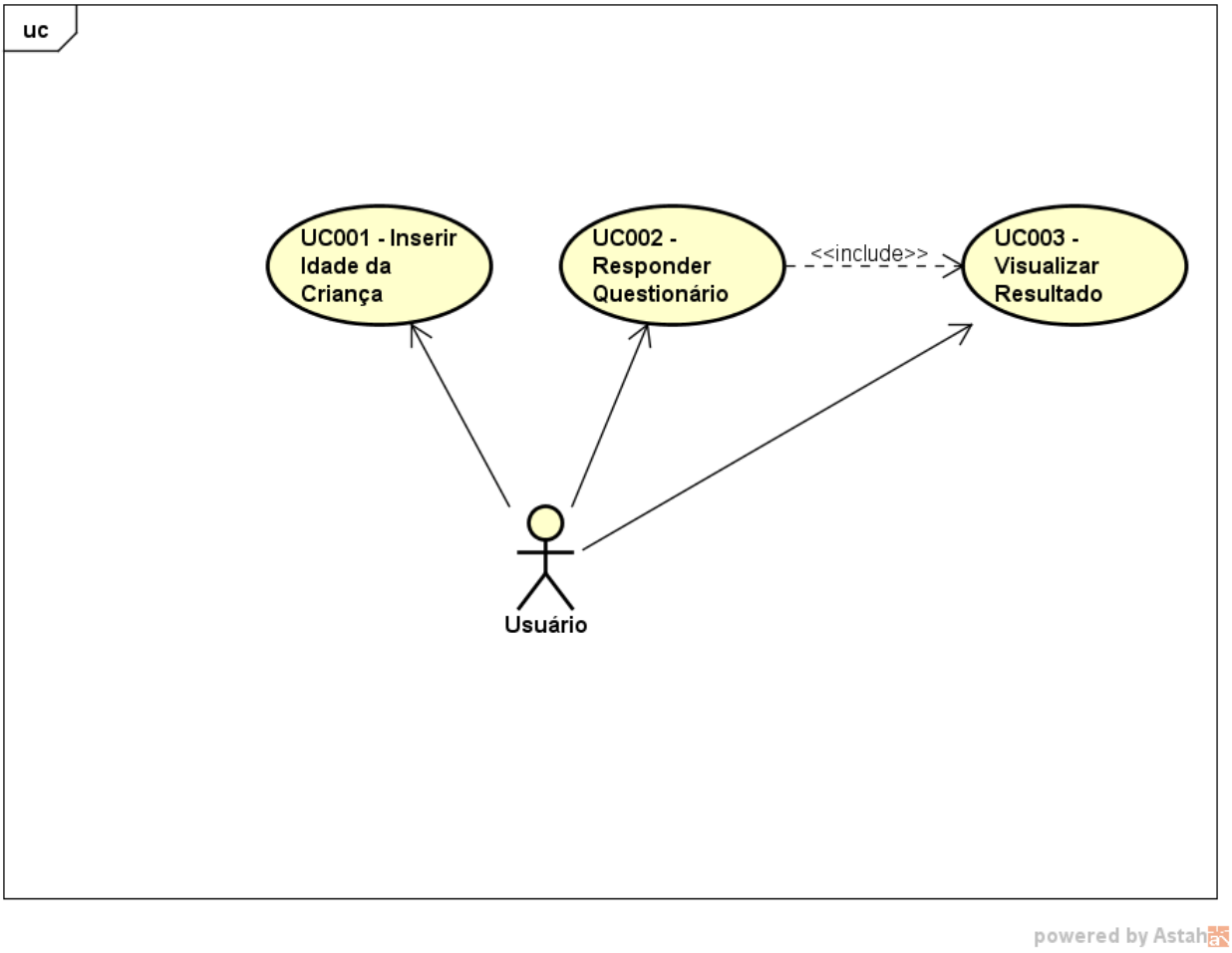


Fonte: O autor.

6.1.2 DIAGRAMA DE CASO DE USO

A **Error! Reference source not found.** 10 mostra o diagrama de caso de uso do Usuário.

Figura 10:Diagrama de Caso de Uso – Astah Community.

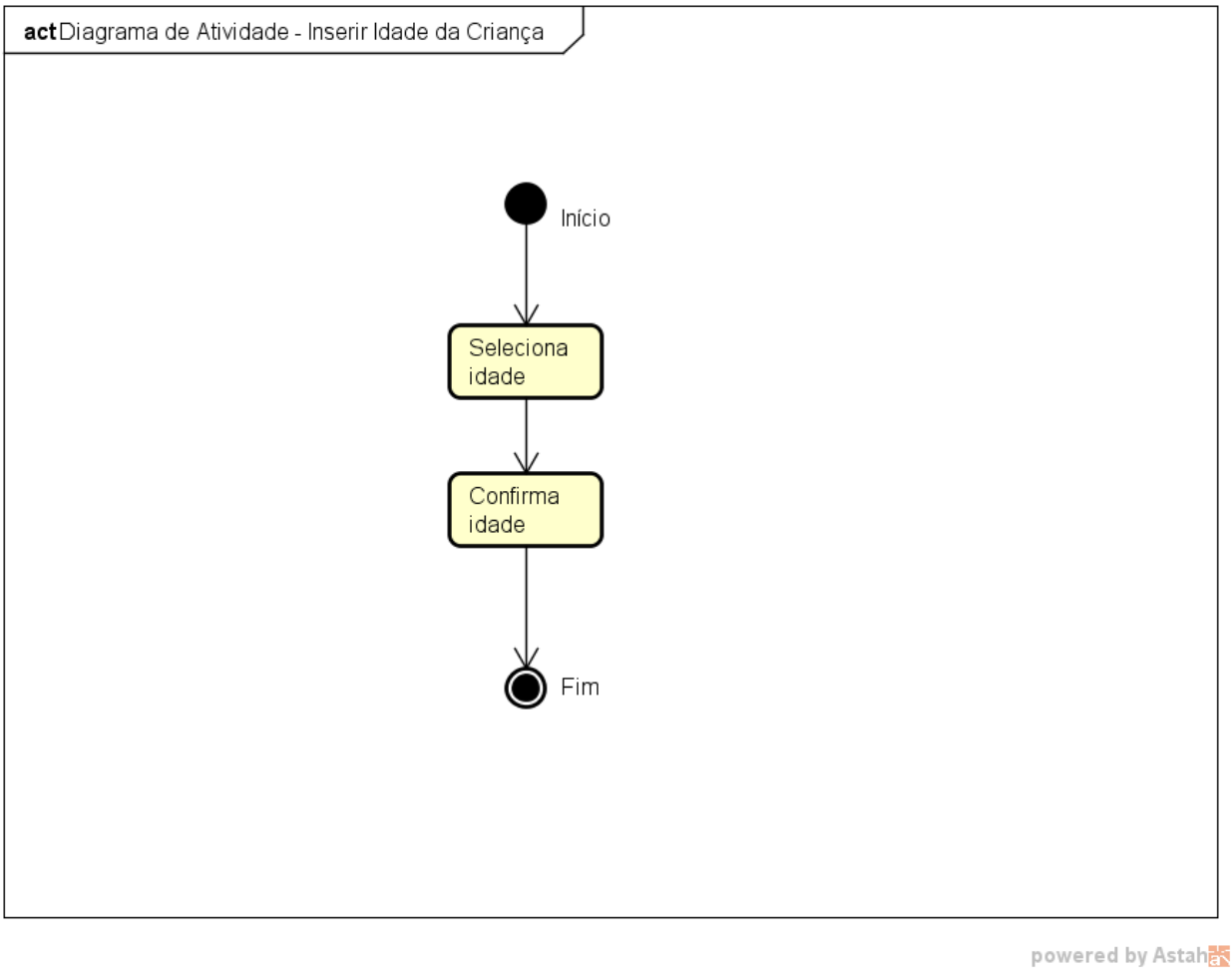


Fonte: O autor.

6.1.2.1 UC001 – INSERINDO IDADE CRIANÇA

O principal foco deste caso de uso é possibilitar que o cliente insira a de idade da criança (em meses). A seguir é apresentado o diagrama de atividade, que são atividades do caso de uso Inserir Idade da Criança.

Figura 11:Diagrama de Atividade - Caso de Uso - UC001 - Inserir Idade da Criança – Astah Community.



Fonte: O autor.

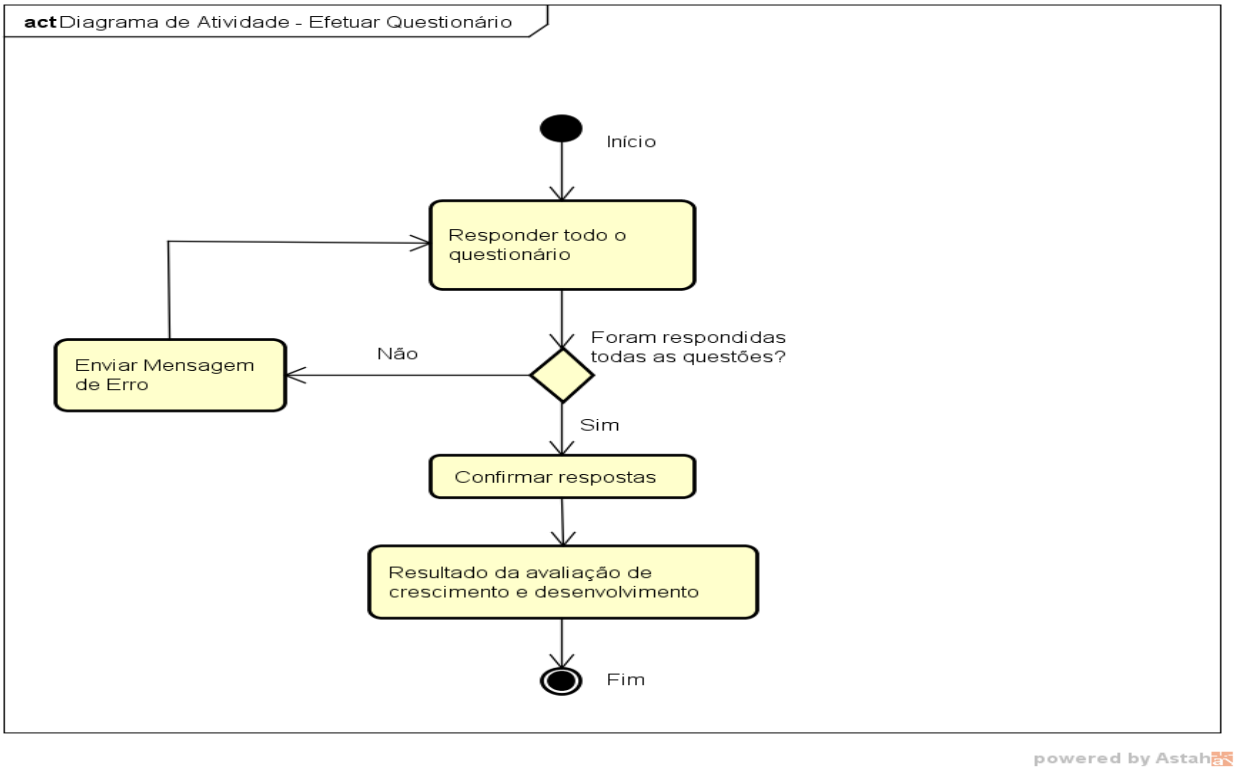
- Fluxo Principal
    - O usuário selecionará a idade (em meses) da criança e posteriormente confirmará a mesma.
    - O sistema disponibiliza a tela do questionário.

6.1.2.2 UC002 – RESPONDER QUESTIONÁRIO

Esse caso de uso tem como objetivo possibilitar que o usuário responda as perguntas de acordo com a idade selecionada no Caso de Uso Inserir Idade da Criança e também abrange o Caso de Uso Visualizar Resultado, pois estão ligados diretamente.

A seguir é mostrado o diagrama de atividade, que são atividades do caso de uso Responder Questionário.

Figura 12:Diagrama de Atividade - Caso de Uso - UC002 – Responder Questionário – Astah Community.



Fonte: O autor.

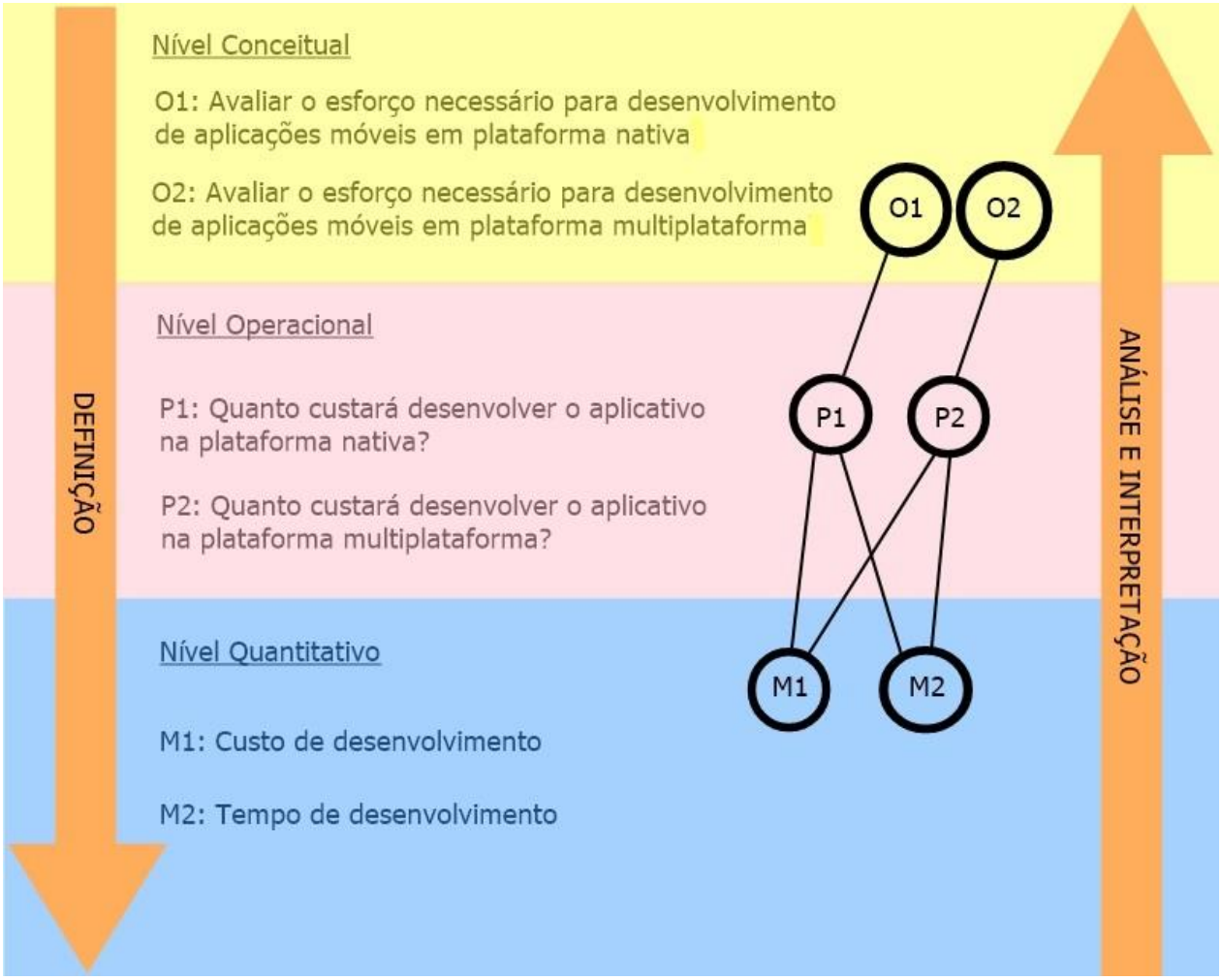
- Fluxo Principal
  - O usuário irá responder todas as questões existentes no questionário, depois confirmará suas respostas e então o sistema irá retornar uma avaliação de crescimento e desenvolvimento da criança.
- Fluxo Alternativo 1

- Será verificado que o usuário não respondeu todas as questões, mostrando uma mensagem informando a situação, voltando ao fluxo normalmente.

6.2 GQM (*Goal-Question-Metric*)

Esta seção possui o método GQM elaborado para facilitar a definição de metas e questões, chegando as métricas que foram utilizadas para atingir o objetivo final do estudo.

Figura 13:Método GQM do Projeto.



Fonte: O autor.

Nesse projeto serão utilizadas métricas orientadas a tempo e custo do desenvolvimento. Como medida para esta métrica foi adotada o número de Pontos por Caso de Uso (PCU) e número de horas por Pontos de Caso de Uso, capaz de determinar resultado para ambas.

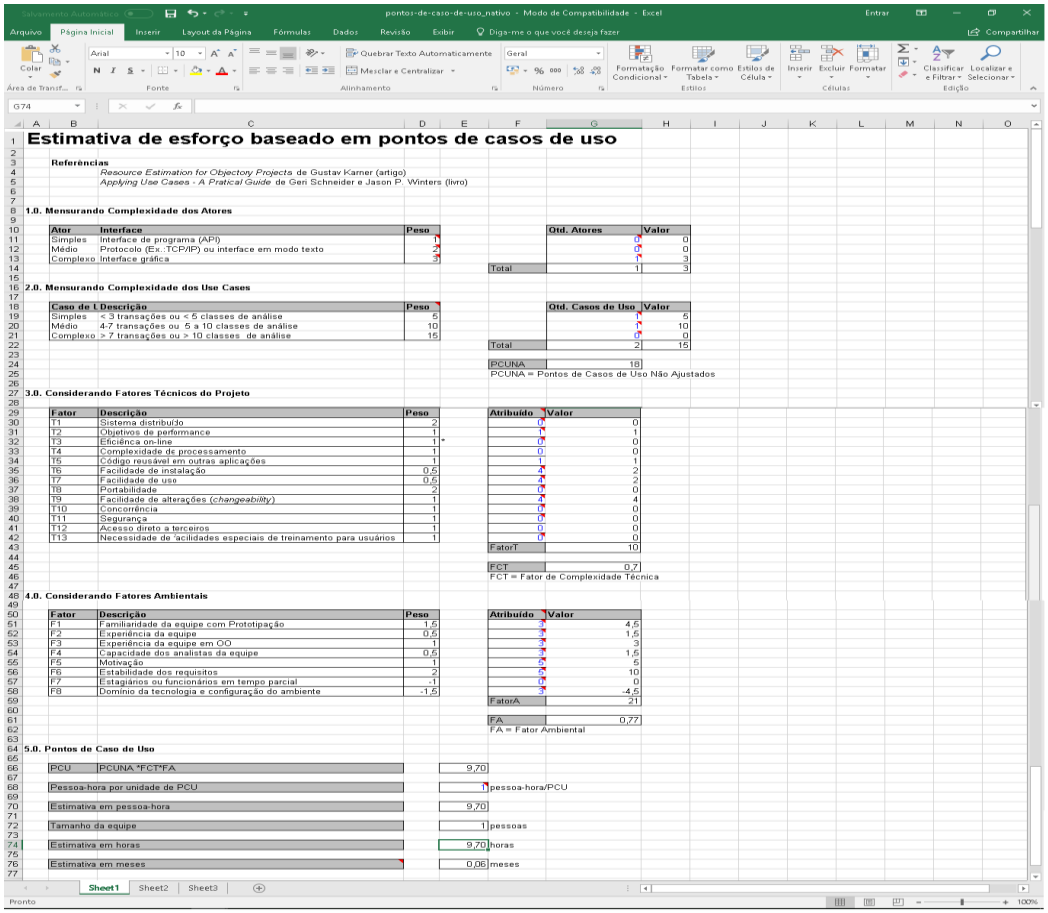
6.3 PROCESSO DE ESTIMATIVA

Inicialmente, o Diagrama de Caso de Uso foi elaborado. A partir dele, duas estimativas usando a planilha de PCU foi realizada.

O processo de estimativa se iniciou com o desenvolvimento na plataforma nativa. Através da planilha do Microsoft Excel, utilizando Pontos por Caso de Uso (PCU), foram inseridas as informações de acordo com o escopo do projeto, desenvolvedor e outras informações necessárias para estimar.

Na **Error! Reference source not found.** 14, pode-se verificar o resultado final dessa estimativa. Foi obtido um valor de número de pontos de caso de uso equivalente a 9,70. O que levou a um número de horas por pontos de caso de uso de 9,7horas, levando em consideração que a equipe de desenvolvimento só possui uma pessoa.

Figura 14:PCU Desenvolvimento Nativo – Excel.



Estimativa de esforço baseado em pontos de casos de uso

Referências

Resource Estimation for Objectory Projects de Gustav Karner (artigo)
Applying Use Cases - A Pratical Guide de Geri Schneider e Jason P. Winters (livro)

1.0. Mensurando Complexidade dos Atores

Ator	Interface	Peso
Simples	Interface de programa (API)	1
Médio	Protocolo (Ex.:TCP/IP) ou interface em modo texto	2
Complexo	Interface gráfica	3

Otd. Atores	Valor
0	0
0	0
1	3
Total	1	3

2.0. Mensurando Complexidade dos Use Cases

Caso de L	Descrição	Peso
Simples	< 3 transações ou < 5 classes de análise	5
Médio	4-7 transações ou 5 a 10 classes de análise	10
Complexo	> 7 transações ou > 10 classes de análise	15

Otd. Casos de Uso	Valor
1	5
1	10
0	0
Total	2	15

PCUNA 18
PCUNA = Pontos de Casos de Uso Não Ajustados

3.0. Considerando Fatores Técnicos do Projeto

Fator	Descrição	Peso
T1	Sistema distribuído	2
T2	Objetivos de performance	1
T3	Eficiência on-line	1
T4	Complexidade de processamento	1
T5	Código reusável em outras aplicações	1
T6	Facilidade de instalação	0,5
T7	Facilidade de uso	0,5
T8	Portabilidade	2
T9	Facilidade de alterações (*changeability*)	1
T10	Concorrência	1
T11	Segurança	1
T12	Acesso direto a terceiros	1
T13	Necessidade de 'acilidades especiais de treinamento para usuários	1

Atribuído	Valor
0	0
1	1
0	0
0	0
1	1
4	2
4	2
0	0
4	4
0	0
0	0
0	0
FatorT	10

FCT 0,7
FCT = Fator de Complexidade Técnica

4.0. Considerando Fatores Ambientais

Fator	Descrição	Peso
F1	Familiaridade da equipe com Prototipação	1,5
F2	Experiência da equipe	0,5
F3	Experiência da equipe em OO	1
F4	Capacidade dos analistas da equipe	0,5
F5	Motivação	1
F6	Estabilidade dos requisitos	2
F7	Estagiários ou funcionários em tempo parcial	-1
F8	Domínio da tecnologia e configuração do ambiente	-1,5

Atribuído	Valor
3	4,5
1	1,5
3	3
1	1,5
5	5
5	10
0	0
-3	-4,5
FatorA	21

FA 0,77
FA = Fator Ambiental

5.0. Pontos de Caso de Uso

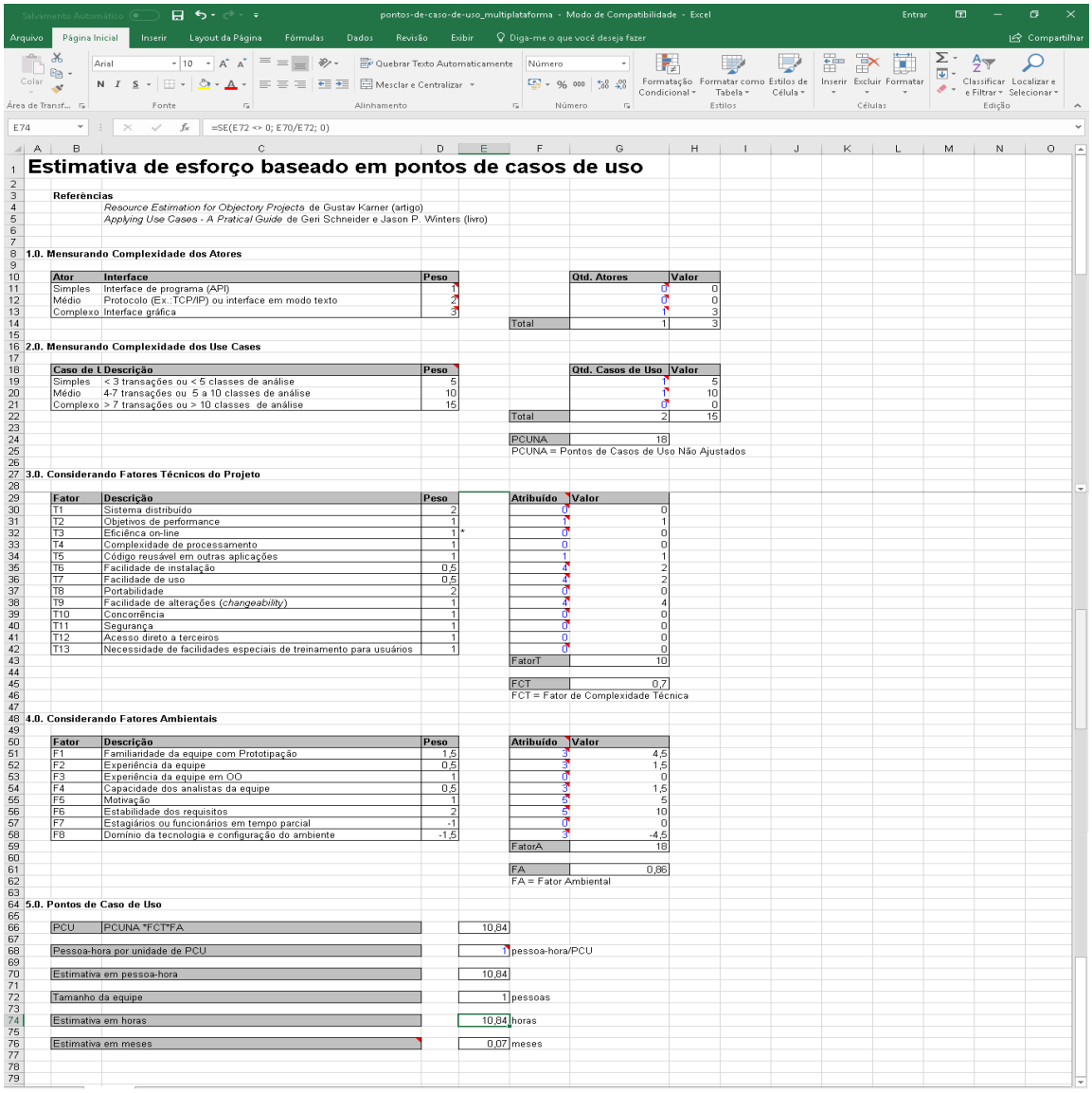
PCU	PCUNA *FCT*FA	9,70	
Pessoa-hora por unidade de PCU	1	pessoa-hora/PCU
Estimativa em pessoa-hora	9,70	
Tamanho da equipe	1	pessoas
Estimativa em horas	9,70	horas
Estimativa em meses	0,06	meses

Fonte: O autor.

Posteriormente, foi estimado o projeto de desenvolvimento de multiplataforma, levando em considerações as mesmas informações. Exceto na parte de fatores ambientais, que é onde a forma nativa se difere da multiplataforma.

Na **Error! Reference source not found.** 15, pode-se ver o resultado final dessa estimativa. Foi obtido um valor de número de pontos de caso de uso equivalente a 10,84. O que levou a um número de horas por pontos de caso de uso de 10,84 horas, levando em consideração que a equipe de desenvolvimento só possui uma pessoa.

Figura 15:PCU Desenvolvimento Multiplataforma – Excel.



Estimativa de esforço baseado em pontos de casos de uso

Referências

Resource Estimation for Objectory Projects de Gustav Karner (artigo)
Applying Use Cases - A Pratical Guide de Geri Schneider e Jason P. Winters (livro)

1.0. Mensurando Complexidade dos Atores

Ator	Interface	Peso	Qtd. Atores	Valor
Simples	Interface de programa (API)	1	0	0
Médio	Protocolo (Ex.: TCP/IP) ou interface em modo texto	2	0	0
Complexo	Interface gráfica	3	1	3
			Total	1	3

2.0. Mensurando Complexidade dos Use Cases

Caso de L	Descrição	Peso	Qtd. Casos de Uso	Valor
Simples	< 3 transações ou < 5 classes de análise	5	1	5
Médio	4-7 transações ou 5 a 10 classes de análise	10	1	10
Complexo	> 7 transações ou > 10 classes de análise	15	0	0
			Total	2	15

PCUNA = 18
PCUNA = Pontos de Casos de Uso Não Ajustados

3.0. Considerando Fatores Técnicos do Projeto

Fator	Descrição	Peso	Atribuído	Valor
T1	Sistema distribuído	2	0	0
T2	Objetivos de performance	1	1	1
T3	Eficiência on-line	1	0	0
T4	Complexidade de processamento	1	0	0
T5	Código reusável em outras aplicações	1	1	1
T6	Facilidade de instalação	0,5	4	2
T7	Facilidade de uso	0,5	4	2
T8	Portabilidade	2	0	0
T9	Facilidade de alterações (*changeability*)	1	4	4
T10	Concorrência	1	0	0
T11	Segurança	1	0	0
T12	Acesso direto a terceiros	1	0	0
T13	Necessidade de facilidades especiais de treinamento para usuários	1	0	0
			FatorT	10

FCT = 0,7
FCT = Fator de Complexidade Técnica

4.0. Considerando Fatores Ambientais

Fator	Descrição	Peso	Atribuído	Valor
F1	Familiaridade da equipe com Prototipação	1,5	3	4,5
F2	Experiência da equipe	0,5	3	1,5
F3	Experiência da equipe em OO	1	0	0
F4	Capacidade dos analistas da equipe	0,5	3	1,5
F5	Motivação	1	5	5
F6	Estabilidade dos requisitos	2	5	10
F7	Estagiários ou funcionários em tempo parcial	-1	0	0
F8	Domínio da tecnologia e configuração do ambiente	-1,5	3	-4,5
			FatorA	18

FA = 0,86
FA = Fator Ambiental

5.0. Pontos de Caso de Uso

PCU	PCUNA *FCT*FA	10,84	
Pessoa-hora por unidade de PCU		1	pessoa-hora/PCU
Estimativa em pessoa-hora		10,84	
Tamanho da equipe		1	pessoas
Estimativa em horas		10,84	horas
Estimativa em meses		0,07	meses

Fonte: O autor.

Foi utilizado uma planilha para o registro da quantidade de horas necessárias para o desenvolvimento em cada plataforma (**Tabela 2**).

Tabela 2: Cômputo das horas necessários para o desenvolvimento do aplicativo.

TOTAL DE HORAS PLANEJADAS TOTAL DE HORAS REALIZADAS

Caso de Uso	**Total de horas estimadas**	**Plataforma**	**Dia**	**Hora início**	**Hora final**	**Total horas (no dia)**	**Quantidade total de horas**
UC001 – Inserir idade da Criança	9,7 horas	Nativa	23/10/2017	14:00	17:30	3 horas e 30 minutos	10 horas e 30 minutos
UC002 – Responder Questionário			26/10/2017	14:00	17:00	3 horas	
UC002 – Responder Questionário			31/10/2017	14:00	17:00	3 horas	
UC003 – Visualizar Resposta			01/11/2017	14:00	16:00	1 horas	
UC UC001 – Inserir idade da Criança	10,84 horas	Multiplataforma	02/11/2017	14:00	18:00	4 horas	13 horas
UC002 – Responder Questionário			03/11/2017	14:00	18:00	4 horas	
UC002 – Responder Questionário			06/11/2017	14:00	17:00	3 horas	
UC003 – Visualizar Resposta			07/11/2017	14:00	16:00	2 horas	

Fonte: O autor.

Pode-se perceber que os valores estimados e os gastos não batem. Devido a algumas dificuldades encontradas no momento de desenvolvimento de ambas plataformas, o tempo gasto para desenvolver excedeu o tempo estimado.

Mesmo existindo experiência nos dois tipos de desenvolvimento, a falta de exercer com frequência trabalhos de desenvolvimentos, na plataforma nativa e multiplataforma, é um fator a ser considerado relevante e por isso um dos motivos do tempo estimado e o gasto não ser os mesmos, ou talvez, mais aproximados.

6.4 PLATAFORMA NATIVA: DESENVOLVIMENTO DO APLICATIVO

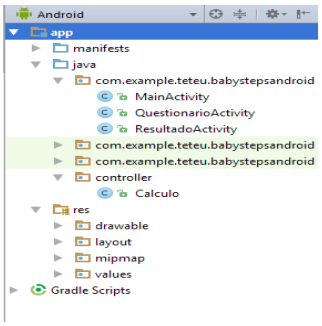
Para o desenvolvimento do aplicativo de forma nativa foram utilizadas seguintes ferramentas:

- *Android Studio*, versão 2.3.3;
- *JDK*, versão 8u151.

Juntamente com essas ferramentas foi utilizada a linguagem de programação *Java*. O aplicativo foi desenvolvimento em um sistema operacional *Windows* 10, 64 *bits*.

Na **Error! Reference source not found.** 16 pode-se observar como é a organização do projeto no ambiente de desenvolvimento especifico para plataforma *Android*. Dentro da pasta *"java"* fica localizado todas as *activitys* (atividades) do projeto. As *activitys* são as atividades que o aplicativo terá. Na pasta *"controller"* (controlador), está localizada a classe *java* cálculo que é responsável por gerar o resultado final. Dentro da pasta *"layout"* estão localizados todos os *layouts* de tela do aplicativo. Já na pasta *"values"* (valores) encontra-se a definição de todas as cores e *strings* (textos) do projeto.

Figura 16:Organização do Projeto - Android Studio.



Fonte: O autor.

A **Error! Reference source not found.** 17, mostra a *MainActivity* do projeto. Essa é a atividade principal do projeto. Ao abrir o aplicativo essa será a primeira atividade que o usuário irá ver. O método *"onCreate"* é onde é inicializado as principais funções da *activity*, como *layouts*, botões e textos.

Figura 17:MainActivity - Android Studio.

```
public class MainActivity extends AppCompatActivity {

    Spinner telainicial;
    Button confirmar;
    WebView view;
    ArrayAdapter adapter;
    int i;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        view = new WebView(this);
        view.setVerticalScrollBarEnabled(false);
        ((LinearLayout)findViewById(R.id.linearlayout_principal)).addView(view);
        view.loadData("<html> <head></head> <body style=text-align:justify;col...","text/html; charset=utf-8", "utf-8");

        telainicial = (Spinner) findViewById(R.id.spinner);
        adapter = ArrayAdapter.createFromResource(this,R.array.spinner_inicial, android.R.layout.simple_spinner_dropdown_item);
        telainicial.setAdapter(adapter);

        confirmar = (Button)findViewById(R.id.btn_confirmar);
        confirmar.setOnClickListener((v) → {
                String opcao = telainicial.getSelectedItem().toString();
                //Toast.makeText(getApplicationContext(), "Item escolhido: " + opcao, Toast.LENGTH_SHORT).show();
                ChamarActivity(opcao);
        });
    }
```

Fonte: O autor.

Na **Error! Reference source not found.** 18, pode-se ver a classe "QuestionarioActivity". Essa classe tem como objetivo possibilitar o usuário de responder todas as perguntas relativas a idade (em meses) inserido na *MainActivity*.

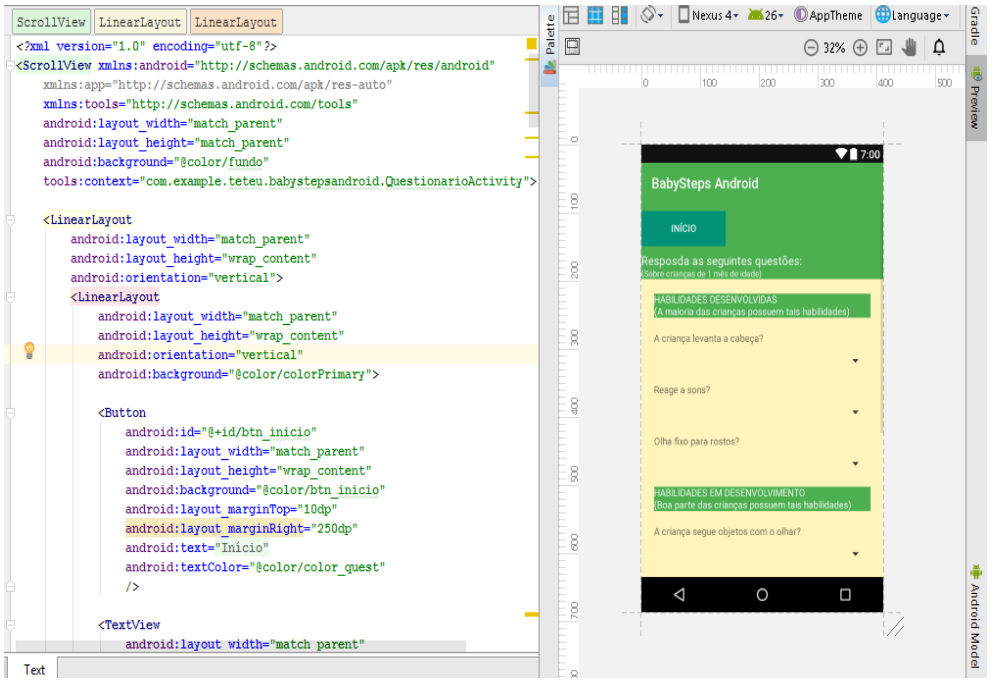
Figura 18:Classe QuestionarioActivity - Android Studio.

```
public class QuestionarioActivity extends AppCompatActivity {

    Button btn1, btn2;
    Spinner spn1, spn2, spn3, spn4, spn5, spn6, spn7;
    ArrayAdapter adapter;
    String a, b, c, d, e, f, g, selec = "Selecione", sim = "Sim", nao = "Não";
    Calculo calc;
    int contador, erroHabDesenv, resultado_final, id;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        Intent it = getIntent();
        Bundle idade = it.getExtras();
        id = idade.getInt("idade");

        if(id == 1)
            setContentView(R.layout.activity_um_mes);
        else if(id == 2)
            setContentView(R.layout.activity_dois_meses);
        else if(id == 3)
            setContentView(R.layout.activity_tres_meses);
        else if(id == 4)
            setContentView(R.layout.activity_quatro_meses);
        else if(id == 5)
            setContentView(R.layout.activity_cinco_meses);
        else if(id == 6)
            setContentView(R.layout.activity_seis_meses);
        else if(id == 7)
            setContentView(R.layout.activity_sete_meses);
        else if(id == 8)
            setContentView(R.layout.activity_oito_meses);
```

Fonte: O autor.

A Figura 19, trata-se do *layout* da tela do questionário de crianças com idade de um mês de idade. No lado esquerdo, onde está organizado a estrutura do projeto, pode-se observar todas os outros *layouts* utilizados na composição do aplicativo.

Figura 19:Layouts - Android Studio.



Fonte: O autor.

Na **Error! Reference source not found.**20, pode-se ver a classe Calculo, dentro do repositório *controller*, que tem como função fazer todos os cálculos para assim obter a resposta de acordo com o questionário respondido pelo usuário.

Figura 20:Classe Calculo - Android Studio.

```
/**...*/

public class Calculo {

    private String a, b, c, d, e, f, g;
    private int contador, erroHabDesenv;

    public int Questions(String q1, String q2, String q3, String q4, String q5, String q6, String q7){

        a = q1;
        b = q2;
        c = q3;
        d = q4;
        e = q5;
        f = q6;
        g = q7;
```

Fonte: O autor.

A **Error! Reference source not found.**, refere-se a classe “ResuladoActivity”. Nessa classe será tratado a exibição do resultado final para o usuário depois de responder todo o questionário.

Figura 21:Classe ResultadoActivity - Android Studio.

```
package com.example.teteu.babystepsandroid;

import ...

public class ResultadoActivity extends AppCompatActivity {

    Button btn;
    int resultado, qtd_erro;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_resultado);

        Intent intent = getIntent();
        Bundle dados = intent.getExtras();
        resultado = dados.getInt("resultado");
        qtd_erro = dados.getInt("erros");

        //Toast.makeText(getApplicationContext(), "valor resultado final: " + resultado, Toast.LENGTH_SHORT).show();

        if(resultado == 1){
            WebView view = new WebView(this);
            view.setVerticalScrollBarEnabled(false);
            ((LinearLayout)findViewById(R.id.linearlayout_resposta)).addView(view);
            view.loadData("<html> <head></head> <body style=text-align:justify;col..." + qtd_erro + "<html> <head></head> <body style=text-a
        }else if (resultado == 2){
            WebView view = new WebView(this);
            view.setVerticalScrollBarEnabled(false);
            ((LinearLayout)findViewById(R.id.linearlayout_resposta)).addView(view);
            view.loadData("<html> <head></head> <body style=text-align:justify;col...","text/html; charset=utf-8", "utf-8");
        }else if(resultado == 3){
            WebView view = new WebView(this);
```

Fonte: O autor.

6.5 PLATAFORMA MULTIPLATAFORMA: DESENVOLVIMENTO DO APLICATIVO

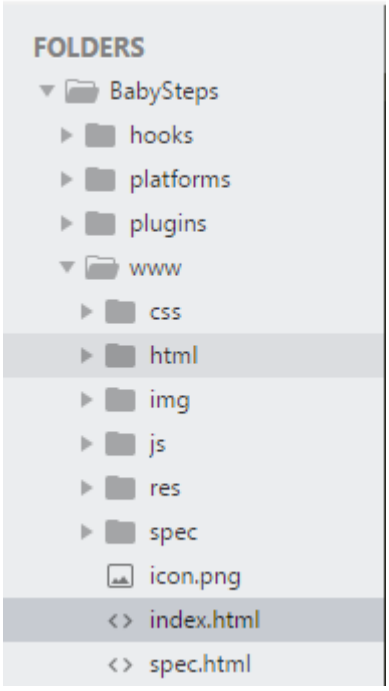
Para o desenvolvimento do aplicativo multiplataforma foram utilizadas as seguintes ferramentas:

- HTML 5;
- CSS3;
- JavaScript;
- PhoneGap, versão 0.4.5;
- Sublime Text, versão 3.0.

O aplicativo foi desenvolvido em sistema operacional *Windows 10*, 64 *bits.*

Na **Error! Reference source not found.**, é possível notar a organização do projeto de acordo com o ambiente PhoneGap. Na pasta *"html"* é onde estão todos os arquivos de questionário de acordo com as idades (em meses) das crianças. O arquivo *"index"* é o arquivo principal, onde a aplicação se inicia. Na pasta *"css"* está a folha de estilo com definições de cores, *layouts* e formatação de texto de todo o projeto. Na pasta *"js"* é onde está o arquivo *JavaScript*.

Figura 22:Organização do Projeto - Sublime Text.



Fonte: O autor.

Na **Error! Reference source not found.**, pode-se ver a o arquivo de extensão *html*, chamado *index*, que é o arquivo principal do aplicativo. Quando iniciado a aplicação será o produto dessa tela que aparecerá.

Figura 23:Index - Sublime Text.

Fonte: O autor

Na **Error! Reference source not found.**, é ilustrado o arquivo referente ao questionário de 1 mês de idade. São esses arquivos que possibilitam que o usuário responda os questionários sobre crescimento e desenvolvimento da criança.

Figura 24:Arquivo de Questionário - Sublime Text.

Fonte: O autor.

A **Error! Reference source not found.**, trata-se do arquivo *JavaScript* que é responsável por destinar o usuário ao questionário de acordo com a idade (em meses) escolhida e também efetuar os cálculos para retornar o resultado final ao usuário. Na **Error! Reference source not found.** pode-se observar a folha de estilo do projeto, onde é especificado toda a parte de design.

Figura 25:JavaScript - Sublime Text.

```
}
window.onscroll = debugtest();
function debugtest(){
    if (document.body.scrollTop > 50 || document.documentElement.scrollTop > 50) {
        document.getElementById("myP").className = "test";
    } else {
        document.getElementById("myP").className = "";
    }

    document.getElementById("teste").innerHTML = "";
}
function getIdade(){
    var v = document.getElementById("selec").value;
    switch(v){
        case "1":
            window.location.href = "html/umMes.html";
            break;
        case "2":
            window.location.href = "html/doisMeses.html";
            break;
        case "3":
            window.location.href = "html/tresMeses.html";
```

Fonte: O autor.

Figura 26:Folha de Estilo - Text Sublime.

```
    display:none;
}

@keyframes fade {
    from { opacity: 1.0; }
    50% { opacity: 0.4; }
    to { opacity: 1.0; }
}

@-webkit-keyframes fade {
    from { opacity: 1.0; }
```

Fonte: O autor.

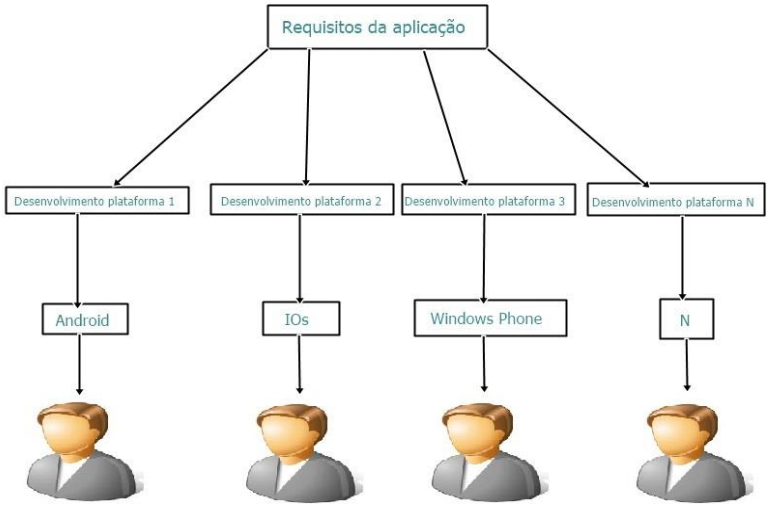
6.6 POTENCIALIDADES E FRAGILIDADES ENCONTRADAS

Uma decisão importante que equipe de desenvolvimento deve tomar é a definição de qual plataforma será utilizada para o desenvolvimento do produto. Um fator que deve ser observado é o fato de não criar o aplicativo para uma determinada plataforma, uma grande parcela do mercado pode estar sendo desconsiderada, o que acaba por reduzir o alcance da aplicação e por consequência os lucros com o aplicativo (CORRAL; JANES; REMENCIUS, 2012).

Cada plataforma possui seu próprio ambiente de desenvolvimento, arquitetura, ferramentas, mercado de distribuição e público alvo (SHAKSHUKI et al, 2013). E cada abordagem de desenvolvimento possui suas características próprias e inerentes à forma como se dá o projeto, desenvolvimento e distribuição de aplicativos (CORRAL; JANES; REMENCIUS, 2012).

A **Error! Reference source not found.** e **Error! Reference source not found.** 28 mostram, respectivamente, como se difere o processo de desenvolvimento utilizando plataformas nativa e multiplataforma.

Figura 27:Desenvolvimento Nativo – PhotoScape.



Fonte: O autor.

Figura 28:Desenvolvimento Multiplataforma – PhotoScape.

Fonte: O autor

A **Tabela 3** mostra as principais potencialidades e fragilidades dos dois tipos de desenvolvimentos abordados nesse estudo.

Tabela 3: Potencialidades e Fragilidades do Desenvolvimento Nativo e Multiplataforma.

	Potencialidades	Fragilidades	Associação com PCU
Nativo	○ Melhor performance; ○ Utiliza todas as capacidades do dispositivo; ○ Menor codificação com maior alcance de produtividade, se comparado ao	○ Não possui facilidade na formatação de texto; ○ Exige conhecimento de linguagens de programação alto nível	○ Facilidade de uso – Fatores Técnicos; ○ Experiência da equipe – Fatores Ambientais.

	desenvolvimento de multiplataforma.		
Multiplataforma	○ É necessário dominar apenas uma linguagem e um ambiente para desenvolver; ○ Recursos de layout facilitados;	○ Menor performance; ○ Não possui acesso a todos as funcionalidades do dispositivo	○ Objetivos de performance – Fatores Técnicos; ○ Objetivos de performance e Portabilidade– Fatores Técnicos e

Fonte: O autor.

Por melhor performance entende-se que o aplicativo nativo tem acesso direto ao sistema operacional do dispositivo e por ser desenvolvido na linguagem nativa do dispositivo são mais velozes que os aplicativos multiplataforma.

A produtividade de código é maior no desenvolvimento nativo, devido a utilização de orientação a objetos que é uma peculiaridade da linguagem de programação utilizada para esse desenvolvimento.

Na Tabela 4, é mostrado o que foi encontrado na literatura como potencialidades e fragilidades no desenvolvimento nativo e multiplataforma.

Tabela 4: Potencialidades e Fragilidades de acordo com a literatura.

	Potencialidades	**Fragilidades**
Nativa	○ Oferece ferramentas de	○ Requer domínio do uso de diversas

	desenvolvimento nativas que exploram o potencial de uma plataforma específica; ○ Oferece aplicativos com uma verdadeira experiência nativa, explorando todos os recursos do dispositivo.	linguagens, sistemas operacionais e ferramentas de desenvolvimento; ○ Limitado a um único mercado de aplicativos.
Multiplataforma	○ Supera a restrição da utilização de diferentes linguagens e ferramentas para cada plataforma; ○ Desenvolva uma vez, implante em qualquer sistema operacional móvel; ○ Maior alcance de mercado de aplicativos.	○ Não permite acesso a alguns recursos do dispositivo móvel

Fonte: CORRAL; JANES; REMENCIUS, 2012.

7. CONSIDERAÇÕES FINAIS

Um escopo foi determinado com base em uma necessidade eminente, conforme o Ministério da Saúde destaca. Esse aplicativo é importante, pois avalia o crescimento e desenvolvimento de crianças de 1 a 24 meses de idade, de acordo com os marcos de desenvolvimentos propostos pelo Ministério da Saúde.

Após a definição das suas funcionalidades e diagrama de caso de uso, através da utilização do método GQM, métricas foram definidas com base nas perguntas que o cliente mais deseja. Utilizou-se o PCU por ser uma medida capaz de atender todas as métricas planejadas no GQM elaborado. Também por ser uma técnica fácil e que utiliza o documento de diagrama de caso de uso.

O processo de estimativa foi feito separadamente para cada tipo de desenvolvimento abordado neste trabalho. Devido à falta de estar praticando com frequência o desenvolvimento nativo e multiplataforma, o tempo gasto para realizar as atividades foi maior que o planejado.

Diferentes potencialidades e fragilidades foram detectadas em ambas as plataformas. Sendo possível, perceber que é preciso analisar qual a necessidade do aplicativo a ser desenvolvido, qual mercado de aplicativos deseja alcançar, quais funcionalidades o aplicativo a ser desenvolvido deve contemplar. Levando sempre em consideração o conhecimento, experiência e tempo de prática da equipe de desenvolvimento.

Durante o desenvolvimento nativo, foi encontrado dificuldade para edição de texto para no aplicativo. Se comparado ao desenvolvimento da multiplataforma, essa função possui maior facilidade de manuseio.

Como trabalhos futuros, propõe-se a realização de novos estudos com base em aplicativos de maior complexidade, com o intuito de obter mais detalhes sobre o esforço para cada tipo de desenvolvimento e aprimorar os resultados obtidos, facilitando cada vez mais o momento de escolha de qual desenvolvimento utilizar.

Esse estudo teve como limitações não contemplar a avaliação do esforço necessário sob outras perspectivas, como exemplo as métricas de qualidade do software, em ambas plataformas.

REFERÊNCIAS BIBLIOGRÁFICAS

ADOBE PHONEGAP BUILD. **Take the pain out of developing mobile apps.** Disponível em < https://build.phonegap.com/> Acesso em 24 de set. de 2016.

ANDROID. **Introdução ao Android.** Disponível em: <http://developer.android.com/guide/index.html> Acesso em 21 de set. de 2016.

APPLE (BRASIL). **Uma aula de inspiração**. Disponível em: <http://www.apple.com/br/> Acesso em 18 de dez. de 2016.

BALBO, Wellington. **Conceitos e Exemplos – Polimorfismo: Programação Orientada a Objetos.** Disponível em < http://www.devmedia.com.br/conceitos-e-exemplos-polimorfismo-programacao-orientada-a-objetos/18701 > Acesso em 14 de nov. de 2016.

BLACKBERRY. **More Smart in the Phone.** Disponível em < http://global.blackberry.com/en/home.html> Acesso em 25 de nov. de 2016.

BELGAMO, Anderson; FABBRI, Sandra. **Um estudo sobre a Influência da Sistematização da Construção de Modelos de Casos de Uso na Contagem dos Pontos de Casos de Uso.** In III Simpósio Brasileiro de Qualidade de Software. Brasília, 2004.

CARVALHO, William. **Conhecendo um pouco do PhoneGap.** Disponível em < http://www.williamcleissondecarvalho.com.br/tag/multiplataforma/ > Acesso em 12 de nov. de 2016.

CORRAL, L.; JANES, A.; REMENCIUS, **T. Potential Advantages and Disadvantages of Multiplatform Development Frameworks - A Vision on Mobile Environments.** Procedia Computer Science , v. 10, p. 1202–1207, jan. 2012. ISSN 1877-0509. Disponível em: http://www.sciencedirect.com/science/article/pii/S1877050912005303> Acesso em 05 de ago. de 2017.

DAL'OSTO, Fábio. **Método para Avaliação de Ambientes de Desenvolvimento de Software Combinando CMM e GQM.** In: Dissertação (mestrado), Universidade Federal do Rio Grande do Sul. Porto Alegre, 2003.

DEITEL, Paul; DEITEL, Harvey; DEITEL Abbey. **Android para programadores uma abordagem baseada em aplicativos.** 2° ed. Porto Alegre: Bookman, 2015.

EL-KASSAS, W. S. et al. **Taxonomy of Cross-Platform Mobile Applications Development Approaches.** Ain Shams Engineering Journal, 2015. ISSN 2090-4479. Disponível em: <http://www.sciencedirect.com/science/article/pii/S2090447915001276> Acesso em 14 de nov. de 2016.

FERNANDES, A. A. **Gerência de software através de métricas: garantindo a qualidade do projeto, processo e produto.** Ed. Atlas, São Paulo, 1995.

FERREIRA, Alessandre. **Frameworks e Padrões de Projeto.** Disponível em: < http://www.devmedia.com.br/frameworks-e-padroes-de-projeto/1111 > Acesso em 05 de dez. de 2016.

GARTNER. **Gartner Says Five of Top 10 Worldwide Mobile Phone Vendors Increased Sales in Second Quarter of 2016**. Disponível em <http://www.gartner.com/newsroom/id/3415117> Acesso em 25 de out. de 2016.

GARAY, Ricardo. **Curso em livro: criação de startups.** Coleção Cursos em Livro, 2014.

HAMANN, Renan**. iOS, Android e Windows Phone: números dos gigantes comparados [infográfico].** Disponível em: <http://www.tecmundo.com.br/sistema-operacional/60596-ios-android-windows-phone-numeros-gigantes-comparados-infografico.htm> Acesso em 25 de dez. de 2016.

IMPACTA. **A importância da programação orientada a objetos na era do mobile.** Disponível em: <https://www.impacta.com.br/blog/2013/10/08/a-importancia-da-programacao-orientada-a-objetos-na-era-do-mobile/ > Acesso em 12 de dez. de 2016.

KOSCIANSKI, André. **Qualidade de Software: aprenda as metodologias e técnicas mais modernas para o desenvolvimento de software.** 2º ed. São Paulo: Novatec Editora, 2007.

LAKATOS, Eva Maria, MARINA, de A. **Fundamentos de metodologia científica.** 5º ed. São Paulo: Atlas, 2003.

LECHETA, Ricardo. **Google Android:** aprenda a criar aplicações para dispositivos móveis com o Android SDK. 2º ed. São Paulo: Novatec Editora, 2010.

LEITE, Mário; JÚNIOR, Nelson Abu Sanra Rahal. **Programação Orientada ao Objeto: uma abordagem didática.** Disponível em: < http://www.ccuec.unicamp.br/revista/infotec/artigos/leite_rahal.html> Acesso em 13 de nov. de 2016.

LECHETA, Ricardo R. **Google Android: aprenda a criar aplicações para dispositivos móveis com o Android SDK.** Ed. Novatec. 2ª ed. São Paulo, 2010.

LOURENÇO, Wellington. **Rumo do Desenvolvimento Mobile.** Disponível em <http://www.devmedia.com.br/rumo-do-desenvolvimento-mobile/24129 > Acesso em 06 de nov. 2016.

LOPES, Sérgio. **Aplicações mobile híbridas com Cordova e PhoneGap.** Série Caelum. São Paulo, 2016.

LOPES, Sérgio. **A web mobile**: programe para um mundo de muitos dispositivos. São Paulo: Casa do Código, 2013.

MEDEIROS, Ernani. **Desenvolvendo Software com UML 2.0.** São Paulo: ed. Makron Books, 2004.

MICROSOFT. **Introducing the new Surface Pro**. Disponível em: < https://www.microsoft.com/en-us/windows/view-all?col=phones> Acesso em 18 de dez. de 2016.

MINISTÉRIO DA SAÚDE. **Saúde da Criança: acompanhamento do crescimento e desenvolvimento infantil**. Série Cadernos de Atenção Básica, n.11, 2002.

MINISTÉRIO DA SAÚDE. **Caderneta de Saúde da Criança: menina.** 2011.

MINISTÉRIO DA SAÚDE. **Caderneta de Saúde da Criança: menino.** 2011.

NABUCO, Thuan Saraiva; FAÇANHA, Agebson Rocha; ARAÚJO, Maria da Conceição Carneiro. **Impactos da diversidade de um ecossistema móvel no desenvolvimento de softwares.** VII CONNEPI (Congresso Norte Nordeste de Pesquisa e Inovação), 2012, Palmas.

PHONEGAP. **PhoneGap Documentation.** Disponível em: < http://docs.phonegap.com/> Acesso em 21 de dez. de 2016.

PRESSMAN, Roger S. **Engenharia de software: uma abordagem profissional.** 8. Ed. AMGH Editora, 2016.

ROUSE, Margaret. **Mobile Operating System.** Disponível em: < http://searchmobilecomputing.techtarget.com/definition/mobile-operating-system > Acesso em 15 de nov. de 2016.

SANTOS, R**. Introdução à Programação Orientada a Objetos Usando Java**. Rio de Janeiro: Campus, 2003.

SHAKSHUKI, E. M. et al. **Component based Framework to Create Mobile Cross-platform Applications.** Procedia Computer Science, v. 19, p. 1004–1011, jan. 2013. ISSN 1877-0509. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1877050913007485> Acesso em 05 de ago. de 2017.

SILVA, Viviane Gomes; GOMES João, Maria. **Dos dispositivos móveis à aprendizagem ubíqua**. REVISTA DE ESTUDIOS E INVESTIGACIÓN EM PSICOLOGÍA Y EDUCACIÓN eISSN: 2386-7418, 2015, Vol. Extr., No. 13. Disponível em < http://revistas.udc.es/index.php/reipe/article/viewFile/610/pdf_398 > Acesso em 04 de dez. de 2016.

SOMERVILLE, Ian. **Engenharia de Software**. 8° ed. São Paulo: Pearson Addison-Wesley, 2007.

TAURION, Cezar**. O desafio de desenvolver apps para o mundo móvel – Parte 02**. Disponível em: < http://imasters.com.br/mobile/o-desafio-de-desenvolver-apps-para-o-mundo-movel-parte 02/?trace=1519021197&source=single > Acesso em 18 de dez. de 2016.

WHITE, James. **Going native (or not): Five questions to ask mobile application developers.** Disponível em: < https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3575060/ > Acesso em 14 de nov. de 2016.

YIN, Robert K. **Estudo de caso: planejamento e métodos.** 2° ed. Porto Alegre: Bookman, 2001.