



UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE CIÊNCIAS EXATAS E NATURAIS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

MÜLLER GABRIEL DA SILVA MIRANDA

**ECOSSISTEMAS DE SOFTWARE MÓVEIS: UM ESTUDO
EXPLORATÓRIO SOBRE ASPECTOS POSITIVOS E
NEGATIVOS NO DESENVOLVIMENTO DE
APLICAÇÕES MÓVEIS**

Dissertação de Mestrado

Prof. Dr. Cleidson R. B. de Souza
Orientador

Belém, Janeiro de 2016



UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE CIÊNCIAS EXATAS E NATURAIS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

MÜLLER GABRIEL DA SILVA MIRANDA

**ECOSSISTEMAS DE SOFTWARE MÓVEIS: ESTUDO
EXPLORATÓRIO SOBRE ASPECTOS POSITIVOS E
NEGATIVOS**

Dissertação submetida à Banca
Examinadora do Programa de
Pós-Graduação em Ciência da
Computação da UFPA para a
obtenção do Grau de Mestre
em Ciência da Computação

Belém, Janeiro de 2016

UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE CIÊNCIAS EXATAS E NATURAIS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

**ECOSSISTEMAS DE SOFTWARE MÓVEIS: ESTUDO
EXPLORATÓRIO SOBRE ASPECTOS POSITIVOS E
NEGATIVOS**

AUTOR: MÜLLER GABRIEL DA SILVA MIRANDA

DISSERTAÇÃO DE MESTRADO SUBMETIDA À AVALIAÇÃO DA BANCA EXAMINADORA A
SER AVALIADA PELO COLEGIADO DO PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA
COMPUTAÇÃO DA UNIVERSIDADE FEDERAL DO PARÁ.

BANCA EXAMINADORA:

Prof. Dr. Cleidson R. B. de Souza
(ORIENTADOR – UFPA – Instituto de Ciências Exatas e Naturais)

Prof. Dr. Fernando Marques FigueiraFilho
(UFRN – Departamento de Informática e Matemática Aplicada)

Profa. Dra. Sabrina Marczak
(PUCRS – Faculdade de Informática)

Dedico este trabalho a meus pais.

AGRADECIMENTOS

Agradeço a Deus, a minha família e a todos os envolvidos no meu processo de formação. Agradeço a meu pai José Gabriel Rodrigues Miranda e minha mãe Simone das Graças da Silva Miranda, que sempre me apontaram os caminhos certos e me conduziram quando foi necessário. Aos meus irmãos que me mantêm firme no desejo de ser um bom exemplo de irmão mais velho. Agradeço aos meus familiares por todo apoio que tive para que eu pudesse me dedicar aos estudos. Agradeço a minha noiva Alessandra Ferreira por estar ao meu lado em todos os momentos, me fortalecendo para que juntos possamos realizar nossos sonhos. Ao meu orientador, Prof. Dr. Cleidson R.B. de Souza pela paciência, incentivo, oportunidades e direções dadas a mim para prosseguimento deste trabalho. Agradeço aos meus amigos Dr. Fábio Bezerra e Carlos Hugo, que sempre me ajudaram e estiveram comigo durante toda essa caminhada. Agradeço a todos os pesquisadores que contribuíram com este trabalho, ao Dr. Fernando Figueira Filho, Andreza Medeiros, Dr. Christoph Treude e Dr. Leif Singer. Agradeço também ao meu caro colega de mestrado Renato Ferreira, que me ajudou em diversas partes deste mestrado, que esteve comigo nos momentos mais complicados desta jornada. Por fim, aos meus demais amigos que são muito importantes e significam muito na minha vida e a quem eu posso sempre recorrer em momentos de dificuldade e alegria.

Müller Miranda

SUMÁRIO

LISTA DE FIGURAS	8
LISTA DE TABELAS	9
RESUMO	10
1. INTRODUÇÃO	11
1.1 Visão Geral	11
1.2 Motivação	13
1.3 Justificativas e Contribuições	16
1.4 Objetivos	17
1.5 Organização do Trabalho	18
2. ECOSISTEMAS DE SOFTWARE	20
2.1 Visão Geral	20
2.2 Classificação	24
2.3 Ecossistema de Software Para Dispositivos Móveis	28
2.4 Benefícios de Participação	30
3. METODOLOGIA	31
3.1 Visão Geral	31
3.2 Pesquisa Qualitativa	32
3.3 Primeira Fase	34
3.3.1 Visão Geral	34
3.3.2 Método de Coleta de Dados: Entrevista	35
3.3.3 Método de Análise de Dados: Teoria Fundamentada em Dados	37
3.3.4 Processo de Coleta e Análise de Dados	40
3.4 Segunda Fase	42
3.4.1 Visão Geral	42
3.4.2 Método de Coleta de Dados: Questionário	42
3.4.3 Processo de Coleta e Análise de Dados	44
4. RESULTADOS	46
4.1 Visão Geral	46
4.2 Primeira Fase (Entrevistas)	46
4.2.1 Aspectos Positivos	47
4.2.2 Aspectos Negativos	52

4.3 Segunda Fase (Questionários).....	57
4.3.1 Estatística Descritiva	58
4.3.2 Classificação dos Respondentes	61
4.3.3 Aspectos Positivos	63
4.3.4 Aspectos Negativos	68
5. DISCUSSÃO	75
5.1 Visão Geral	75
5.2 Aspectos Positivos, Negativos e Ambíguos.....	75
5.3 Relações entre Diferentes Aspectos	76
5.4 Limitações	88
6. CONCLUSÕES	89
6.1 Visão Geral	89
6.2 Contribuições do trabalho	89
6.3 Trabalhos Futuros	90
REFERÊNCIAS	92
APENDICE A - Roteiro das Entrevistas	97
ANEXO A – Questionário	98

LISTA DE FIGURAS

Figura 1 Quantidade em milhões de aparelhos vendidos por fabricante.	15
Figura 2 Crescimento de venda de tablets em relação aos computadores.	16
Figura 3 Porcentagem das plataformas escolhidas como principais.	17
Figura 4 Porcentagem preferencial para desenvolvimento móvel.	21
Figura 5 Relação de competição, mutualismo, comensalismo e neutralismo com o ecossistema de software Linux. Fonte: (YU, RAMASWAMY & BUSH, 2008)	25
Figura 6: Níveis de escopo em ecossistemas de software	26
Figura 7: Uma estrutura para a gestão dos ecossistemas software.	27
Figura 8: Panorama dos atores e seus papéis em ecossistema de software móvel.	29
Figura 9 Exemplo de codificação aberta (BANKS et al, 2000)	39
Figura 10 Exemplo de categorias resultantes da codificação axial (BANKS et al, 2000)	39
Figura 11 Exemplo do uso da ferramenta MaxQDA para codificação	41
Figura 12 Ilustração do processo de coleta e análise dos dados.	42
Figura 13: Intervalo e número de respostas diárias.	45
Figura 14 Correlações entre diferentes os diferentes aspectos identificados.	76
Figura 15 Diferentes experiências de usuários para um mesmo aplicativo baseado no sistema operacional Android, uma das maiores complexidades em desenvolver aplicativos segundo os resultados deste estudo.	79
Figura 16 Mensagem que informa ao usuário que seu dispositivo de hardware é incompatível com o aplicativo.	80
Figura 17 Smartphone Market Share (Fonte: Developer Economics. State of The Developer Nation Q3 2014).....	81
Figura 18 Principais plataformas utilizadas em diferentes continentes.	82

LISTA DE TABELAS

Tabela 1: Diferentes tipos de funções dentro de um ecossistema de software	25
Tabela 2 Aspectos positivos e negativos identificados durante entrevistas.	46
Tabela 3 Localização parcial dos participantes do questionário.	58
Tabela 4 Níveis de experiência em desenvolvimento de software móvel.	59
Tabela 5 Plataformas mais utilizadas de acordo com os respondentes deste estudo.	59
Tabela 6 Tempo de experiência em desenvolvimento de aplicativos móveis.	60
Tabela 7 Número de aplicativos desenvolvidos.	60
Tabela 8 Disponibilidade de aplicativo na loja virtual.	60
Tabela 9 Desenvolvimento de algum aplicativo móvel nos últimos dois anos.	61
Tabela 10 Ferramentas mais utilizadas para o desenvolvimento de aplicativos móveis.	61
Tabela 11 Relação entre tempo de experiência e números de aplicativos desenvolvidos.	62
Tabela 12 Classificação das respostas para a pergunta: <i>What did you like best about developing the app for its platform and why?</i>	63
Tabela 13 Pontos positivos de acordo com os engenheiros de <i>software</i> respondentes. .	64
Tabela 14 Resultados do número de respostas consideradas, desconsideradas ou não respondidas para as principais questões que tratam de aspectos negativos.	69
Tabela 15 Classificação das respostas para a pergunta: <i>What is the most crucial difference between mobile development and desktop or web development?</i>	69
Tabela 16 Pontos negativos de acordo com os engenheiros de software respondentes. .	70
Tabela 17 Visão geral dos aspectos positivos, negativos e ambíguos.	76

RESUMO

Um ecossistema de software consiste em um grupo de atores incluindo engenheiros de software, usuários finais e empresas que funcionam como uma grande unidade interagindo com um mercado comum. Existem vários ecossistemas de software móveis (Android, iOS, Windows Phone) que concorrem entre si para atrair desenvolvedores externos com objetivo de aumentar o número de aplicativos disponíveis. Desta forma, o sucesso de um ecossistema depende da forma como o mesmo gerencia suas relações, por exemplo, o esforço para criação de novas aplicações para o ecossistema. Este esforço de desenvolvimento pode estar associado à complexidade de utilização de ferramentas, baixo engajamento da comunidade, linguagem de programação utilizada para desenvolver as aplicações, ambientes de desenvolvimento, etc. Portanto, o objetivo desta dissertação é investigar os diversos aspectos, positivos e/ou negativos, que engenheiros de software de ecossistemas móveis enfrentam ao longo do processo de criação de aplicativos. Assim, este estudo pretende contribuir com uma visão geral sobre os aspectos que podem influenciar o engenheiro de software a aceitar, rejeitar ou a permanência em um determinado ecossistema de software móvel. Para que este objetivo pudesse ser alcançado, um estudo qualitativo de duas fases foi realizado: na primeira fase foram realizadas entrevistas semiestruturadas com nove (9) engenheiros de software que desenvolviam ativamente para Android (5), iOS (3), ou ambos (1). A primeira fase é chamada de *local*, pois todos os entrevistados eram brasileiros. Dado o número pequeno de entrevistas e a pequena diversidade contextual dos participantes selecionados nesta fase, a segunda fase é chamada de *global*, pois contou com a participação de engenheiros de software de diferentes partes do mundo, pois nesta fase a coleta de dados foi realizada via questionário eletrônico com oitenta e seis respostas (86). Os dados coletados foram analisados em conjunto utilizando métodos da Teoria Fundamentada em Dados. Os resultados mostram a ocorrência de diversos aspectos positivos (Documentação, Engajamento da Comunidade, etc) e negativos (Interface, Recursos, Emuladores, etc), mas também foram revelados aspectos ambíguos (Fragmentação, Popularidade, etc), isto é, aspectos que podem ser tanto positivos quanto negativos. Isto é determinado pelo ecossistema do qual o engenheiro de software faz parte ou até mesmo de sua opinião particular.

Palavras chaves: Ecossistemas de Software, Adoção, Permanência, Aspectos Positivos e Negativos, Estudo Empírico.

1. INTRODUÇÃO

1.1 Visão Geral

O desenvolvimento de software em larga escala é uma atividade complexa, cara, lenta e imprevisível, e este é um dos grandes desafios para as organizações. Então, de forma a apoiar esta indústria, a engenharia de software tem avançado em pesquisas teóricas e práticas para gerenciar a complexidade do desenvolvimento de sistemas de software. Porém, com o crescimento cada vez maior desses sistemas surge a necessidade de novas abordagens para gerenciar esta complexidade (Bosch & Bosch-Sijtsema, 2010). Uma dessas abordagens trata-se dos chamados ecossistemas de software. Bosch & Bosch-Sijtsema (2010) definem um ecossistema de software como “*um conjunto de soluções de software e uma comunidade especialista no domínio de serviços para a comunidade de usuários, que compõe soluções relevantes para as suas necessidades*”. Google Android, Windows Kinect, SAP, Linux e iOS são alguns exemplos de ecossistemas para diferentes domínios. Em outras palavras, um ecossistema de software consiste em um grupo de atores, por exemplo, engenheiros de software, usuários finais e empresas que funcionam como uma grande unidade, interagindo com um mercado comunitário de software e serviços. Estas relações são frequentemente apoiadas por uma plataforma tecnológica ou por um mercado comum e instanciadas através de troca de informações, recursos, etc.

Um ecossistema de software possui basicamente três elementos (Iansiti & Levien, 2004): um centralizador (*keystone*), uma plataforma (e.g., tecnologia ou mercado) e um conjunto de agentes de nicho. Neste contexto, o centralizador (*keystone*) é o proprietário da plataforma e os agentes do nicho podem utilizá-la para gerar valor para ela e para si mesmos. Esta abordagem encoraja um conjunto de desenvolvedores externos, ou seja, que não fazem parte do “elemento” centralizador e sim dos agentes de nicho, a utilizarem uma plataforma de software pertencente a uma *keystone*, e a contribuir para o desenvolvimento de produtos que complementam a plataforma (Campbell *et al.*, 2010).

O desenvolvimento de software, para algumas empresas, não funciona através unidades isoladas que podem fornecer produtos separadamente. No fundo, estas empresas se tornam dependentes de outras no que se refere ao compartilhamento de

componentes de software e infraestrutura, tais como sistemas operacionais, APIs, lojas de componentes e plataformas. Devido à rápida mudança de tecnologia e à alta complexidade no desenvolvimento de software em larga escala, estas empresas procuram pela integração através de alianças para estabelecer redes de influência e interoperabilidade. Estas redes são chamadas ecossistemas de software. Bosch (2010) alega que as organizações que querem se reposicionar para agirem como atores em rede e reduzirem a sua força de trabalho procuram desenvolver cada vez mais produtos ao mercado com ciclos mais reduzidos. Nesse sentido, os agentes do nicho, ou também chamados “desenvolvedores externos”, tornam-se elementos importantes, por contribuírem com o desenvolvimento do ecossistema de software. Assim, abordagens como de lojas virtuais (Apple Store e Play Store) ganham destaque por proporcionarem meios que contribuem com o aumento das funcionalidades para os usuários finais.

Por outro lado, de acordo com o estudo de Barbosa et al (2013), um dos fatores de sucesso entre a empresa fornecedora da plataforma central (*keystone*) e os engenheiros de software externos (agentes de nicho), depende da forma como esta empresa gerencia suas relações. Cataldo & Herbsleb (2010) argumentam que o potencial dos ecossistemas de software é inversamente proporcional à quantidade de barreiras técnicas e organizacionais. Por exemplo, o esforço para criação de novas aplicações para o ecossistema, pode prejudicar no número de aplicativos, soluções para a comunidade de usuários. Este esforço de desenvolvimento pode estar associado à complexidade de utilização das ferramentas para o rápido desenvolvimento de novas aplicações, a indisponibilidade de qualquer tipo de informação relacionada ao processo de desenvolvimento, de documentos de projeto a código fonte, interações entre os envolvidos no ecossistema de software, etc.

O termo ecossistema de software, conforme visto anteriormente, pode ser definido como uma rede informal de unidades ou atores, que influenciam uns aos outros de forma positiva quando se trata de sucesso econômico e outros benefícios. Assim, a saúde de um ecossistema pode ser determinada pela saúde destes atores existentes. Portanto, engenheiros e empresas vendedoras de produtos de software devem considerar seus papéis estratégicos neste novo modelo de negócio emergente.

A hipótese desta dissertação é que a adoção, rejeição ou permanência nos ecossistemas de software móveis, por parte dos engenheiros de software, é influenciada

por uma série de aspectos diferentes. Esta hipótese é baseada em trabalhos anteriores, como por exemplo, o trabalho de Koch e Kerschbaum (2014), que apresenta alguns fatores que influenciam na escolha por parte de engenheiros de software entre as duas plataformas mais comuns Google Android e Apple iOS. Eles relatam que a maior motivação para os desenvolvedores é a diversão, o estímulo intelectual e o aprendizado de novas habilidades. Além disto, Joorabchi et al. (2013) buscam obter uma compreensão das principais práticas e desafios técnicos, enfrentados pelos engenheiros de software para o desenvolvimento de aplicativos para diferentes dispositivos móveis em plataformas nativas. Entretanto, estes trabalhos são limitados pois apresentam um foco maior nos aspectos técnicos. Por exemplo, a facilidade para configurar o ambiente de trabalho, APIs propensas a falhas, fragilidade na segurança da plataforma, etc.

Desta forma, esta dissertação investiga os diversos aspectos que podem ser determinantes para aceitação, rejeição ou permanência de um engenheiro de software em um ecossistema de software móvel específico, não se restringindo a apenas aspectos técnicos. Para alcançar este objetivo, esta dissertação apresenta um estudo de duas fases com metodologias de coleta e análise de dados distintas. Ambas as fases visam compreender os principais aspectos enfrentados pelos engenheiros de software durante o desenvolvimento de aplicativos para dispositivos móveis.

O desenvolvimento de aplicativos para dispositivos móveis se distingue do desenvolvimento de software para desktop ou web. Por exemplo, *smartphones* possuem recursos de hardware tais como localização GPS, acelerômetro, interação do usuário é via touch screen, tempo de vida da bateria do *smartphone*, tamanho de tela para interface do sistema, etc. Por tanto, este trabalho tem como foco ecossistemas móveis por considerar que as dificuldades no desenvolvimento atual para dispositivos móveis difere dos aspectos inerentes ao desenvolvimento de software de modo geral.

1.2 Motivação

O desenvolvimento “tradicional” de software, onde uma única empresa desenvolve um sistema como um único produto de software tem sido substituído pelo desenvolvimento de múltiplos produtos (Bosch & Bosch-Sijtsema, 2010). Este novo paradigma tem sido chamado de ecossistemas de software. Assim, o estudo deste tema surge como um tópico de pesquisa atual e aplicada na área de Engenharia de Software. De acordo com alguns estudos recentes (Barbosa & Alves, 2011; Manikas & Hansen, 2013), o termo

ecossistema de software tem se tornado um tema de pesquisa ativo no meio acadêmico por levar em consideração os modelos abertos de negócio, que geram novos papéis e padrões para colaboração, inovação e proposta de valor.

Por ser um tema relativamente recente, a comunidade acadêmica tem se esforçado em publicar artigos que tratam do assunto. Uma das principais vertentes estudadas pela comunidade acadêmica trata de ecossistemas de software móveis, em que diversas perspectivas de análises são discutidas. Por exemplo, Mansfield-Devine (2012) apresenta algumas fragilidades técnicas quanto a segurança da plataforma Android. Ele também aponta que a fragilidade da segurança não é um fator puramente ocasionado pela popularidade da plataforma, mas sim pela plataforma não possuir rígidos controles de segurança.

Um outro exemplo de estudo sobre as plataformas móveis é o trabalho de Goadrich & Rogers (2011), que apresenta uma análise comparativa de desenvolvimento entre as plataformas móveis Android e iOS. O objetivo destes autores era identificar qual plataforma o setor acadêmico deveria ensinar para alunos de graduação, comparando requisitos de software e hardware de ambas as plataformas, e fatores como interface gráfica, documentação, *Software Development Kit*, etc.

Em (Ingen *et al*, 2011) um método é proposto para analisar a relação entre o lançamento de uma nova versão da plataforma Android com o aumento da participação de usuários em comunidades. Os resultados mostram picos de participação da comunidade coincidindo com o lançamento de novas versões da plataforma. Entretanto, o artigo não identifica quais os assuntos mais frequentes são tratados pela comunidade após o lançamento de uma nova versão.

Esta dissertação tem como motivação analisar a percepção dos engenheiros de software quanto à adoção, rejeição ou permanência de plataformas móveis, por exemplo, Android e iOS. Porém, sabendo que o desenvolvimento de software enfrenta desafios que estão além da alçada puramente técnica este trabalho tem sua contribuição ao estudar não apenas aspectos técnicos mas também aspectos sociais e econômicos, que podem ser determinantes para o sucesso de plataformas. Ele também apresenta um panorama atual das principais dificuldades ou desafios encontrados a partir de relatos quanto ao desenvolvimento de aplicativos móveis, assim contribuindo com a comunidade da área de ecossistemas de software.

As demandas do mercado consumidor vêm aquecendo o mundo da telefonia móvel. Segundo a International Data Corporation¹ (IDC), empresa que realiza pesquisas de mercado, análise e consultoria especializada em tecnologia da informação, telecomunicações e tecnologia de consumo, o mercado de *smartphones* superou a marca de 1 bilhão de aparelhos vendidos em 2013, com a sul-coreana Samsung liderando como a fabricante que vendeu mais unidades, com 31,3% das vendas mundiais. Tal informação corrobora com o gráfico apresentado pelo instituto Gartner, empresa que desenvolve tecnologias relacionadas a introspecção necessária para seus clientes tomarem suas decisões, em parceria com a VisionMobile, empresa que realiza análises sobre a economia aplicativos e modelos de negócios móveis, e a própria IDC, Figura 1.



Figura 1 Quantidade em milhões de aparelhos vendidos por fabricante.

De acordo com a Associação Brasileira da Indústria Elétrica e Eletrônica (Abinee), os *tablets* devem representar quase metade das vendas de itens de informática em 2014, apresentando crescimento maior que em relação a venda de computadores. A expectativa é de que sejam vendidos 10,79 milhões de tablets em 2014, o que representaria uma alta de 36% sobre o volume esperado para este ano, de 7,91 milhões. Assim, os tablets deverão responder por um total de 45% das vendas do setor de informática no próximo ano, ante 36% projetado para 2013. De forma a corroborar tal informação com outros estudos, a Figura 2 ilustra o crescimento de vendas tablets em relação a venda de computadores.

¹ <https://www.idc.com>



Figura 2 Crescimento de venda de tablets em relação aos computadores.

Percebendo este crescimento, engenheiros de software, empresas de tecnologia e comunicação vem sendo atraídas pelas oportunidades de negócios geradas por essa nova categoria de produto. Segundo Cusumano (2010), “*ainda estamos nos estágios iniciais de entender como plataformas e suas relações são realmente importantes para seu sucesso*”. Cusumano também afirma que o vencedor de competições tecnológicas é aquele que tem a melhor estratégia de plataforma e os melhores ecossistemas para apoiá-lo.

1.3 Justificativas e Contribuições

Esta dissertação visa responder algumas questões, tais como “Quais os obstáculos que podem interferir na interação entre engenheiros de software e seus ecossistemas de software?”, ou “Quais aspectos motivam e facilitam a adoção de ecossistemas de software *mobile*?”, ou ainda “Quais os principais desafios enfrentados por engenheiros de software *mobile*?”. Caso essas dificuldades não sejam compreendidas e trabalhadas de forma a excluí-las ou, pelo menos, atenuá-las, a saúde do ecossistema poderá ser afetada de forma negativa. Portanto, informações desse tipo podem ser úteis aos ecossistemas de software, em especial ao *keystone*, quando se trata de estratégia, podendo auxiliar a tomada de decisões e, portanto, influenciar positivamente a saúde do ecossistema de software a qual eles pertencem.

Butle (2011) e Richer (2012), bem como outros autores, revelam que alguns ecossistemas de software móveis são de fato mais adotados que outros, como mostra a Figura 3 que apresenta a porcentagem das plataformas escolhidas como principais. Isto

instiga o desafio da convergência para um corpo de conhecimento, o que impacta na compreensão dos ecossistemas de software móveis.

Dada a grande importância dos ecossistemas de software, faz-se necessário um estudo que contribua para o entendimento deste novo modelo de desenvolvimento de software e de negócio. Isso se faz necessário, principalmente para entender sobre a ótica dos engenheiros de software que desenvolvem para estes ecossistemas, os motivos que fizeram os mesmos a adotar ou a rejeitar ecossistemas específicos.

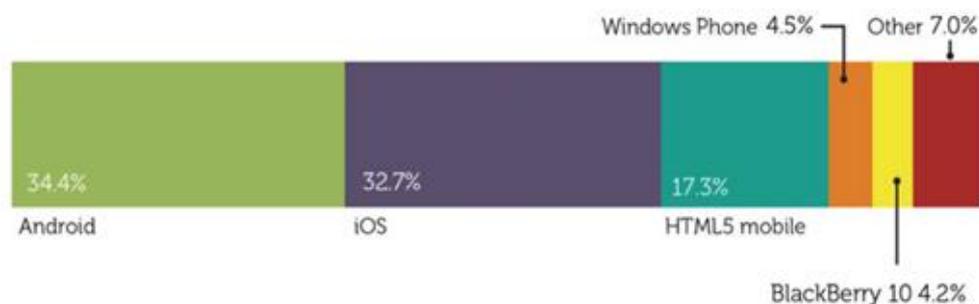


Figura 3 Porcentagem das plataformas escolhidas como principais.

No que diz respeito aos ecossistemas de software móveis, a disponibilidade de aplicações móveis, ou simplesmente *apps*, para uma plataforma móvel influencia diretamente as decisões de compra dos consumidores, o que significa que atrair desenvolvedores de software a um ecossistema móvel pode fazer uma diferença crucial para as empresas que competem por participação de mercado. Assim, as *keystones* (Android, Apple, Mozilla, etc.) fazem um esforço deliberado para atrair novos engenheiros de software, e garantir a qualidade de sua plataforma. Caso contrário a saúde do ecossistema poderá ser afetada de forma negativa, prejudicando, portanto, a rede existente de atores que compõem o ecossistema. Por outro lado, se tais diferenças forem trabalhadas de maneira satisfatória juntamente com as potenciais oportunidades, fazer parte de um ecossistema de software pode resultar em grandes benefícios e diferenciais competitivos que devem ser aproveitados. Como exemplo, a Apple lançou recentemente uma nova linguagem de programação, Swift², que visa facilitar o desenvolvimento de aplicativos para o ecossistema iOS.

1.4 Objetivos

- Objetivo Geral

² <https://developer.apple.com/swift>

Considerando o contexto da pesquisa e o problema apresentado, o objetivo geral desta dissertação é analisar os fatores positivos e negativos que podem ser determinantes e importantes para aceitação, rejeição ou permanência em um ecossistema de software móvel por parte de engenheiros de software. Assim, este trabalho visa compreender os fatores que devem ser levados em consideração para o sucesso de um ecossistema de software móvel partindo das opiniões, observações ou relatos de engenheiros de software.

- **Objetivos Específicos**

1. Identificar aspectos que facilitem (positivos), dificultem (negativos) ou ambíguos para a aceitação, rejeição ou permanência em um ecossistema de software móvel;
2. Verificar o nível de experiência de cada participante desta pesquisa (informantes do questionário e entrevistados);
3. Compreender as razões dos engenheiros de software para a escolha de um ecossistema de software e os desafios que eles enfrentam.
4. Identificar as principais diferenças entre o desenvolvimento para dispositivos móveis em relação ao desenvolvimento de aplicações desktop e web.

1.5 Organização do Trabalho

Além deste capítulo de introdução que apresenta conceitos importantes acerca do tema abordado, as motivações desta proposta de dissertação, a apresentação do problema de pesquisa assim como os objetivos gerais e específicos deste trabalho, esta dissertação está organizada em outros cinco capítulos..

O **Capítulo 2** realiza um levantamento sobre os principais conceitos, definições, estruturas, características e fundamentos teóricos de ecossistemas de software. Além disso, um exemplo de ecossistema de software móvel é apresentado, bem como os benefícios de participação em um ecossistema de software.

O **Capítulo 3** apresenta a metodologia adotada para coleta e análise dos dados, descrevendo os principais conceitos relacionados e como a pesquisa foi conduzida. Especificamente, os conceitos de entrevistas semi-estruturadas e questionário são apresentados pois foram estes os métodos de coleta de dados utilizados.

O **Capítulo 4** apresenta os resultados obtidos com a adoção da metodologia descrita. Ele identifica os principais aspectos positivos e negativos relacionados ao desenvolvimento de aplicativos para ecossistemas de software móveis. Ele também ilustra que alguns aspectos são ambíguos, ou seja, podem ser positivos ou negativos dependendo do contexto.

O **Capítulo 5** apresenta a discussão sobre os resultados encontrados apresentados no Capítulo 4, comparando-os entre si, bem como com outros trabalhos existentes na literatura. Além disso, são apresentadas as limitações desta dissertação.

Finalmente, o **Capítulo 6** conclui esta dissertação, resumindo as principais contribuições da mesma. Ainda, as questões que ainda estão em aberto neste estudo e que podem ser abordadas em trabalhos futuros são apresentadas.

2. ECOSSISTEMAS DE SOFTWARE

2.1 Visão Geral

Nas últimas décadas tem-se testemunhado várias estratégias para gerenciar a complexidade do desenvolvimento de software em grande escala, incluindo *Software Product Lines* (SPLs) e Ecosistemas de Software (Bosch & Bosch-Sijtsema, 2010). SPLs consistem tipicamente de uma plataforma de software compartilhado por um conjunto de produtos. Cada produto pode tipicamente selecionar e configurar componentes na plataforma para os seus próprios fins e estender a plataforma com funcionalidades específicas do produto. Ecosistemas de software ampliam esta estratégia envolvendo um conjunto muito mais amplo de atores, tais como infraestrutura, prestadores de serviços, usuários e desenvolvedores externos. Em vez de depender exclusivamente de interações intraorganizacionais, a abordagem de ecosistemas de software tem uma "perspectiva da comunidade" (Bosch & Bosch-Sijtsema, 2010). Na verdade, o sucesso de um ecossistema de software depende em grande parte da eficácia do ecossistema e suas partes constituintes.

A definição de ecosistemas de software é melhor compreendida ao conhecer as suas origens. Segundo Bosch (2009), a noção de ecosistemas tem origem na Ecologia. De acordo com Odum & Barret (2005), um ecossistema é definido como um sistema composto por um conjunto de comunidades de seres vivos, por exemplo, plantas, animais e bactérias, que vivem e interagem entre si e com os componentes não vivos, ou seja, minerais, pedras, luz solar, ar, etc. Os ecossistemas biológicos oferecem metáforas úteis para ajudar a entender o que são ecosistemas de software. Iansiti & Levien (2004) utilizam o exemplo do jaguar ou, como conhecemos no Brasil, a onça-pintada. Este animal é conhecido por se alimentar de quase 85 espécies de animais, ajudando no controle de populações dentro do ecossistema em que ele está inserido, mas o jaguar é apenas uma pequena parte que forma esse ecossistema. Essa metáfora pode ser utilizada para entender o papel de uma governante de plataforma (*keystone*), como a Google, dona do Android, que apesar de ser uma das gigantes do mercado tecnológico (ver Figura 4), é apenas uma parte do ecossistema de software móvel (Vision Mobile, 2013).

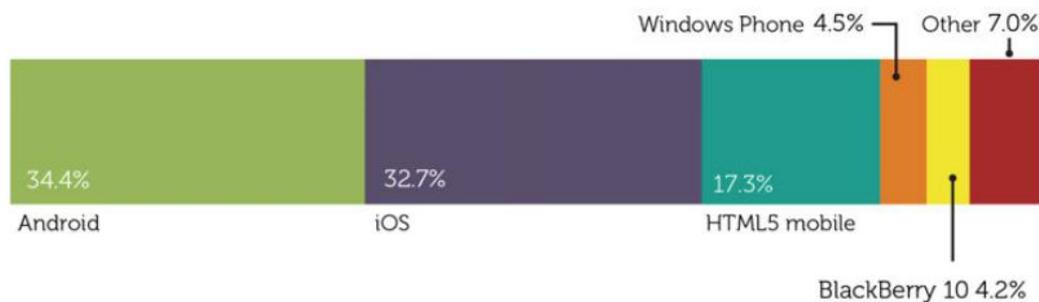


Figura 4 Porcentagem preferencial para desenvolvimento móvel, 2013.

Dhungana et al. (2010) ainda ressaltam que existem outras metáforas que podem ser estabelecidas entre ecossistemas de software e ecossistemas biológicos, como: ambos os ecossistemas tem uma quantidade finita de recursos e a colaboração e a competição são elementos fundamentais para que os mesmos prosperem. A Figura 4 representa a porcentagem preferencial para desenvolvimento móvel. 34,4% dos desenvolvedores preferem a plataforma Android, logo atrás, a plataforma iOS com 32,7%.

Embora tais metáforas sejam aceitáveis, as comparações podem ser feitas até um determinado ponto, visto que elas se distanciam da discussão acerca de ecossistemas de software, pois, ecossistemas biológicos são estudados para determinar que fatores externos influenciam em suas dinâmicas, enquanto que ecossistemas de software tem como principais estudos quais fatores de sucesso e crescimento influenciam em sua dinâmica. E o maior fator que diferenciam ambos, é que em ecossistemas de software existe o fator humano, e por se tratar de seres humanos, os seus participantes podem conscientemente decidir se vão continuar naquele ecossistema ou se vão sair, e podem até mesmo, destruí-lo. Assim, a noção de ecossistemas humanos pode ser mais apropriada para entendermos o conceito de ecossistema de software.

Bosch (2009) define um ecossistema humano como um conjunto de atores e suas conexões e atividades, bem como as transações realizadas a partir dessas conexões, considerando fatores físicos e não físicos. Pode-se distingui-lo entre ecossistemas comerciais ou de negócio e sociais. Em um ecossistema comercial, os atores são os negócios, fornecedores e clientes; os fatores são os bens e serviços; e as transações incluem não somente as financeiras, mas também o compartilhamento de conhecimento e de informação, investigação, contatos pré e pós-vendas etc. Já os ecossistemas sociais consistem em um conjunto de usuários e suas conexões sociais.

Assim, Bosch (2009) explica que a expressão ecossistema de software foi inspirada pelos conceitos de ecossistemas dito biológicos e de ecossistemas comerciais, e que geralmente é complementada com um ecossistema social onde os usuários podem contribuir ativa ou passivamente com informações, ideias, soluções e feedback. Um exemplo de ecossistema social trata-se do Facebook (Jansen, Brinkkemper & Cusumano, 2013).

Ecossistemas comerciais são definidos por Morre (1996) como uma comunidade econômica apoiada por uma fundação: organizações e indivíduos interagindo como organismos do mundo de negócios. Os membros deste mundo, incluem: fornecedores, produtores, competidores, entre outros. Esses papéis são semelhantes aos papéis presentes em ecossistemas de software. Morre (1996) também explica que durante o tempo esses membros aprendem a evoluir em conjunto e tendem a alinhar-se em torno de uma ou mais empresas centrais, que no caso de ecossistemas de software são chamadas são chamadas de *keystone* (Iansiti & Levien, 2004) ou *platform leadership* (Cusumano & Gawer, 2002).

Iansiti & Levien (2004), apresentam basicamente três elementos para tais ecossistemas: um centralizador (*keystone*), uma plataforma e um conjunto de agentes de nicho. Assim, o centralizador é o proprietário da plataforma (e.g. Google, Microsoft, Apple, etc.), e os agente do nicho (desenvolvedores externos, fabricantes de hardwares ou de parte de software, consumidores finais, etc) que podem utilizá-la para gerar valor para ela e para si mesmos. Portanto, ecossistema de software trata-se de um termo utilizado para descrever a crescente colaboração entre unidades, por exemplo, empresas, engenheiros de software que desenvolvem soluções de sistemas de informação, clientes, entidades governamentais, entre outros que fazem parte deste contexto. Além disso, são governados e dirigidos por uma ou mais partes que lucram quando o ecossistema prospera e, geralmente esses governantes controlam a tecnologia que serve como base para o ecossistema. Assim, ecossistemas de software têm se tornado um campo de pesquisa ativo dentro da área de Engenharia de Software pelo aparecimento de novos modelos de negócio que levam à redefinição dos papéis tradicionais e padrões de colaboração, inovação e proposição de valor, pois comunidades de rede complexas de organizações ou atores são criadas (Hanssen & Dyba, 2012), isto é um salto radical em como a Engenharia de Software está sendo feita, de maneira que o desenvolvimento de software está se tornando um processo aberto em um complexo ambiente distribuído.

Por ser um tema relativamente recente, observa-se na literatura a existência de diversas definições para ecossistemas de software, uma vez que diferentes grupos de pesquisa têm investigado o tema de modo independente. Além das definições já apresentadas, abaixo algumas definições para ecossistemas de software são listadas. Tais definições podem ser encontradas na revisão sistemática desenvolvida por Manikas & Hansen (2012).

MESSERCHMITT & SZYPERSKI (2003)

“Tradicionalmente, um ecossistema de software refere-se a uma coleção de produtos de software que tem algum determinado grau de relacionamento simbiótico.” Esta é a mais antiga definição para ecossistemas de software.

JANSEN, FINKELSTEIN & BRINKKEMPER (2009)

“Um ecossistema de software é em um conjunto de atores funcionando como uma unidade que interage com um mercado distribuído entre software e serviços, juntamente com as relações entre estas entidades. Estas relações são frequentemente apoiadas por uma plataforma tecnológica ou por um mercado comum e realizadas pela troca de informação, recursos e artefatos.”

Por esta definição ser mais abrangente e genérica, é uma das mais citadas, assim como a definição apresentada por Bosch & Bosch-Sijtsema (2010).

BOSCH (2009)

“Um ecossistemas de software consiste em um conjunto de soluções de software que possibilitam, apoiam e automatizam as atividades e transações realizadas por atores em um ecossistema social ou de negócios e por organizações que proveêm estas soluções.”

BOSCH & BOSCH-SIJTSEMA (2010)

“Um ecossistemas de software consiste em uma plataforma de software, um conjunto de desenvolvedores internos e externos e uma comunidade de especialistas no domínio a serviço de uma comunidade de usuários que compõem elementos de solução relevantes

para satisfazer as suas necessidades.”

LUNGU et al. (2008)

“Um ecossistemas de software é uma coleção de projetos de software que são desenvolvidos e evoluem juntos em um mesmo ambiente.”

2.2 Classificação

De acordo com Yu, Ramaswamy & Bush (2008), participantes de um ecossistema de software podem assumir seis tipos de relacionamento: mutualismo, parasitismo, comensalismo, amensalismo, competição e neutralismo. Yu explica que nem todas essas relações são aparentes. As mais comuns são mutualismo, comensalismo e competição. A definição destes conceitos vem de ecossistemas dito biológicos.

O **mutualismo** é uma relação harmônica na qual há vantagens recíprocas para as espécies que se relacionam. Por exemplo, entre o boi e as bactérias e protozoários existentes em seu estomago ocorre uma relação mutualística, pois as bactérias fabricam uma enzima que os bois não possuem para digerir os alimentos ricos em celulose que ingerem. Em troca o boi fornece as bactérias e aos protozoários, abrigo e nutrição. O parasitismo é uma relação desarmônica entre dois organismos na qual um deles (o parasita) é beneficiado e, o outro (hospedeiro) é prejudicado. Por exemplo, Muito parecido com um parasita biológico, os “adware” e “spyware”, causam danos em seus hospedeiros. Já o **comensalismo** é uma relação harmônica entre organismos que se caracteriza por ser benéfica para um, não causando prejuízo a outra. Por exemplo, “máquinas virtuais” que nada mais são que software de ambiente computacional em que um sistema operacional ou programa pode ser instalado e executado sem prejudicar o hospedeiro. O **amensalismo** é uma relação desarmônica na qual uma espécie inibidora (*keystone*) prejudica a outra espécie amensal (organizações externas) sem com isso se beneficiar. A **competição**, onde ambos os indivíduos envolvidos são prejudicados. Finalmente, o **neutralismo**, onde nenhum dos organismos é prejudicada. A Figura 5 ilustra algumas possíveis relações simbióticas entre o Linux com outros sistemas de software.

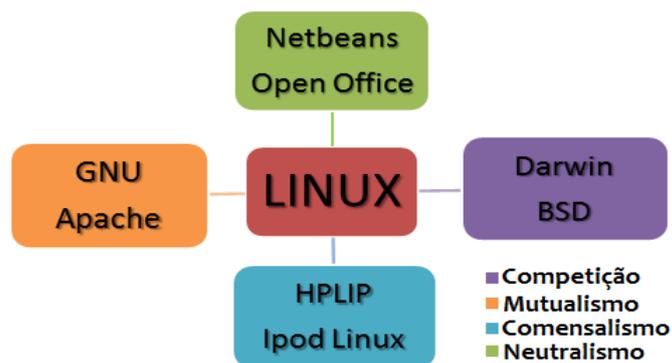


Figura 5 Relação de competição, mutualismo, comensalismo e neutralismo com o ecossistema de software Linux. Fonte: (Yu, Ramaswamy & Bush, 2008)

A partir da Figura 5, nota-se que algumas relações são fáceis de caracterizar enquanto outras são difíceis. Por exemplo, a relação entre Linux e BSD pode parecer ser competitiva, porque ambos são sistemas operacionais e competem pelo mesmo mercado consumidor. Mas também podemos caracterizar uma relação mutualista porque podem se ajudar em trocas de código fonte, componentes ou mesmo arquitetura, permitindo maior flexibilidade no desenvolvimento. Isto ilustra que o conceito de ecossistema pode referir-se a uma grande variedade de configurações. No entanto, todos eles envolvem dois conceitos fundamentais: uma rede de organizações e um interesse comum na tecnologia de software central (Hanssen, 2012). Estas organizações podem ter diferentes relações com a tecnologia de software central, e por este motivo, diferentes funções no ecossistema, e de acordo com Hanssen (2012), há três tipos principais de funções (ver Tabela 1).

Tabela 1: Diferentes tipos de funções dentro de um ecossistema de software

FUNÇÃO	DESCRIÇÃO
<i>Keystone</i>	Organização que conduz o desenvolvimento da tecnologia de software central.
Usuários Finais	Papel chave para a tecnologia de software central, pois representa quem precisa dela para realizar seu negócio.
Organizações Externas	Utilizam a tecnologia de software central como base para produzir soluções ou serviços relacionados.

De acordo com Jansen et al. (2009a) os ecossistemas de software podem ter três níveis de escopo (Figura 6). Estes níveis estão relacionados basicamente ao tamanho a ser analisado em um ecossistema de software. O primeiro nível (1), chamado de **organizacional** trata das relações diretas no ecossistema de software. Já o segundo nível (2), chamado de **visão interna de um ecossistema de software** apresenta não somente as relações diretas, mas também as indiretas, ou seja, o alvo é no conjunto de participantes. No terceiro nível (3) de escopo, **chamado de visão externa de um ecossistema de software**, o alvo está na complexa rede em torno do ecossistema.

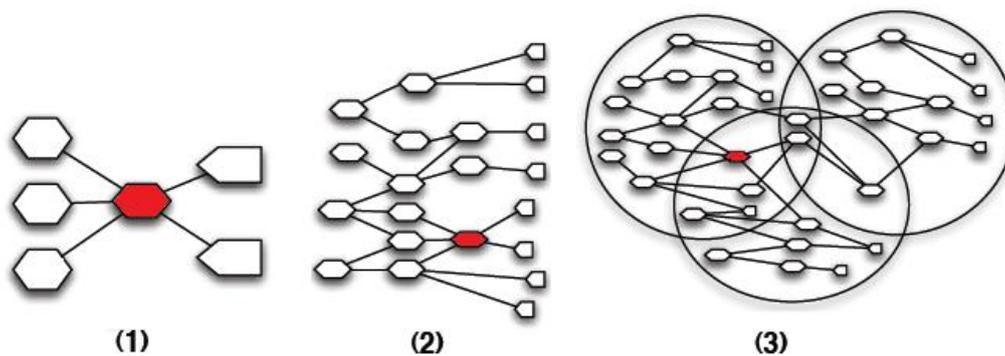


Figura 6: Níveis de escopo em ecossistemas de software

Conforme visto na Figura 6, percebe-se que a **visão externa de um ecossistema de software**, permite um ator estar em diferentes ecossistemas com diferentes papéis. No exemplo do ecossistemas de software Linux, a ferramenta Netbeans também faz parte de outros ecossistemas de software, tal como o ecossistema Windows.

Segundo Campbell & Ahmeds (2010), pode-se analisar ecossistemas de software em três dimensões (negócio, técnica e social). A dimensão de negócio envolve três pilares: visão do ambiente, inovação e planejamento estratégico. Esta dimensão refere-se ao conhecimento sobre o mercado, definição do portfólio de produtos do ecossistema de software, estratégias de licenças e de vendas, etc. A dimensão técnica envolve três pilares: engenharia da plataforma, arquitetura e gerência de requisitos. Esta dimensão trata da plataforma e da infraestrutura tecnológica em que o ecossistema está inserido. A dimensão social envolve três pilares: utilidade, promoção e ganho de conhecimento. Esta dimensão define como atores se relacionam para atingir seus objetivos (lucro).

Todas as três dimensões estão intimamente integradas através de processos de engenharia de software. A motivação organizacional permite o estabelecimento de uma

plataforma em que o ecossistema de software pode ser iniciado. Indivíduos ou grupos ganham benefícios intrínsecos quando contribuem na forma de desenvolvimento de novos produtos utilizando a plataforma padrão.

Dhungana et al. (2010) apresentam uma estrutura para a gestão dos ecossistemas software (Figura 7). Esta estrutura mostra que é possível monitorar e melhorar vários parâmetros, por exemplo, competição, *market*, saúde do ecossistema, a partir de diferentes perspectivas, no caso, técnica, negócio e de participação.

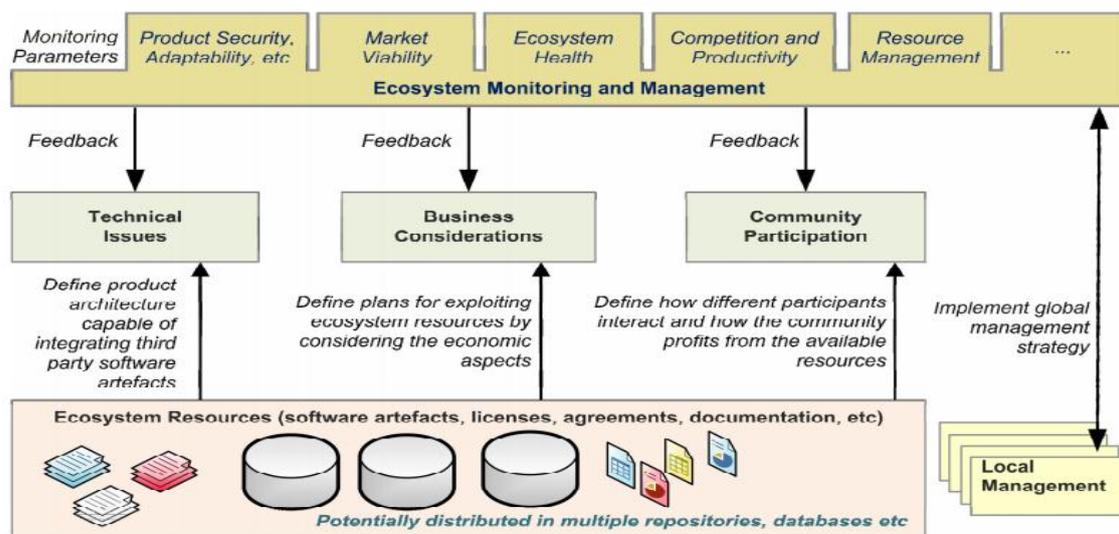


Figura 7: Uma estrutura para a gestão dos ecossistemas software.

O ecossistema de software Facebook é apresentado como exemplo ilustrativo seguindo uma análise nessas três perspectivas. Quanto a perspectiva técnica, o Facebook promove uma arquitetura que permite fáceis integrações com software de terceiros. Quanto a perspectiva de negócio, organizações externas podem ganhar dinheiro com a monetização de aplicações no Facebook, por outro lado, o Facebook lucra com publicidade. Quanto à participação, tanto a comunidade de desenvolvedores do Facebook e usuários ganham. A comunidade de desenvolvedores são beneficiados pela alta quantidade de usuários e os usuários pela variedade de aplicações desenvolvidas.

Andrade et al. (2011) descreve o ciclo de vida de um ecossistema de software a partir dos estágios evolucionários apresentados por Moore (1993). Este ciclo de vida apontado contém quatro estágios: nascimento, expansão, liderança e auto renovação. O **nascimento** refere-se a uma proposta inicial fornecida pela *keystone* para atender a

alguma demanda de mercado. O foco é identificar as necessidades do cliente e definir o que agrega valor. Com isto, organizações externas irão aderir e contribuir com o ecossistemas. A **expansão** é uma etapa aonde ocorrem disputas entre ecossistemas rivais, gerando uma complexa rede em torno do ecossistema. Quando o ecossistema ganha estabilidade o estágio de **liderança** é alcançado, embora ainda possa ocorrer disputas entre os participantes. Finalmente, a **auto renovação**, neste estágio o ecossistema se encontra robusto para lidar com mudanças.

2.3 Ecossistema de Software Para Dispositivos Móveis

Utilizando as definições de Bosch & Bosch-Sijtsema (2010) e Jansen, Finkelstein & Brinkkemper (2009), que foram previamente apresentadas, um ecossistema de software é formado por um conjunto de atores, que desempenham seus papéis dentro do mesmo. Na Figura 8 é ilustrado um exemplo de ecossistema de software, nesse caso, um ecossistema de software móvel.

Na Figura 8 tem-se a Apple que é uma das plataformas mais populares de todos os tempo Seminer (2009), e alguns atores que atuam em seu ecossistema móvel, o iOS. De acordo com Jansen & Cusumano (2012), podemos classificar este ecossistema como: “O ecossistema de software Apple iOS é baseado em uma plataforma de software e coordenado por uma entidade privada com uma extensão comercial mercado em que os participantes podem enviar extensões depois de fazer um pagamento mento”. Além disso, conforme visto em Iansiti & Levien (2004), existem três elementos chave em um ecossistema: um centralizador, uma plataforma, e um conjunto de agentes de nicho. Assim, conforme apresentado na Figura 8, temos que: o centralizador nada mais é que a *keystone*, neste caso, a Apple. O iOS como a plataforma central e os agente do nicho, que neste contexto são: as operadoras de telefonia móvel, os desenvolvedores de aplicações, os usuários e os fabricantes de hardware. A seguir é a apresentado em mais maiores detalhes cada um desses atores.

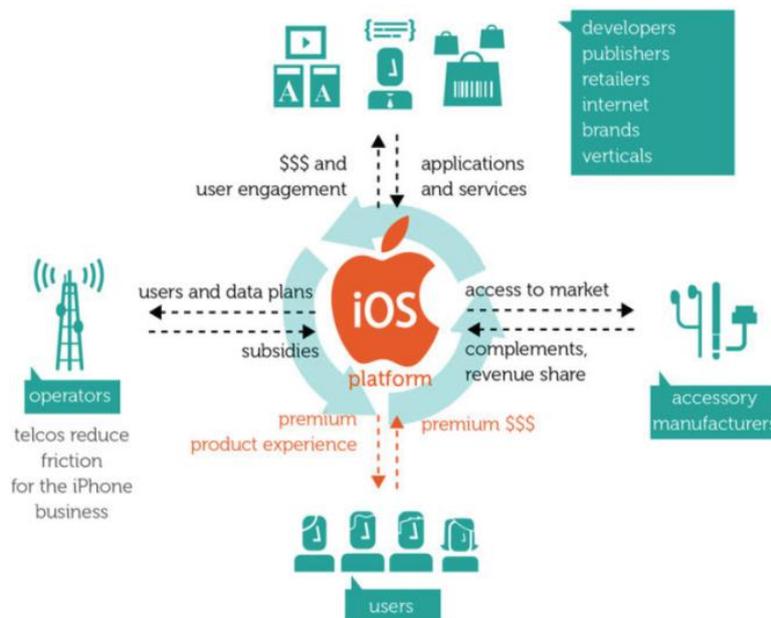


Figura 8: Panorama dos atores e seus papéis em ecossistema de software móvel (iOS).

A Apple é a “dona da plataforma” (centralizador ou *keystone*). Ela desenvolve a plataforma iOS, que é um sistema operacional para *smartphones*. É em torno desta plataforma que o ecossistema de software gira em torno (Hanssen & Dyba, 2012). É por meio desta plataforma que os demais atores se relacionam. Por exemplo, os usuários finais da plataforma, que são as pessoas que utilizam o *smartphone* com o iOS instalado, e também fazem uso das aplicações que são desenvolvidas pelo desenvolvedores de software. Estas aplicações estão presentes em uma loja virtual, chamada App Store. Os usuários pagam para usufruir desta plataforma, seja apenas comprando o *smartphone*, seja comprando produtos de software na loja virtual.

Os desenvolvedores de software criam aplicações/serviços para a plataforma. Eles desenvolvem produtos de software que são utilizados pelos usuários finais da plataforma, e podem disponibilizá-los de forma gratuita ou paga. O seu lucro é oriundo de usuários que pagam pelas aplicações que são desenvolvidas por eles. Dado que a Apple não fabrica nenhum dispositivos de hardware, ela age em mutualismo com os fabricantes de hardware, pois tanto a Apple como os fornecedores podem obter benefícios mútuos pelo crescimento do ecossistema.

Os fabricantes de hardware fornecem componentes que serão acoplados ao *smartphone* da Apple, e vão desde produtores de chips que são utilizados diretamente na fabricação e montagem dos *smartphones*, até fabricantes de acessórios, como fones

de ouvido e capas protetoras. Eles prosperam quando o *smartphone*, que possui o sistema operacional iOS, é utilizado por muitos usuários, pois aumenta a demanda por seus produtos.

A operadora de telefonia móvel é um ator exclusivo deste tipo de ecossistema de software. Uma vez que esses *smartphones* são evoluções dos telefones celulares comuns, ou *dumbphones*, o usuário geralmente precisa estar atrelado a uma operadora de telefonia móvel. Além disso, caso queira utilizar todo o potencial de seu *smartphone*, ele precisa de um plano de pacote de dados, para o acesso à internet. Então as operadoras tem uma nova fonte de receita, planos de dados específicos para *smartphones*, assim os usuários podem acessar a loja virtual de aplicações, podem comprar os produtos de software e, com isso, gerar receita não só para a si e *keystone*, mas também para os desenvolvedores. É possível notar que todos os atores estão interligados de alguma forma, caracterizando o cenário de ecossistema de software.

No caso do ecossistema de software Apple, a plataforma iOS, compete diretamente com o ecossistema de software móvel da Google, o Android. Os concorrentes são responsáveis por retirar valor da *keystone* e suas parceiras. Bosch (2010) explica que as organizações abrem suas plataforma no intuito da redução da força de trabalho, pois os agentes do nicho passam a ter um papel importante pela participação no desenvolvimento de soluções para a plataforma. Cataldo & Herbsleb (2010), explicam também que o sucesso de um ecossistema de software, de forma geral, está relacionado a dois fatores: transparência e modularidade. A transparência refere-se a disponibilidade de qualquer tipo de informação relacionada ao processo de desenvolvimento, no caso da plataforma iOS, a documentação da API, exemplos de código fontes, tutoriais, tarefas de desenvolvimento, defeitos e interações entre os envolvidos no ecossistema de software. A modularidade consiste na aplicação do princípio tradicional da Engenharia de Software a partir do qual um sistema complexo é decomposto em partes gerenciáveis, minimizando o acoplamento técnico entre os elementos constituintes. A estratégia para iniciar um ecossistema de software depende de como o *keystone* convence organizações externas a adotar sua tecnologia.

2.4 Benefícios de Participação

O conceito de ecossistema de software está ganhando popularidade entre grandes organizações e com isto, um conjunto de desenvolvedores externos passam a utilizar

uma plataforma de software pertencente a um grupo de organizações e também a contribuir para o desenvolvimento de vários produtos (Campbell & Ahmed, 2010). O sucesso de um ecossistema de software depende da forma como o mesmo administra suas relações com as organizações parceiras (Hoch, 2000).

A participação e esforço mutuo dos atores que fazem parte do ecossistema de software (*keystone*, e os agentes do nicho) pode ser trazer benefícios para estes atores, ao estender, melhorar a plataforma, acelerar o processo de criação de software, reduzir custos de desenvolvimento pela participação de desenvolvedores externos, aumentar o número de envolvidos, fazendo com que o ecossistema agregue novos envolvidos, externos à organização e conseqüentemente seu crescimento. A proximidade entre a *keystone* e os a gentes do nicho se torna fundamental para o sucesso de um empreendimento de software. Mas para isto, a *keystone* deve criar abordagens transparentes e modular (Cataldo & Herbsleb, 2010).

3. METODOLOGIA

3.1 Visão Geral

Este capítulo tem por objetivo explicar detalhadamente a metodologia, e conseqüentemente, as atividades realizadas nesta dissertação. Ele visa descrever o tipo de pesquisa realizado, o tempo previsto, a equipe de pesquisadores, as técnicas de coleta e análise de dados utilizadas entre outros aspectos. Enfim, tudo que foi utilizado durante o trabalho.

O trabalho desenvolvido nesta dissertação caracteriza-se como exploratório, do tipo levantamento de dados e foi escolhida uma abordagem prevalentemente qualitativa, pois a mesma baseia-se nas observações das pessoas que realizam a atividade que é o objeto de estudo (Strauss & Corbin, 2008). Para que o objetivo de pesquisa proposto nesta dissertação pudesse ser alcançado (ver seção 1.4 que descreve o objetivo), um estudo prevalentemente qualitativo de duas fases foi realizado. Na *primeira* fase foram realizadas entrevistas com nove engenheiros de software que desenvolviam software ativamente para Android (5), iOS (3), ou ambos (1). Esta fase é classificada como local, pois todos os entrevistados eram brasileiros e residiam na mesma cidade (Belém, PA). Nesta fase, optou-se por utilizar entrevistas semiestruturadas (Seaman, 2008) para a

coleta de dados, porque com elas é possível uma maior liberdade no momento da entrevista, sem a necessidade de seguir um roteiro fechado, podendo assim explorar aspectos, da forma que os entrevistadores acharem melhor (Pinsonneault & Kraemer, 1993). Na *segunda* fase, dado o número pequeno de entrevistas e a pequena diversidade de participantes selecionados na fase anterior (brasileiros e que desenvolvia aplicativos para o ecossistema de software Android ou iOS), optou-se por um questionário eletrônico aberto obtendo-se assim a participação de engenheiros de software de diferentes partes do mundo que desenvolviam aplicações para diferentes ecossistemas de software (Android, iOS, Windows Phone, etc). Para ser mais específico, foram recebidas oitenta e seis respostas. Em ambas as fases, os dados foram analisados utilizando métodos da teoria fundamentada em dados (Strauss & Corbin, 2008). Na primeira fase, as entrevistas foram transcritas e analisadas, enquanto que na segunda fase apenas as perguntas *abertas* foram analisadas. A pesquisa desta dissertação enfatizou os aspectos qualitativos das respostas dos informantes.

Os resultados oriundos da análise dos dados tanto da primeira fase quanto os da segunda fase são apresentados separadamente no Capítulo 4. Entretanto, no Capítulo 5 (Discussão) os dados foram integrados para que pudessem ser discutidos.

3.2 Pesquisa Qualitativa

A pesquisa qualitativa é uma abordagem utilizada quando se deseja compreender detalhadamente os motivos pelos quais um indivíduo realiza determinada ação. A pesquisa qualitativa costuma ser usada para trazer a tona a "lógica de compra", que é a explicação do porque um indivíduo compra um produto ou serviço ou produto específico (Moresi, 2003). Essa é a base para identificar segmentos de mercado reais ou grupos de pessoas que compram pelos mesmos motivos e razões (Moresi, 2003). De forma semelhante, este estudo visa analisar os fatores positivos e negativos que podem ser determinantes e importantes para aceitação, rejeição ou permanência em um ecossistema de software móvel específico.

Diferentemente dos estudos quantitativos, a pesquisa qualitativa não segue a rigor um plano previamente estabelecido, mas sim é direcionada ao longo do seu desenvolvimento. Além disso, não é seu objetivo enumerar ou medir eventos, seu foco de interesse é amplo e parte de uma perspectiva diferenciada da adotada pelos métodos

quantitativos. Nas pesquisas qualitativas, é comum que o pesquisador procure entender os fenômenos, a partir da visão que os participantes da situação estudada tem, e com isto, situe sua interpretação destes fenômenos. Assim, a pesquisa qualitativa também é chamada de interpretivista, pois o pesquisador, precisa *interpretar* os dados que ele coletou (Neves, 1996).

Seaman (2008) diz que a principal vantagem de se realizar uma pesquisa qualitativa reside no fato de que métodos qualitativos obrigam os pesquisadores a mergulhar nos fatos observados, sem abstrai-los. Tendo isso em mente, os resultados tendem a ser mais ricos em detalhes e informativos.

Segundo Strauss & Corbin (2008), há três componentes principais nesse tipo de pesquisa: dados, procedimentos e relatórios. Os *dados* são oriundos de entrevistas, gravações de áudio ou vídeo, documentos, etc. Portanto, o dado é a matéria prima que será estudada através de *procedimentos*; estes que possuem o objetivo de reduzir esses dados de forma a relacionar conceitos. Finalmente, os *relatórios* são produzidos com o objetivo de apresentar os resultados encontrados.

Embora haja muita diversidade entre os trabalhos denominados qualitativos, algumas características identificam os estudos deste tipo de pesquisa: (1) *O ambiente natural como fonte direta de dados e o pesquisador como instrumento fundamental*. Os estudos qualitativos têm como preocupação fundamental o estudo e a análise do mundo empírico em seu ambiente natural (Godoy, 1995). Além disso, a abordagem qualitativa valoriza o contato direto e prolongado do pesquisador com o ambiente e a situação que está sendo estudada. (2) *O caráter descritivo*. A descrição é fundamental para o processo de obtenção dos dados e disseminação dos resultados. Os pesquisadores qualitativos estão preocupados com o processo e não simplesmente com os resultados ou produto. (3) *O significado que as pessoas dão às coisas e à sua vida como preocupação do investigador*. Pesquisas qualitativas tentam compreender os fenômenos que estão sendo estudados a partir da perspectiva dos participantes. Finalmente, (4) *o enfoque indutivo*. Como pesquisas qualitativas não partem de hipóteses estabelecidas *a priori*, elas não se preocupam em buscar dados ou evidências que corroboram ou neguem tais suposições (Godoy, 1995). Elas partem de questões ou focos de interesse amplos, que vão se tornando mais diretos e específicos no transcorrer da investigação .

3.3 Primeira Fase

3.3.1 Visão Geral

A primeira fase de coleta de dados contou com a participação de nove engenheiros de software de diferentes partes do Brasil. Estes engenheiros foram entrevistados e escolhidos por conveniência, ou seja, foram selecionados a partir de contatos pessoais e indicações por parte dos primeiros entrevistados, esta abordagem é conhecida como Snowballing, onde alguns entrevistados potenciais são contatados e perguntados se eles sabem de alguém com o mesmo perfil (engenheiros de software com habilidades para desenvolvimento de aplicativos móveis). Todos os engenheiros entrevistados desenvolviam aplicativos para dispositivos de diferentes ecossistemas de software móveis. A seguir, estas nove pessoas são brevemente descritas com fins apenas de caracterização dos desenvolvedores. Todos os dados são anônimos, por questões de confidencialidade. Os dados foram obtidos através de entrevistas semiestruturadas conduzidas no período de Junho de 2013 à Agosto de 2014.

Entrevistado 1 – 24 anos de idade, graduando em Licenciatura da Computação, com 6 meses de experiência em desenvolvimento para Android.

Entrevistado 2 – 24 anos de idade, mestre em Engenharia Elétrica com ênfase em Computação Aplicada, com 2 anos de experiência em desenvolvimento para iOS.

Entrevistado 3 – 25 anos de idade, Bacharel em Ciência da Computação e Engenharia de Controle de Automação, com 2 anos e meio de experiência em desenvolvimento iOS.

Entrevistado 4 – 22 anos de idade, Bacharel em Ciência da Computação, com 2 anos e meio de experiência em desenvolvimento iOS.

Entrevistado 5 – 26 anos de idade, Mestre em Ciência da Computação, com 2 anos de experiência em desenvolvimento para Android e 4 meses em desenvolvimento para iOS.

Entrevistado 6 – 33 anos de idade, Bacharel em Engenharia da Computação, com 2 anos de experiência em desenvolvimento para Android.

Entrevistado 7 – 27 anos de idade, Bacharel em Ciência da Computação com MBA em Engenharia da Computação Avançada, com experiência em desenvolvimento para o ecossistema de software Android e iOS.

Entrevistado 8 – 24 anos de idade, Bacharel em Engenharia da Computação, com 2 anos de experiência em desenvolvimento para o ecossistema de software Android.

Entrevistado 9 – 29 anos de idade, formado em Filosofia, com 1 ano e meio de experiência em desenvolvimento Android.

É importante mencionar que nesta fase de coleta de dados, as entrevistas foram realizadas com o apoio do mestrando Renato Pina Ferreira da UFPA, que também contribuiu para a transcrição e análises das mesmas.

3.3.2 Método de Coleta de Dados: Entrevista

Um dos métodos de coleta de dados mais comuns em pesquisas qualitativas é a condução de entrevistas, abordagem que foi usada nesta primeira fase.

A coleta de dados, como o próprio nome diz, trata de coletar documentos ou provas, buscar por informações sobre um determinado tema ou conjunto de temas correlacionados e agrupá-los de forma a facilitar uma análise posterior. A coleta de dados facilita uma posterior análise dos fatos ou fenômenos que estão ocorrendo em uma organização, sendo o ponto de partida para a elaboração e execução de um trabalho de pesquisa. Para tanto, um ou mais métodos para coleta de dados deve ser utilizado.

Por se tratar de um estudo exploratório, isto é, o pesquisador desenvolve conceitos, ideias e entendimentos a partir de padrões encontrados nos dados, este primeiro estudo optou por entrevistas semiestruturadas (Singer et al, 2008) como método de coleta de dados, pois através delas é possível identificar como os entrevistados exercem suas atividades na prática.

As entrevistas são largamente usadas em pesquisa de diferentes segmentos. A entrevista é um método flexível para a obtenção de informações sobre um assunto. Este método requer um planejamento prévio e a habilidade do entrevistador para seguir um roteiro de questionário com possibilidade de introduzir variações que se fizerem necessárias durante sua aplicação, dependendo do tipo de entrevista (não estruturada ou semi-estruturada).

As entrevistas vão de estruturadas a não estruturadas. O que varia entre elas é o nível de controle por parte do pesquisador (Strauss & Corbin, 2008). No caso de entrevistas estruturadas, o pesquisador tem o foco bem estruturado e definido com

perguntas bem específicas. As entrevistas não-estruturadas são exatamente o oposto disso, ou seja, elas são amplas e procuram coletar o máximo de informações possíveis sobre o tópico de estudo.

De acordo com Marconi e Lakatos (2003), existem vantagens e desvantagens na coleta de dados através de entrevistas. A entrevista é um método muito eficiente para a coleta de dados em profundidade acerca do comportamento humano, os dados obtidos são suscetíveis de classificação e de quantificação. Em geral, a aplicação de uma entrevista pode fornecer uma quantidade de informações muito maior do que o questionário, em contrapartida, a entrevista requer um tempo maior do que o de respostas a questionários. Além disso, a falta de compreensão do entrevistado quanto ao significado das perguntas; o fornecimento de respostas falsas; a influência da presença do pesquisador; etc. De forma a mitigar tais desvantagens, é importante conhecer previamente os entrevistados e se manter o mais imparcial possível, procurando não emitir opiniões, insegurança ou admiração. Assim, promove-se mais segurança e confiabilidade ao entrevistado, portanto a própria entrevista. Além disso, o prévio planejamento da entrevista ajuda manter focados no objetivo. Um dos requisitos para aplicação desta técnica é que o entrevistador possua as habilidades para conduzir o processo (Santos, 2005).

Nesta primeira fase de coleta de dados optou-se pelo método de entrevistas semiestruturadas, pois elas são a interseção entre os dois modelos citados anteriormente, possuem uma variedade de perguntas, específicas ou abertas, permitindo assim que o pesquisador não só colete informações sobre pontos definidos anteriormente, mas também tenha a oportunidade de levantar informações inesperadas, ou seja, informações que não poderiam ser coletadas seguindo um roteiro restrito. A entrevista semiestruturada, no fundo, se parece muito com uma conversação, mas se diferencia por ter o objetivo básico de coletar dados. Além disso, uma entrevista semiestruturada considera a existência de um guia (Apêndice A) que consiste de uma lista de perguntas para garantir que todos os tópicos de interesse serão abordados.

A construção do guia de entrevista foi elaborado com base na Teoria da Difusão da Inovação de Rogers (2003). Em 1962 Everett Rogers publicou sua tese de doutorado: A Teoria da Difusão de Inovação (TDI) onde ele havia feito uma extensa revisão de estudos relativos à difusão de inovações, notadamente de inovações tecnológicas no

meio agrícola e que também incluía estudos em diferentes áreas, dentre elas marketing. A TDI busca explicar como ou por que uma inovação é aceita ou rejeitada por indivíduos. Segundo Rogers (1995): “*a difusão é o processo pelo qual uma inovação é comunicada através de certos canais de comunicação ao longo do tempo para os membros de um sistema social*”. Portanto, quatro elementos dão base a TDI de Rogers: inovação, canais de comunicação, tempo e sistema social. Apesar disto, esta dissertação não pretende abordar tal teoria, e sim elencar os aspectos positivos e negativos, ou seja, as questões que foram mapeadas para os aspectos da inovação descritas acima foram analisadas e classificadas como positivas ou negativas.

3.3.3 Método de Análise de Dados: Teoria Fundamentada em Dados

Em uma pesquisa qualitativa a análise dos dados é dita interpretativa, devendo-se utilizar portanto métodos de análise qualitativos (Strauss & Corbin, 2008). A análise qualitativa é o processo não-matemático de interpretação, que tem por objetivo descobrir conceitos e relações nos dados brutos e de organizar tais conceitos e relações em um esquema explanatório teórico que explique os dados coletados, uma vez que, neste tipo de análise não se tem inicialmente uma hipótese sobre os dados a ser testada (Strauss & Corbin, 2008).

Geralmente, pesquisa qualitativas geram uma grande quantidade de dados que devem ser analisados. Para isso, o conjunto de dados deve ser reduzido para um formato compreensível, o que é tradicionalmente feito através do processo de codificação (Singer *et al.*, 2008). Tal processo consiste de usar os objetivos da pesquisa como um guia e assim desenvolver um esquema para categorizar os dados. Depois dessa categorização, os dados podem ser analisados para prover uma caracterização baseada nos esquemas de codificação dos pesquisadores (Singer *et al.*, 2008).

A seguir aspectos da Teoria Fundamentada em Dados serão brevemente descritos. Vale ressaltar que esta dissertação apenas utilizou algumas técnicas desta abordagem para análise dos dados, mais precisamente a codificação aberta e axial que serão explicadas posteriormente.

O método da Teoria Fundamentada em Dados insere-se na investigação qualitativa, surgindo no contexto dos estudos sociológicos de Glaser e Strauss (Glaser & Strauss, 1967). Ela é uma metodologia que envolve a descoberta de uma teoria derivada

dos dados coletados (Strauss & Corbin, 2008). Seu objetivo é justamente gerar uma teoria que explica os dados coletados, ou seja, gerar uma “teoria [efetivamente] fundamentada em dados”.

Para que se desenvolva a teoria é necessário uma correta interpretação dos dados, que servirá de decisão sobre quais dados serão trabalhados. Portanto, a primeira etapa da teoria fundamentada em dados trata de analisar os dados cuidadosamente, linha por linha (codificação). Em outras palavras, a base da teoria fundamentada em dados está nas codificações, que consistem de atribuir um código, conceito ou categoria a uma parte dos dados. A ideia é utiliza-los para abstrair um evento, objeto, ação ou interação de interesse do pesquisador, assim, uma teoria é um conjunto de categorias bem desenvolvidas que são sistematicamente relacionadas entre si através de declarações de relação para formar uma estrutura teórica que explique o fenômeno estudado. A codificação refere-se, então, aos procedimentos utilizados para rotular e analisar os dados coletados e envolve comparações constantes entre fenômenos, casos e conceitos, as quais conduzem ao desenvolvimento de teorias por meio da abstração e relações entre os elementos. As abstrações (rótulos para os dados) são criadas para minimizar o número de elementos que precisam ser considerados pelo pesquisador. Para atingir este objetivo, a teoria fundamentada em dados possui três etapas principais: codificação aberta, codificação axial e codificação seletiva (Strauss & Corbin, 2008).

A codificação aberta, segundo Strauss & Corbin (1990), é o processo pelos quais os conceitos são identificados e desenvolvidos em relação a suas propriedades e dimensões. Os dados são micro analisados, o que envolve examinar, comparar, conceituar e categorizar os dados que serão sumarizados em códigos.

A Figura 9 a seguir foi retirada do trabalho de Banks et al (2000) que analisa do uso de cartões postais como exemplo de codificação aberta. Nela observam-se as categorias numeradas e exemplos de trechos retirados dos cartões postais relacionados a estas categorias, seguidos do número de vezes que cada categoria foi utilizada.

2. Description of especially enjoyable experience; great fun	“He continually pushes himself to the limit (and us) to have as much fun as he possibly can, a true ‘ fun hog. ’” [#50]	63
3. Reference to extraordinary work effort or work demands	“Was a working dynamo — couldn’t keep enough jobs ahead of him!” [#120]	3
4. Relates connections to celebrity or drops name of celebrity	“Alas, Patty did not get mentioned in humorist Dave Barry’s newspaper column this year ...” [#98]	2
5. Mention of experience in cultural arts — fine arts museums, opera, ballet, symphony, etc.	“She is also showing quite an interest in ballet.” [#51]	19

Figura 9 Exemplo de codificação aberta (Banks et al, 2000)

A segunda etapa consiste na codificação axial, que aprimora e diferencia os códigos resultantes da codificação aberta, ou seja, os códigos são quebrados em conceitos. Enquanto os códigos representam fenômenos, os conceitos são criados para responder perguntas sobre esses fenômenos. Os dados, portanto, são agrupados através das conexões entre códigos e conceitos. A Figura 10 a seguir também foi retirada do trabalho de Banks et al, (2000) para servir como exemplo de codificação axial. Nela observam-se as subcategorias da categoria “*Positive Experience/Adventure*” resultantes da codificação axial e o número de trechos (*tokens*) anexados a cada uma.

- 1 ***POSITIVE EXPERIENCE/ADVENTURE*** (278 tokens):
 - 2) fun experience/great time/I-we enjoyed (63)
 - 4) celebrity experience/name drop (2)
 - 5) cultural activities (19)
 - 8) exciting adventure (34)
 - 9) you’d be amazed at what I experienced (3)
 - 61) writer/we traveled (91)
 - 62) family member traveled (37)
 - 78) vacation (31)

Figura 10 Exemplo de categorias resultantes da codificação axial (Banks et al, 2000)

Durante o processo de codificação, o pesquisador pode alterar entre codificação aberta e axial, pois tanto os códigos e conceitos devem ser constantemente verificados pelos dados que os compõem e que podem ser reorganizadas.

Finalmente, na terceira etapa, a codificação seletiva tem por objetivo integrar e refinar conceitos em categorias, um nível mais abstrato. A tarefa é elaborar **as categorias mais importantes** para que a teoria seja descrita através delas.

3.3.4 Processo de Coleta e Análise de Dados

Como descrito anteriormente, entrevistas semiestruturadas foram utilizadas como método de coleta de dados neste estudo. Quanto à análise, a mesma foi realizada por meio de métodos da Teoria Fundamentada em Dados.

Nove entrevistas com desenvolvedores de ecossistemas de software móveis foram realizadas sendo os informantes oito homens e uma mulher. Entre eles 5 desenvolvem ativamente para o Android, 3 para iOS e 1 desenvolve para ambas as plataformas. Dois dos entrevistados eram estudantes, um de graduação em licenciatura em computação e outro em pós-graduação em ciência da computação, ambos com menos de 2 anos de experiência no desenvolvimento de aplicativos para ecossistemas de software móveis. Os outros eram profissionais com mais de dois anos de experiência em desenvolvimento de aplicativos para dispositivos móveis. Os entrevistados tinham idades entre 19 a 29 anos. Todos são brasileiros e residentes no Brasil.

As entrevistas foram realizadas no período de agosto a dezembro de 2013, através de uma amostra não probabilística, isto é, os entrevistados foram escolhidos por questões de localização, facilidade de acesso e disponibilidade para participar na pesquisa. Após a realização da coleta de dados (entrevistas), foram feitas as transcrições das mesmas a fim de serem analisadas utilizando técnicas da Teoria Fundamentada em Dados (Strauss e Corbin, 2008).

Conforme dito anteriormente, as entrevistas seguiram um roteiro de perguntas abertas, mas que foram previamente mapeadas para aspectos sócio computacionais, de acordo com a teoria de inovação de Rogers (2003). Por exemplo, Rogers apresenta o conceito de *complexidade* e este conceito poderia ser mapeado a partir da (in)existência de ferramentas e exemplos de aplicações que permitem o desenvolvimento de aplicações para o ecossistema, ou ainda, a *testabilidade* poderia ser mapeada a partir da dificuldade em se obter as ferramentas necessárias para o desenvolvimento de aplicações, etc. Apesar da utilização de Rogers no guia de entrevistas, esta dissertação abordará os resultados de forma diferente ao que é proposto pela teoria de inovação de Rogers. A forma como esta dissertação apresentará os resultados será dada pelo entendimento de aspectos positivos e negativos, isto é, tudo que for identificado como complexo para criação, adoção ou permanência no ecossistema de software móvel será listado como aspectos negativos, enquanto que os aspectos que facilitam o

desenvolvimento sernao considerados aspectos positivos. As entrevistas foram transcritas e analisadas usando a ferramenta MaxQDA (Figura 11).

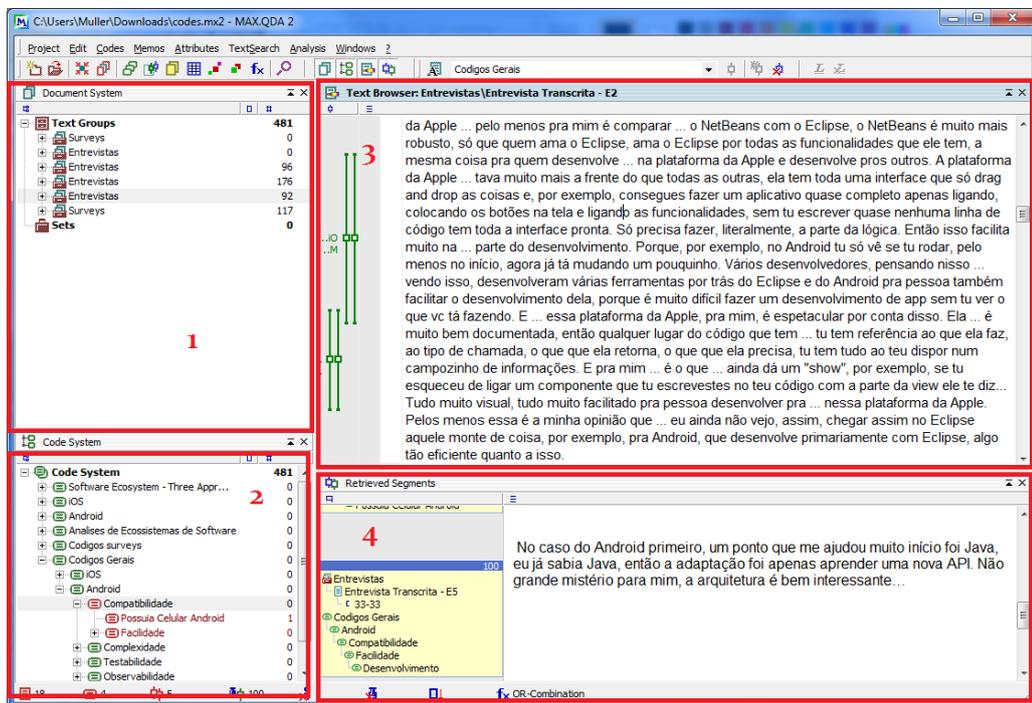


Figura 11 Exemplo do uso da ferramenta MaxQDA para codificação

A Figura 11 traz uma imagem da ferramenta MaxQDA2 contendo um trecho de uma entrevista já categorizado após as etapas de codificação aberta e axial. A região marcada com o número 3 apresenta um trecho da transcrição de uma entrevista. Ao lado do texto é possível ver colchetes que indicam que aquele pedaço de texto foi anexado a uma categoria (na ferramenta utilizada, é possível ver o nome da categoria posicionando o mouse sobre o colchete). A região 1 indica os arquivos (entrevistas) que são codificados. A região 2 mostra o conjunto de categorias criada durante a etapa de codificação aberta. Por fim, a região 4 serve para visualizar os trechos anexados a uma determinada categoria.

A etapa de codificação aberta foi realizada separando os dados em categorias e subcategorias. Em seguida, a etapa de codificação axial foi iniciada para compreender o que os dados significavam.

3.4 Segunda Fase

3.4.1 Visão Geral

A Figura 12 ilustra resumidamente o processo de pesquisa durante a segunda fase da pesquisa o que inclui a criação de um questionário e atividades de coleta e análise dos dados. Após o desenvolvimento do questionário, foi construído um programa de mineração de dados para identificação de projetos de plataformas móveis, e assim, conseguir os contatos dos responsáveis por manterem o projeto no GitHub. Após a identificação e listagem dos contatos, o questionário foi enviado para os respectivos contatos por email. Finalmente, pôde-se analisar os resultados após *feedback* das pessoas que foram contatadas.

A utilização de um questionário deve-se ao fato de que este método consegue atingir uma grande amostra para análises. Ou seja, ele possibilita atingir um grande número de pessoas, mesmo que estejam dispersas numa área geográfica muito extensa, já que o questionário pode ser enviado pelo correio ou por email (Gil, 1999).



Figura 12 Ilustração do processo de coleta e análise dos dados.

É importante mencionar que este processo foi realizado em parceria com o aluno Renato Pina Ferreira (Universidade Federal do Pará), a aluna Andreza Medeiros, o Dr. Fernando Figueira Filho, ambos da Universidade Federal do Rio Grande do Norte, e o Dr. Leif Singer da Universidade de Victória, Canadá.

Este estudo deu origem ao artigo intitulado “Os Aspectos Sociais dos Ecosystemas de Software” publicado no XII Simpósio Brasileiro de Sistemas Colaborativos (SBSC 2015) e que recebeu o prêmio de *Best-Paper* do simpósio.

3.4.2 Método de Coleta de Dados: Questionário

O questionário é um dos procedimentos mais utilizados para obter informações em pesquisas empíricas. Ele pode ser descrito como a obtenção de dados ou informações sobre características, ações ou opiniões de um determinado grupo de pessoas; grupo este

indicado como representante de uma população alvo. O questionário é um conjunto de perguntas que uma pessoa, desta população alvo, lê e responde sem a presença de um entrevistador (Pinsoneault & Kraemer, 1993).

Dado o número pequeno de entrevistas e a pequena diversidade de participantes selecionados na primeira fase para esta fase foi elaborado um questionário (APÊNDICE) com 15 questões, divididas em três seções. A primeira seção visava a obtenção de informações sobre o participante (experiência de trabalho no desenvolvimento de aplicativos para dispositivos móveis, a(s) plataforma(s) usada(s) etc). A segunda seção coletou informações sobre a aplicação mais importante desenvolvida pelo participante, se o aplicativo estava disponível na loja de aplicativos e quais ferramentas foram utilizadas para desenvolvê-lo. Finalmente, a terceira seção perguntava sobre as percepções dos desenvolvedores em relação ao desenvolvimento de aplicativos móveis, incluindo quais foram os aspectos positivos e negativos que tiveram de enfrentar durante o desenvolvimento de seus aplicativos, uma comparação entre desenvolvimento móvel e Web/desktop, e o problema mais difícil enfrentado na opinião dos informantes. Todas as questões desta última seção eram questões abertas.

As vantagens do uso do método do questionário em relação às entrevistas são as seguintes: utiliza-se menos pessoas para ser executado e proporciona economia de custo, tempo, viagens, com obtenção de uma amostra maior e não sofre influência do entrevistador (Marconi & Lakatos, 1996; Mattar, 1996). Dentre as desvantagens podem ser citadas: baixo índice de devolução de respostas; grande quantidade de perguntas em branco; dificuldade de conferir a confiabilidade das respostas; demora na devolução do questionário e a impossibilidade do respondente tirar dúvidas sobre as questões o que pode levar a respostas equivocadas (Marconi & Lakatos, 1996; Mattar, 1996).

Pinsonneault & Kraemer (1993) classificam a pesquisa com questionários quanto ao seu propósito em: explanatória, exploratória e descritiva. A explanatória tem por objetivo testar uma teoria e relações causais, ou seja, ela estabelece a existência de relações causais, mas também questiona porque a relação existe. A exploratória tem por objetivo familiarizar-se com o tópico ou identificar conceitos iniciais sobre um tópico, dar ênfase na determinação de quais conceitos devem ser medidos e como devem ser medidos. A descritiva busca identificar quais situações, eventos, atitudes ou opiniões estão manifestas em uma população; descreve a distribuição de algum fenômeno na

população ou entre subgrupos da população ou ainda faz uma comparação entre estas distribuições. O tipo de questionário utilizado neste estudo é classificado como exploratório.

3.4.3 Processo de Coleta e Análise de Dados

O processo de coleta dos dados deste estudo, trata-se de uma submissão/envio de um questionário para um grupo de engenheiros de software que desenvolve aplicativos para ecossistemas de software móveis. Especificamente, os engenheiros de são cadastrados no serviço GitHub. O Github é um serviço de hospedagem web compartilhado para projetos que usam o controle de versionamento Git, ou seja, ele possibilita a pessoas ou empresas, guardar, compartilhar ou divulgar projetos de software pela Web. Por isto, ele é também considerado uma rede social. O GitHub permite a troca de mensagens e atualizações de perfil de usuários.

O Github possui, entre outras APIs, uma API chamada de Search API³ que tem como principal objetivo ajudar a encontrar itens (arquivos) específicos em projetos. Esta API funciona de forma similar a uma pesquisa no Google, em que é suficiente informar uma palavra para obter resultados de busca que se relacionam com a palavra indicada.

Após a análise da documentação das várias plataformas e projetos presentes no GitHub, análise esta desenvolvida pelo Dr. Fernando Figueira Filho e Andreza Medeiros, observou-se que projetos para o Android, Windows Phone e plataformas de FirefoxOS continham arquivos com nomes específicos, enquanto que nos projetos para iOS a palavra `LSRequiresiPhoneOS` ocorria em alguns arquivos. Com base nessas informações, foi escrito um parser que acessa o GitHub através de sua Search API e identifica projetos com esses indicadores. Em outras palavras, os participantes foram selecionados usando um *parser* que extrai informações do GitHub através da análise da documentação de várias plataformas para desenvolvimento de aplicativos para dispositivos móveis e projetos no GitHub.

Uma pesquisa foi realizada para os identificadores que indicam as plataformas e uma lista de projetos que possui o arquivo, ou um trecho de um arquivo com o texto que foi enviado, foi obtida. A partir desta lista de projetos, uma busca foi feita pelos dados dos usuários que estão relacionados aos projetos, seja como dono do projeto, ou como

³ <https://developer.github.com/v3/search/>

empregados ou como colaboradores. Assim, 1.000 possíveis participantes do questionário foram identificados e 400 informantes foram selecionados aleatoriamente para os quais o questionário foi distribuído utilizando o email disponível em seus perfis. Os dados foram coletados entre os dias 24 de janeiro de 2014 e 20 de fevereiro de 2014. A Figura 13 abaixo apresenta o número de respostas diárias.

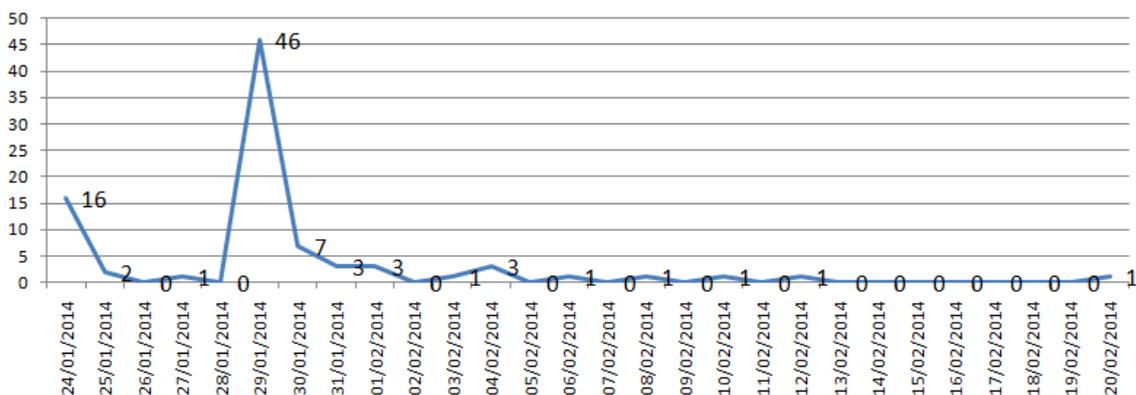


Figura 13: Intervalo e número de respostas diárias.

As questões abertas (6 questões), foram analisadas utilizando técnicas da Teoria Fundamentada em Dados (Strauss & Corbin, 2008), apresentada na Seção 3.3.3, ou seja, cada pergunta foi categorizada em um ou mais códigos (aspectos) que fizessem sentido com o que o respondente gostaria de transmitir.

4. RESULTADOS

4.1 Visão Geral

Essa dissertação visa compreender quais os principais aspectos positivos e negativos de para criação de aplicações em ecossistemas de software móveis a partir da perspectiva dos engenheiros de software. Assim, foram realizadas entrevistas semiestruturadas e o envio de um questionário para engenheiros de software. Nesse capítulo serão apresentados os resultados obtidos com a adoção da metodologia descrita.

Primeiramente serão apresentados os resultados identificados nas entrevistas, separando-os em pontos positivos e negativos. Pontos positivos, são todos os aspectos que algum engenheiro relatou como algo que traz algum benefício, ou algo que seja motivador para a prática da construção de aplicativos para ecossistemas de software móveis. Por outro lado, os aspectos considerados negativos são aqueles que são prejudiciais ou que dificultam a criação de aplicativos, ou até mesmo algo que traz desmotivação para adoção ou permanência em um ecossistema. Após a apresentação dos resultados das entrevistas, serão apresentados os resultados do conjunto de dados coletado via questionário. É importante deixar claro que esta dissertação não tem objetivo comparar diferentes ecossistemas de software móveis, portanto, os aspectos foram abstraídos apenas em positivos ou negativos.

4.2 Primeira Fase (Entrevistas)

Primeiramente será apresentado os resultados identificados nas entrevistas, separando-os em pontos positivos e negativos. A Tabela 2, apresenta estes aspectos. Observe que foram identificados sete aspectos positivos que foram categorizados por diferentes nomes, cada aspecto pode conter uma ou mais justificativas

Tabela 2 Aspectos positivos e negativos identificados durante as entrevistas.

Aspectos Positivos	Aspectos Negativos
Engajamento da Comunidade	Fragmentação
Linguagem de Programação e Ambiente de Desenvolvimento	Linguagem de Programação e Ambiente de Desenvolvimento
Popularidade	Emuladores
Documentação	Equipamentos de Hardware Específicos
Fragmentação	Distribuição de Aplicativos
Distribuição de Aplicativos Rentabilidade	Rentabilidade

4.2.1 Aspectos Positivos.

Engajamento da Comunidade

A participação em comunidades pode ajudar do ponto de vista das atualizações de melhorias e inovações para o ecossistemas de software. O engajamento da comunidade, portanto, é apontado como um aspecto positivo. A seguir, um trechos de entrevista que afirma o que foi apresentado.

“A comunidade é essencial, ela diminui bastante a curva de aprendizado. Isso eu posso afirmar nas duas. Ainda bem que ambas as comunidades são bem desenvolvidas” – Entrevistado 5.

O rápido tempo de resposta pela comunidade de engenheiros de software também foi mencionado.

“A comunidade ajuda bastante, é bem engajada, tu faz uma pergunta e logo em seguida tem a resposta ou te dá a dica” – Entrevistado 1.

Muito além de atualizações, a participação em comunidades engajadas resulta em informação novas, isto é, a própria difusão de inovações.

“Me ajuda muito, porque além de eu poder contribuir eu posso receber muita informação que eu nem sabia que existia” – Entrevistado 4.

Os resultados apresentam a figura do líder de opinião. Pessoas que influenciam a decisão dos outros sobre a adoção de inovações, fornecendo conhecimento, informações, conselhos, etc.

“Acompanho as publicações e livros de engenheiros do Google, que possuem um conhecimento absolutamente profundo. Acompanho o próprio canal do Android Developers, acompanho também o canal de ferramentas que são mais tradicionais no Java como o Maven que facilita ai a aplicação continua no Android. Canal da Caelum que tem várias pessoas que sabem responder os problemas ou Google Plus que é uma rede mais técnica” – Entrevistado 9.

Os entrevistados relataram que o canal utilizado para comunicação é a Internet, onde existem vários grupos (blogs, sites oficiais da plataforma, fóruns, etc) que compartilham informações para o entendimento de um ecossistema de software móvel.

“Faço diversas buscas, muitas delas obtenho a resposta pelo Stack Overflow ou pessoal no GitHub” – Entrevistado 9.

O principal fórum de discussão relatado trata-se do Stack Overflow⁴. Apesar da importância deste site, a busca é feita geralmente pelo site de busca do Google que quase sempre apresenta o Stack Overflow como fórum que possui a solução/informação desejada, conforme pode ser visto nas citações a seguir:

“Geralmente minhas dúvidas são sanadas pelo Stack Overflow. Ele geralmente aparece em primeiro no Google” – Entrevistado 7.

“Eu procuro no Google e geralmente as primeiras respostas vem do Stack Overflow, eu tendo, mesmo quando as primeiras respostas não são do Stack Overflow, eu procuro primeiro lá” – Entrevistado 5.

Um dos entrevistados relata que tem inclusive maior preferência em comunicações interpessoais, por isso sua participação é mais por esta forma de comunicação:

“Para a comunidade externa eu não tenho assim uma colaboração maior, apenas alguns tutorias. Em geral eu sou mais contributivo com o pessoal do meu trabalho, gosto também de dar aulas, fazer DOJOS, coisas mais ao vivo que participação online” – Entrevistado 9.

A comunicação interpessoal não foi mencionada com frequência, porém em alguns trechos de entrevistas é possível notar que este tipo de comunicação é bastante relevante para a adoção ou aceitação da plataforma, conforme trecho a seguir:

“Eu acho muito importante o pessoal ter um grupo para participar, para tirar dúvidas, ter pessoas que tu possa contar”. – Entrevistado 3.

Outro entrevistado revela que se comunica com outros desenvolvedores via internet porém de forma pessoal através de e-mails ou Skype. Neste caso, percebe-se a utilização da internet como facilitadora da comunicação interpessoal.

⁴ stackoverflow.com

“Sim, tem cada vez mais desenvolvedores trabalhando para a plataforma. Então fica mais fácil encontrar informações. Geralmente tenho conversas via Skype ou trocamos e-mails” – Entrevistado 6.

Popularidade

O aspecto da popularidade trata do número de usuários finais, ou seja, os resultados apontaram que um ecossistema de software com uma grande quantidade de usuários finais é um atrativo a mais para adoção e permanência no ecossistema. Conforme podem ser observadas em citações a seguir:

“Muita gente utiliza o Android, a maior parte, então vou desenvolver para uma plataforma que bastante pessoas utilizem” – Entrevistado 1.

“Eu diria que ia depender do objetivo, se ele quiser atingir uma quantidade maior de pessoas, ele tem que desenvolver para Android, pois pouca gente no Brasil tem iOS” – Entrevistado 5.

De acordo com a análise de dados realizada, pôde-se perceber que a popularidade para a plataforma Android tem maior grau de se comparado a com o iOS, um dos fatores que contribuem para isto é vantagem que tem o consumidor final em pode comprar um smartphone mais barato que os aparelhos da Apple, o iPhone.

“O Android é o sistema operacional que deve ter mais de 50% da fatia de usuários, até porque o Android está presente em vários modelos de smartphones e preços acessíveis” – Entrevistado 3

Documentação

O aspecto da documentação trata sobre a localização das diretrizes oficiais fornecidas pela fabricante da plataforma, as boas práticas de codificação, dentre outras especificações da plataforma escolhida pelos informantes. Conforme mencionado, a documentação é um aspecto positivo para adoção de um ecossistema, pois é algo que facilita o desenvolvimento no mesmo.

“A documentação é sensacional, isso é uma das melhores coisas que tem neles, a documentação é muito boa, quando tu tem o Xcode, tu consegue direto ver a documentação dentro dele” – Entrevistado 3

“É muito boa, eles fizeram uma documentação bem detalha e tem todos os passos. As vezes pra você achar uma coisa, tem tantas coisa detalhando que até chegar nela demora um tempo pois a documentação está tão bem detalhada que isso pode acontecer” – Entrevistado 5

A documentação pode ser também definida como não oficial, ou seja, documentos promovidos pela comunidade de engenheiros de software, tais como livros, tutoriais na forma de texto, vídeo-aulas, etc.

“Acompanho as publicações, livros de engenheiros do Google que possuem conhecimentos profundos, fizeram muito para a plataforma. Dominam muito sobre performance de interfaces e soluções de interface. Acompanho o próprio do Android Developers, acompanho muito o canal de ferramentas que são mais tradicionais no Java como o Maven que facilita ai a aplicação continua no Android. Próprio canal da Caelum que tem pessoas que sabem responder os problemas” – Entrevistado 9

Fragmentação

A fragmentação refere-se a existência no mercado de diferentes dispositivos de hardware baseados no mesmo sistema operacional mas que possuem atributos diferentes. Esses atributos podem ser: capacidade de memória interna, velocidade da unidade de processamento central (CPU), resolução, tamanhos de tela, versão do sistema operacional, entre outros.

A fragmentação contribui com o grande número de usuários finais (Popularidade). Ela pode ser considerada um aspecto positivo para adoção da plataforma de um ecossistema, uma vez que o sucesso de uma aplicação depende de um grande número de usuários finais.

De acordo com a análise de dados realizada, pôde-se perceber que a popularidade para a plataforma Android tem maior grau de se comparado a com o iOS, conforme citado por alguns dos respondentes:

“Muita gente utiliza o Android, a maior parte, então vou desenvolver para uma plataforma que bastante pessoas utilizem” – Entrevistado 1.

“Se quiser atingir uma quantidade maior de pessoas, ele tem que desenvolver para Android, pois pouca gente no Brasil tem iOS” – Entrevistado 5.

Um ecossistema fragmentado apresenta dispositivos de hardware com preços mais baixos, conforme citação de um dos entrevistado.

“O Android tem diversos dispositivos baratos, não é a toa que, disparadamente, ele vai ser a maior plataforma usada, não tem como combater isso” – Entrevistado 2.

Linguagem de Programação e Ambiente de Desenvolvimento

De acordo com os resultados da análise das entrevistas, a união entre Java e Eclipse configura um ambiente de desenvolvimento bem mais familiar para a maioria dos entrevistados. As citações a seguir apoiam este resultado.

“No caso do Android, um ponto que me ajudou muito início foi Java, eu já sabia Java, então a adaptação foi apenas aprender uma nova API. Tive um pouco mais de dificuldade no iOS, pois não conhecia a linguagem” – Entrevistado 5.

“Gostei da parte de desenvolvimento, achei muito parecido com alguns frameworks em Java no qual já havia trabalho, então eu optei por continuar desenvolvendo em Android” – Entrevistado 8.

Os trechos de entrevista acima também revelam que de forma geral os desenvolvedores estão mais costumados com a linguagem Java.

Distribuição de Aplicativos

A distribuição de aplicativos refere-se as diferentes formas que os desenvolvedores possuem para disponibilizar seus aplicativos aos usuários finais.

Alguns respondentes afirmam que quanto melhor for a burocracia para distribuição ou publicação, melhor será a agilidade do processo de instalação para o usuário final.

“Primeiramente foi no Android, pois é um plataforma livre e então você tem uma gama de facilidades para desenvolver pra ela. Além disso, ela não tem tantas restrições que o iOS tem. É mais fácil um desenvolvedor independente

entrar neste caminho e começar a lançar seus aplicativos. O iOS tem restrições quanto, por exemplo, pagamento anual” – Entrevistado 7.

Diferente do iOS, para disponibilizar uma aplicação na loja virtual do Android (Google Play) o desenvolvedor precisa pagar uma taxa única de USD \$25 (vinte e cinco dólares). Isto, é tratado como uma vantagem.

“A plataforma facilita sim, pois você paga um valor único de 25 dólares, isto uma vez só e você pode disponibilizar sua aplicação no Google play” – Entrevistado 7.

Rentabilidade

A partir da análise dos dados das entrevistas pode-se concluir que a rentabilidade pode ser considerada como um aspecto importante que influencia na adoção de uma determinada plataforma. Conforme é apresentado na citação abaixo:

“Não vou mentir que foi por dinheiro, eu percebi que os aplicativos para mobile iriam entrar em ascensão e gerariam muito dinheiro e eu percebi aí a oportunidade e fiz dela uma carreira para seguir” – Entrevistado 3.

4.2.2 Aspectos Negativos

Fragmentação

Se por um lado a fragmentação foi identificada como um aspecto positivo, por outro lado também é um aspecto negativo por forçar o engenheiro de software a testar seu aplicativo em diferentes dispositivos e tendo que se preocupar com a interface de usuário conforme é apresentado abaixo:

“Tendo desenvolvido pros dois [sistemas operacionais móveis: Android e iOS], pra mim, foi mais interessante desenvolver para iOS, por que você se preocupa menos com vários layouts, é algo que me faz não gostar de trabalhar com Android. Android tem uma gama muito grande de tamanhos de tela e dependendo do tamanho, o layout pode ficar bem diferente entre elas. Já pra iOS não, você tem o iPhone e iPod Touch que são do mesmo tamanho e tens o iPad. São poucos dispositivos” – Entrevistado 5.

A fragmentação é prejudicial também no que tange a versão do sistema operacional. Muitos dispositivos apresentam-se desatualizados e por isso não são compatíveis com aplicativos novos. As citações a seguir afirmam isso:

“Um problema de desenvolvimento do Android é a retro-compatibilidade, a gente deixa de desenvolver coisas mais legais porque a aplicação deve ser compatível com o Android 2.2” – Entrevistado 5.

“Por eu ter smartphones de diferentes versões eu sei mais ou menos o que cada versão me oferece. Então, to sempre procurando saber se tem versão nova. O Android tem um problema: Instabilidade, pois ele tem milhões de versões e sua aplicação final deve ser compatível com todas e isso faz com que o desenvolvedor tenha que ter pelo menos três modelos de smartphones com diferentes versões” – Entrevistado 7.

O alto grau de complexidade por parte de um ecossistema de software fragmentado reflete em vários testes em diferentes dispositivos para que o desenvolvedor tenha maior garantia que a experiência no uso do aplicativo por usuários seja semelhante; em outras palavras, que a aplicação se comporte bem em qualquer dispositivo de *hardware*. O seguinte trecho de entrevista ilustra este fato.

“Eu sempre faço um deploy no meu próprio dispositivo conectado ao computador. Tenho uma coleção de dispositivos para testar diferentes densidades, e versões” – Entrevistado 9.

Emuladores

Os emuladores são duramente criticados por não possuírem todas as características dos dispositivos móveis reais. Por exemplo, sensores de temperatura, acelerômetro, gestos, touch screen, etc. Além disso, outro fator que causa grande impacto durante os testes é que o emulador tem um desempenho muito inferior em relação a um dispositivo real.

“O emulador é a pior coisa para o desenvolvimento, pois é muito lento. E quando você precisa debugar, ele fica ainda mais lento” – Entrevistado 8.

Outro problema encontrado é que os emuladores não representam todos os diferentes dispositivos de hardware conforme visto na seguinte citação.

“Nem tudo dá pra emular nos emuladores, então é um pouco complicado desenvolver só utilizando o emulador. O melhor mesmo é testar com o dispositivo móvel no USB. Ai você roda sua aplicação direto no eclipse com o dispositivo móvel” – Entrevistado 8.

“Há uma certa dificuldade, por exemplo, como você faria um zoom no emulador? fazer isto com o mouse pra quando está testando no emulador. Existem algumas soluções, mas em geral todos os desenvolvedores que eu conheci tem um aparelho para testes” – Entrevistado 9.

A utilização de emuladores pode mascarar a realidade, ou seja, por mais que a aplicação se comporte bem durante a emulação o comportamento em dispositivos reais não é o mesmo, o Entrevistado 5 é enfático ao falar sobre este aspecto.

“No caso do Android, muitas vezes quando tu executa com código no emulador na sua máquina o código funciona, mas quando você testa em um dispositivo real, não funciona” – Entrevistado 5.

Emuladores são piores ainda quando a aplicação exige muito recursos de hardware, por exemplo, processamento, memória, etc.

“O Emulador é lento e como desenvolvemos uma aplicação em 3D imagina o quanto ficaria lento isto no emulador, A gente quando vai fazer o build da aplicação, a gente só faz o build direto para o celular” – Entrevistado 7.

Equipamentos de Hardware Específicos

Equipamentos específicos são dispositivos de hardware necessários ao desenvolvimento de aplicativos para uma determinada plataforma. Por exemplo, para desenvolver aplicativos para iOS é necessário ter um Macintosh. Neste sentido, quanto maior a necessidade de máquinas/equipamentos específicos para o desenvolvimento, menor será a adoção do ecossistemas de software. Isto é ainda mais agravado se o custo destes equipamentos for alto.

“O problema é o fato de você precisar de um Macintosh para desenvolver, eles não são baratos, então isso dificulta a adoção por novos desenvolvedores” – Entrevistado 3.

“Eu decidi desenvolver para Android por não ter um Macintosh. Então, pra quem está começando e não tem recursos financeiros é complicado desenvolver para iOS” – Entrevistado 7.

Diferente do ecossistema de software iOS, os entrevistados relatam que o ecossistema Android é exatamente o oposto quanto ao aspecto negativo da necessidade de compra de equipamentos de hardware específicos conforme citação abaixo:

“Um desenvolvedor Android não precisa de uma máquina específica, pode desenvolver em qualquer computador que seja compatível com a JVM [Java Virtual Machine]” – Entrevistado 9.

A citação abaixo mostra claramente que este aspecto é negativo para adoção de ecossistemas de software móveis:

“Não tenho interesse em comprar uma máquina para desenvolver iOS, acho que ela é uma plataforma muito fechada.” – Entrevistado 9.

Linguagem de Programação e Ambiente de Desenvolvimento

É importante observar que a compatibilidade ou familiaridade foi considerado um aspecto ambíguo, pois existem relatos negativos em relação a este aspecto, conforme citação abaixo.

“Objective-C é bem diferente de qualquer linguagem orientada objeto que a gente conhece hoje em dia. A leitura dos métodos, a leitura de chamada de métodos, de criação de método é completamente diferente das [linguagens] orientadas a objeto que a gente tá acostumado, como Java” – Entrevistado 2.

O ecossistema iOS, que utiliza como linguagem de programação o Objective-C e o Xcode como ambiente integrado de desenvolvimento, não se mostram familiares ou compatíveis com experiências passadas dos desenvolvedores, possivelmente pelo fato da linguagem Objective-C não ser tão popular quanto o Java.

“Uma coisa que eu não gosto no Objective-C é rastrear exceções, no Java quando uma exceção ocorre, ele mostra o Stack Trace e você consegue ver qual foi a linha de código de qual classe que desencadeou erro” – Entrevistado 5.

“A IDE travava, dava erros que você não sabia o que era, ai tinha que fechar a IDE para parar o erro” – Entrevistado 8.

Distribuição de Aplicativos

As atualizações do aplicativo após terem sido publicadas também é algo questionado nas entrevistas, relatam negativamente sobre o tempo que o cliente deve aguardar para que a aplicação esteja disponível. A seguir, trechos de entrevistas que ilustram estes fatos são apresentados.

“... tem sérios problemas como é feito o upload para a loja” – Entrevistado 9.

Esta demora é feita em detrimento de avaliações quanto a segurança do aplicativo, pois é feita uma avaliação da aplicação antes da disponibilização, como afirma o entrevistado:

“No caso quando tu submete para App Store eles tem um comitê, que teoricamente avalia a tua aplicação e libera para o mercado. Isso acontece por restrição de segurança mesmo” – Entrevistado 1.

Tanta liberdade no que se refere a distribuição de aplicativos foi apontado como uma característica negativa, conforme é relatado por um dos entrevistados.

“Eu, particularmente, não gosto muito do modelo do Android, porque dá muita liberdade das pessoas colocarem qualquer tipo de conteúdo lá, liberdade é bom, mas tem muito conteúdo inútil na Play Store. É difícil, você encontrar aplicativos novos. Isso já não acontece tanto com a Apple” – Entrevistado 5.

Rentabilidade

De acordo com as entrevistas, diferentes ecossistemas de software móveis apresentam diferentes rentabilidades, por exemplo, o ecossistema Android é mencionado por muitos entrevistados como pior em relação ao iOS, pois a perfil de compra dos usuários são diferentes.

“Eu acho que a cultura das pessoas que possuem iPhone é diferenciada na postura de compra de aplicativos, elas não tem pena em gastar em um aplicativo, são mais consumistas, diferente do Android, que já é mais popular,

as pessoas ainda relutam em comprar aplicativos, tanto que o maior número de aplicativos que ela tem é de graça” – Entrevistado 4.

Entretanto, outros entrevistados afirmam que rentabilidade é diferenciada em outros países, ou seja, este aspecto depende do contexto onde o engenheiro de software reside.

“Eu achava [usuários do ecossistema de software Android não pagam por aplicativos] que isso era no mundo todo, mas não. Hoje com uma aplicação no mercado vejo que isto é muito mais forte aqui no Brasil... pessoas de outras partes do mundo compram e isto foi uma surpresa, chegamos em mais que 500 downloads em menos de um mês” – Entrevistado 7.

4.3 Segunda Fase (Questionários)

Após a submissão do questionário para 400 e-mails, obteve-se oitenta e seis (86) respostas para nossa pesquisa (taxa de resposta de 21,5%). O questionário possuía três questões de múltipla escolha, que tinham como objeto identificar se o respondente era de fato um engenheiro de software, o quanto ele/ela classificava sua própria experiência e para quais plataformas móveis desenvolvia. Tais perguntas eram obrigatórias, devendo o respondente selecionar uma ou mais opções de resposta. Além disso, o questionário apresentava três questão de múltipla resposta, ou seja, o respondente deveria selecionar apenas uma opção dentre as apresentadas. O objetivo dessas questões eram identificar o número de aplicativos desenvolvidos, se o informante tinha desenvolvido algum aplicativo nos dois últimos anos, e se sua melhor aplicação está publicada em alguma loja virtual. Tais questões eram obrigatórias, assim como nas perguntas de múltipla escolha.

As questões abertas do questionário procuravam entender pontos negativos ou positivos relacionados as plataformas de desenvolvimento, mas não eram obrigatórias. Por exemplo, 15 foi o número de respondentes que não relataram nada para a pergunta *“What did you like best about developing the app for its platform and why?”*, 25 para a pergunta *“In what aspects does this app's platform fall short? Why?”*, 15 para a pergunta *“What is the most crucial difference between mobile development and desktop or web development?”*, 23 para a pergunta *“What is the hardest problem that you need to cope with in mobile development?”* e 41 para a pergunta *“Which problems have actually kept*

you from succeeding, ones that you were unable to solve?”. Cada uma dessas perguntas serão analisadas detalhadamente nas próximas seções.

É importante destacar que as questões fechadas ajudaram a estabelecer diferentes níveis de análise, por exemplo, os aspectos foram separados de acordo com opiniões de engenheiros com menos e mais experiência. Além disso, isto foi interessante para se obter uma análise descritiva dos dados. As questões abertas, foram analisadas utilizando técnicas da teoria fundamentada em dados, ou seja, cada pergunta foi categorizada em um ou mais códigos (aspectos) que fizessem sentido com o que o respondente gostaria de transmitir. Por exemplo, a seguinte citação: *“Well documented and standardized UI requirements”*, foi categorizada com o nome de “documentação”, pois é sobre disso que o respondente está tratando, e é um aspecto positivo pois é um elogio e não uma crítica.

Uma tabela relacionada aos pontos positivos e negativos desta fase será apresentada após a estatística descritiva dos dados e classificação dos respondentes.

4.3.1 Estatística Descritiva

Esta seção apresenta a estatística descritiva com os resultados do questionário, descrevendo assim de um modo geral as respostas obtidas.

A Tabela 3 apresenta a localização parcial dos respondentes do questionário. A partir das informações de perfil fornecida pelo desenvolvedor, foi possível identificar a localidade da maioria dos respondentes. Não foi possível obter a informação de *todos* os desenvolvedores, pois esta informação em alguns casos não existia por não ser obrigatória no GitHub. As dez localidades com maior número de respostas foram: 1º USA (24); 2º Canadá (4); 3º Brasil, Índia, Holanda e Reino Unido (3); 4º Bélgica, Camboja, China e França (2). Observe que o total de respondentes que não apresentaram localidade foi de 17.

Tabela 3 Localização parcial dos participantes do questionário.

Localidade	Total	%
Estados Unidos	24	27,9
Não Respondido	17	19,7
Canada	4	4,6
Brasil	3	3,4

Índia	3	3,4
Holanda	3	3,4
Reino Unido	3	3,4
Bélgica	2	2,3
Camboja	2	2,3
China	2	2,3
França	2	2,3

A Tabela 4 apresenta os diferentes níveis de experiência em desenvolvimento de aplicativos móveis, ou seja, a quantidade de desenvolvedores que possuem **experiências profissionais, não profissionais** e se os mesmos **contribuem em projetos open source**. Neste caso, esta era uma questão de múltipla escolha, ou seja, o desenvolvedor poderia selecionar mais de uma alternativa (resposta).

Tabela 4 Níveis de experiência em desenvolvimento de software móvel.

Níveis de Profissionalismo	Total	%
Profissional	56	38
Não Profissional	60	41
Contribuição em Projetos Open Source	30	21

As cinco plataformas mais citadas, referente a pergunta sobre para quais plataformas o respondente do questionário desenvolve, foram: Android (51), Aplicação Web (42), iOS (41), Windows Phone (16) e Firefox OS (15). Observe que esta pergunta também é de múltipla escolha. A Tabela 5 apresenta um panorama das plataformas mais utilizadas. 180

Tabela 5 Plataformas mais utilizadas de acordo com os respondentes deste estudo.

	Total	%
Android	51	59
IOS	41	48
Firefox OS	15	17
Symbian	2	2
J2ME	3	3
Windows Phone	16	19
Web Application	42	49
Multiplataforma	7	8
Outros	3	3

De forma a compreender ainda mais o nível de experiência dos desenvolvedores respondentes do questionário, a Tabela 6 apresenta em cinco escalas o tempo de

experiência de desenvolvimento de aplicativos móveis por parte dos desenvolvedores que responderam ao questionário. Observe que o maior conjunto de informantes está presente na faixa de 1 a 2, e 3 a 5 anos de desenvolvimento, representando ao todo 77% da amostra.

Tabela 6 Tempo de experiência em desenvolvimento de aplicativos móveis.

Tempo de Desenvolvimento	Total	%
Nada	3	3
Menos de 1 Ano	12	14
1 a 2 Anos	35	41
3 a 5 Anos	31	36
Mais de 5 Anos	5	6
Total	86	

Outra informação que também reflete a experiência dos respondentes do questionário refere-se ao número de aplicativos desenvolvidos em suas carreiras, conforme ilustra a Tabela 7. Novamente, pode-se observar que 89% dos respondentes dizem ter desenvolvido entre 2 ou mais aplicativos móveis.

Tabela 7 Número de aplicativos desenvolvidos.

Número de Aplicativos	Total	%
0	1	1
1	9	10
2 a 3	28	33
4 a 6	24	28
Mais de 6	24	28
Total	86	

Outra informação que complementa a anterior trata da disponibilidade de aplicativos nas lojas virtuais tais como: Play Store e Apple Store. É interessante observar que apesar das plataformas disponibilizarem portais para que aplicativos sejam publicados, mais de 44% dos desenvolvedores participantes deste estudo, revelam não ter aplicativos disponíveis em lojas virtuais, conforme Tabela 8.

Tabela 8 Disponibilidade de aplicativo na loja virtual.

	Total	%
Sim	47	0,552941

Não	39	0,447059
Total	86	

Ao serem questionados se desenvolveram algum aplicativo móvel nos últimos dois anos, 9% disseram que não desenvolveram, 33% que desenvolveram porém sozinhos, 38% disseram que sozinho e com times (equipe de trabalho) e por fim, 20% afirmam terem desenvolvido apenas em times, ver Tabela 9.

Tabela 9 Desenvolvimento de algum aplicativo móvel nos últimos dois anos.

	Total	%
Não.	8	0,09
Sim, Sozinho.	28	0,33
Sim, Com Time.	17	0,20
Sim, Sozinho e Com Time.	33	0,38
Total	86	

Ao serem questionados sobre quais ferramentas utilizam enquanto desenvolvem aplicativos móveis, os participantes citaram diversas ferramentas. Porém, as 10 mais expressivas estão listadas em ordem decrescente na Tabela 10. Deve-se observar que esta também era uma questão de múltipla escolha.

Tabela 10 Ferramentas mais utilizadas para o desenvolvimento de aplicativos móveis.

Eclipse	31
Xcode	25
Emuladores	11
Sublime Text	8
Android Studio	7
Visual Studio	7
Dispositivos Reais	6
Photoshop	5
Phonegap	5
Testflight	5
Crashlytics	3
Netbeans	3

4.3.2 Classificação dos Respondentes

Para análises posteriores a respeito de pontos positivos e negativos em relação ao desenvolvimento de aplicativos para dispositivos móveis, bem como a respeito das diferenças entre desenvolvimento móvel comparado ao desenvolvimento desktop e web,

optou-se pela classificação dos respondentes do questionário em dois grupos: os de menor experiência e maior experiência. Esta separação se deu da seguinte forma: do total de respondentes do questionário (86), 37 foram classificados como menos experientes enquanto que 48 foi a quantidade de respondentes mais experientes. Esta classificação se deu em função da quantidade de aplicativos desenvolvidos, ou seja, quem alegou ter desenvolvido entre 1 à 3 aplicativos foi classificado como menos experiente, enquanto que aqueles que afirmaram ter desenvolvido entre a 4 ou mais aplicativos foram classificados como mais experientes. Conforme mencionado anteriormente, (ver Tabela 11), de um modo geral é possível observar que os desenvolvedores com mais anos de experiência, geralmente desenvolveram um maior número de aplicativos moveis. Observe também que o total desta tabela (85) não representa o total de respondentes do questionário (86). Entretanto, este foi desconsiderado na classificação por afirmar não ter tido nenhum aplicativo desenvolvido e por não ter nenhum tempo de desenvolvimento. Assim, para as próximas análises o número de respondentes tratado será de exatamente 85.

Tabela 11 Relação entre tempo de experiência e números de aplicativos desenvolvidos.

Número de Aplicativos Desenvolvidos	Tempo de Desenvolvimento (Ano)				
	Menos - 1	1 - 2	3 - 5	5 - Mais	Total
1	4	5	0	0	9
2 - 3	10	15	3	0	28
4 - 6	0	12	10	2	24
6 - Mais	0	3	18	3	24
Total	14	35	21	5	85

Conforme mencionado anteriormente, o resultado da classificação baseada na experiência foi de 37 *desenvolvedores menos experientes*, pois a soma da quantidade de respondentes que desenvolveram apenas um aplicativo com a quantidade de respondentes que desenvolveram entre dois ou três aplicativos é de trinta e sete (37). Por outro lado, 48 desenvolvedores foram classificados como mais experientes, novamente, ao aplicar uma simples soma entre os que desenvolveram acima de quatro aplicativos.

Apesar dos respondentes relatarem que desenvolviam para mais de uma plataforma, as demais questões abertas do questionário infelizmente o respondente

geralmente relatou com foco em apenas uma plataforma, talvez a que fosse mais expressiva em seu cotidiano, embora algumas vezes o aspecto pudesse ser aplicado para mais de uma. Por exemplo, o respondente 10 afirma que uma das melhores coisas para as plataformas em que ele desenvolve aplicativos é o fato da existência de um grande número de usuários finais.

“Both Android and iOS -- Huge users” – Respondente 53.

4.3.3 Aspectos Positivos

Esta seção tem por objetivo apresentar os resultados relacionados aos pontos positivos identificados pelos respondentes do questionário. Isto é, o questionário apresentou uma questão (*What did you like best about developing the app for its platform and why?*) que refere-se ao que o respondente (desenvolver ou engenheiro de software) gosta mais ao construir aplicativos para o ecossistema escolhido, ou seja, esta é uma pergunta direta sobre os pontos positivos do ecossistema de software pois trata exatamente o que o respondente percebe de bom no mesmo.

A Tabela 12 apresenta a quantidade de respostas consideradas, desconsideradas e quantos não responderam a questão. Uma tabela semelhante será apresentada nas demais análises.

Tabela 12 Classificação das respostas para a pergunta: *What did you like best about developing the app for its platform and why?*

	Menor Experiência	Maior Experiência
Não Responderam a Pergunta	8	7
Respostas Desconsideradas	9	11
Respostas Consideradas	20	30

Do grupo de respondentes menos experientes, oito (8) não responderam a questão resultando em apenas vinte e nove (29) relatos a respeito do que gostam mais quanto ao desenvolvimento de aplicativos para dispositivos móveis para o ecossistema escolhido. Entretanto, do total de respostas (29), nove (9) não puderam ser classificadas por não conter nenhum conteúdo significativo ao que se foi perguntado ou se tornaram de difícil compreensão. Resumindo, do grupo de menos experientes (37) foi considerado apenas vinte (20) respostas por permitir abstrair uma categoria para o que foi mencionado.

De forma semelhante ao que foi apresentado no paragrafo acima, sete (7) é o número dos desenvolvedores mais experientes que não responderam a esta pergunta, enquanto que onze (11) respostas foram desconsideradas por não conter conteúdo significativo ou por serem de difícil compreensão. Desta forma, o total de respostas válidas para o grupo de mais experiência é de trinta (30).

A Tabela 13 apresenta todos os aspectos positivos identificados a partir das respostas dos respondentes do questionário relacionado a seguinte pergunta: “*What did you like best about developing the app for its platform and why?*”. Além disso, conforme visto na Seção 4.3.2, a Tabela 13 separa a opinião em dois grupos de desenvolvedores: os de menos experiência e mais experiência.

Tabela 13 Pontos positivos de acordo com os engenheiros de *software* respondentes.

Aspectos Positivos	Menos Experientes	Mais Experientes	Total
Desenvolvimento Multiplataforma	6	4	10
Ambiente de Desenvolvimento	2	7	9
Documentação	6	2	8
Linguagem de Programação	2	5	7
Distribuição	1	6	7
Engajamento da Comunidade	2	2	4
Fragmentação	2	2	4
Popularidade	0	2	2
Satisfação do Usuário Final	0	1	1
Total	21	31	52

De um modo geral, 9 pontos positivos foram relatados pelos informantes, conforme visto na Tabela 13. Os próximos parágrafos apresentam algumas das respostas que fizeram com que se chegasse nessas conclusões. Será mantido o texto original das respostas, ou seja, o texto em inglês.

Desenvolvimento Multiplataforma

O **desenvolvimento multiplataforma** é visto como um aspecto positivo por dez (10) respondentes, pois eles alegam que desenvolver um aplicativo uma única vez é algo mais simples permitindo assim aproveitar o mesmo aplicativo em diferentes plataformas. Como é de se esperar, a maioria dos respondentes que afirmam isto são

desenvolvedores que trabalham com ferramentas relacionadas a multiprogramação, por exemplo, o Phonegap. A seguir, é uma resposta relacionada ao que foi discutido neste parágrafo é exemplificada.

“We're developing apps for multiple platforms at once using Phonegap. This saves us from redeveloping the app multiple times” – Respondente 72.

“I like developing apps both android and iOS at the same time in same application, i think its more challenging and it makes how can i make it easy to develop in cross platform easily” – Respondente 13.

Documentação

O aspecto da **documentação** a localização das diretrizes oficiais fornecidas pela fabricante da plataforma, as boas práticas de codificação, dentre outras especificações - da plataforma que foi escolhida pelos informantes. Bem como, a documentação não oficial, ou seja, documentos promovidos pela comunidade de engenheiros de software, tais como tutoriais na forma de texto, vídeo aulas, etc.

Uma boa documentação é muito útil para iniciantes, pois eles podem se guiar pela experiência das pessoas que projetaram a plataforma.

“Well documented and standardized UI requirements” – Respondente 66.

“It is well documented and flexible, allowing homescreen widgets, which was the main purpose of my app” – Respondente 19.

“In Android (...) it has a good documentation and it's easy to get results in a short period” – Respondente 26.

É importante ressaltar que os respondentes 66, 19 e 26 eram desenvolvedores com menos de um ano de experiência, conforme relataram.

A documentação é útil também para desenvolvedores experientes, quando os mesmos possuem um problema específico na plataforma, não algo comum a programação geral. Por exemplo, o respondente 12 e 26 possuem mais de 2 anos de experiência, 5 aplicativos desenvolvidos e ainda assim afirmaram o seguinte sobre a documentação.

“Google had a lot of good tutorials and example projects. So the information was out there” – Respondente 12.

Linguagem de Programação e Ambiente de Desenvolvimento

Outro ponto identificado como positivo trata-se do **ambiente de desenvolvimento**, isto é, o programa de computador que reúne várias ferramentas de apoio ao desenvolvimento de software. Abaixo são apresentadas algumas das respostas que apontam como positivo o ambiente de desenvolvimento para a plataforma a qual desenvolvem.

“Xcode is a great IDE. Apple's new iOS 7 and flat design makes life easier and everything looks better, I think” – Respondente 57.

“iOS, because I love Apple developing enviroment” – Respondente 51.

“Android and Apple both have excellent IDEs” – Respondente 15.

A familiaridade com a **linguagem de programação** e com o **ambiente de desenvolvimento** fornecido pela plataforma é um aspecto que facilita e atrai desenvolvedores para a mesma, conforme pode-se verificar nas respostas a seguir:

“Java is easy and I have a big background in this language” – Respondente 2.

“I appreciate that Android is written in a commonly used language, as opposed to iOS Objective-C” – Respondente 73.

“Windows Phone is a .NET based platform. I love C# and the philosophy behind WP” – Respondente 40.

Engajamento da Comunidade

Outro ponto positivo trata-se do **engajamento da comunidade** ou seja, a percepção dos respondentes quanto a rede social que fazem parte dentro do qual ocorre a comunicação de inovações e que visam resolver problemas comuns. A seguir, duas respostas que refletem que o suporte dado pela comunidade é um ponto positivo para permanência no ecossistema.

“The community is also great, lots of developers.” – Respondente 28.

“Lot of developer support” – Respondente 52.

“The community has really made development much easier” – Respondente 9.

O engajamento de desenvolvedores nessas comunidades é essencial para a sua participação continuada e permanência nos ecossistemas, um dos respondentes afirma que um dos melhores aspectos do ecossistema é a quantidade de engenheiros de software engajados a compartilhar conhecimento.

“I appreciated the very extensive and supportive community” – Respondente 73.

Distribuição

Uma das formas de se **distribuir** aplicativos trata-se da **loja virtual**, esta é a principal forma de distribuir um aplicativo. A **loja virtual** é o intermediário entre o desenvolvedor e o usuário final. Ela oferece um ambiente de distribuição onde o desenvolvedor pode obter informações sobre o número de usuários que instalaram sua aplicação, de que país são, ler *reviews* dos usuários e receber os valores de venda do aplicativo, caso seja pago. A presença de uma **loja virtual** para o ecossistema de software móvel é apontado como uma grande ponto positivo, não somente isto, mas também por permitir perceber se o aplicativo está sendo bem utilizado pelos usuários finais, a baixo segue algumas respostas que refletem esta afirmação:

“Easily distributing work to thousands of customers through the App Store” – Respondente 65.

“I enjoy the feeling I get when I run an app that I just spent the past few days working on, seeing it work, and then publishing it and seeing thousands of other people enjoying it too” – Respondente 58.

Fragmentação

A **fragmentação** da plataforma é apresentada como um aspectos positivo por alguns respondentes. Relatam que diferentes classes sociais possuem dispositivos para o ecossistema em questão.

“Android - A wide range of devices are covered and also different ranges of social” – Respondente 58.

Um dos respondentes mostra que a fragmentação traz vantagens uma vez que os dispositivos ficam mais baratos.

“I prefer Java and Android. it is cheaper to develop for, the hardware is cheaper” – Respondente 7.

Popularidade

A **popularidade** que o ecossistema de software possui, ou seja, os usuários finais que de fato utilizam os aplicativos desenvolvidos se mostra como atrativo para adoção ou permanência no ecossistema. Este resultado pode ser visto nas citações a seguir:

“Both Android and iOS -- Huge users” – Respondente 10.

“Android. Because it is more popular in China” – Respondente 25.

A ideia é que quanto mais usuários, maior será a utilização de aplicativos. Além disso, perceber que a solução/aplicativo está de fato sendo realmente utilizada e aproveitada pelos usuários finais, traz muita satisfação para quem desenvolve.

“Taking advantage of iOS features to improve people's quality life” – Respondente 82.

4.3.4 Aspectos Negativos

Esta seção tem por objetivo apresentar os resultados relacionados aos pontos negativos identificados pelos respondentes do questionário.

O questionário apresenta três questões diretas que possibilitam identificar tais pontos negativos. As três questões são: *In what aspects does this app's platform fall short? why?*, *What is the hardest problem that you need to cope with in mobile development?* e *“Which problems have actually kept you from succeeding?”*. Além disso, outros aspectos negativos foram identificados na questão, *What is the most crucial difference between mobile development and desktop or web development?*. Esta questão não necessariamente trata de aspectos negativos, porém nenhum aspecto positivo foi relatado.

Tabela 14, apresenta a quantidade de respostas consideradas, desconsideradas e quantos não responderam as três questões abordadas no parágrafo anterior.

Tabela 14 Resultados do número de respostas consideradas, desconsideradas ou não respondidas para as principais questões que tratam de aspectos negativos.

<i>What is the hardest problem that you need to cope with in mobile development?</i>		
	Menor Experiência	Maior Experiência
Não Responderam a Pergunta	11	12
Respostas Desconsideradas	0	1
Respostas Consideradas	26	36
<i>In what aspects does this app's platform fall short? why?</i>		
	Menor Experiência	Maior Experiência
Não Responderam a Pergunta	11	14
Respostas Desconsideradas	10	11
Respostas Consideradas	16	25
<i>Which problems have you actually kept you from succeeding?</i>		
	Menor Experiência	Maior Experiência
Não Responderam a Pergunta	20	21
Respostas Desconsideradas	10	11
Respostas Consideradas	7	16

Tabela 15, apresenta a quantidade de respostas consideradas, desconsideradas e quantos não responderam a questão *“What is the most crucial difference between mobile development and desktop or web development?”*.

Tabela 15 Classificação das respostas para a pergunta: *What is the most crucial difference between mobile development and desktop or web development?*

	Menor Experiência	Maior Experiência
Não Responderam a Pergunta	7	9
Respostas Desconsideradas	4	7
Respostas Consideradas	26	32

A Tabela 16 apresenta os aspectos negativos relacionados as questões anteriormente mencionadas. As opiniões foram separadas em dois grupos: os engenheiros de software com de menos e mais experiência. Onze pontos negativos foram identificados. Vale destacar que o problema da fragmentação foi considerada

como o pior dentre os demais por atingir uma grande quantidade de citações negativas, no total foram 67 relatos negativos.

Tabela 16 Pontos negativos de acordo com os engenheiros de software respondentes.

Aspectos Negativos	Menos Experientes	Mais Experientes	Total
Fragmentação	27	40	67
Interface	21	23	44
Recursos	9	21	30
Emuladores	6	8	12
APIs	7	2	9
Ambiente de Desenvolvimento	3	3	6
Rentabilidade	1	4	5
Distribuição de Aplicativos	1	4	5
Código Fonte Fechado	1	3	4
Documentação	0	2	2
Linguagem de Programação	0	2	2
	76	112	188

Fragmentação

No que se refere a **fragmentação** da plataforma o problema está relacionado a criar ou construir aplicativos que sejam, desde o início, projetados para funcionar em diversos modelos de *hardware* que podem não conter recursos importantes (câmeras, acelerômetro, etc.), versões do sistema operacional ou até mesmo em diferentes plataformas. A seguir algumas respostas que refletem bem a fragmentação como um aspecto muito negativo para o desenvolvimento de aplicativos móveis.

“Too many compatibility problems” – Respondente 25.

A fragmentação dificulta o processo de teste do aplicativo, pois cada dispositivo possui diferentes resoluções, tamanhos de tela, processamento, etc. Um dos respondentes afirma isto na citação a seguir:

“A lot of different devices/manufacturers to test” – Respondente 2.

“Android is hard to maintain for with backwards compatibility, iOS has more limited reach.” – Respondente 68.

Apesar de muitos pontos negativos identificados, existem poucas referências a cada uma delas ou as vezes apenas uma única referência.

Interface

A **Interface** com usuário trata-se aqui da ligação, conexão ou transferência de informação do aplicativo para quem o utiliza. A função de um aplicativo, por exemplo, é trazer ao usuário maneiras eficientes para realizar tarefas que lhe sejam úteis. Portanto, alguns dos respondes do questionário apontaram que desenvolver boas interfaces é realmente um ponto muito diferente se comparado a programação desktop ou web, pois existem limitações em tamanho de tela ou até mesmo em pensar como um usuário. A seguir algumas respostas confirmam o que foi dito:

“More focus on user experience is needed in mobile” – Respondente 80.

“Most likely it is the different mind set needed: it's completely different, the UI, the way it's interacted with, the hardware available” – Respondente 59.

“UI can be tricky to display alot of data on a small screen” – Respondente 60.

Ter o cuidado com a interface com usuário se torna um trabalho mais árduo uma vez que existem vários dispositivos diferentes para serem testado.

“The amount of different devices you need to test for compatibility is ridiculously large.” – Respondente 12.

Recursos

Outro aspecto importante trata-se do cuidado que o desenvolvedores devem ter em relação aos limites de **recursos**, este foi também um dos grande aspectos que diferenciam em relação ao desenvolvimento desktop e web. A seguir algumas respostas que refletem este aspecto.

“Optimizing the internet usage, size and usability for mobile” – Respondente 38.

“Mobile devices have limitations in battery/performance” – Respondente 35.

Os emuladores foram relatados negativamente, este resultado converge com o que foi apresentado durante a fase de entrevistas. Abaixo é apresentado algumas citações que ilustram bem esta afirmação:

“One thing that's annoying is the emulator testing with Android - very slow and sluggish” – Respondente 35.

“Debugging can sometimes be hard. Emulators can't emulate for every device”
– Respondente 60.

Application Programming Interface

Alguns respondentes criticaram algumas *Application Programming Interface* (API), a API é um conjunto de rotinas e padrões estabelecidos por um software para a utilização das suas funcionalidades por aplicativos. Os respondentes afirmam que a algumas APIs apresentam bug ou que não apresentam **documentações** suficientes.

“Bugs in APIs API not open source” – Respondente 54.

“Lack of documentation for private frameworks... although that is to be expected” – Respondente 59.

“Bugs occurred by the low level code which we don't have access to” –
Respondente 62.

Código Fonte Fechado

Além disso, reclamam que tais APIs não são abertas, ou seja, o engenheiro não consegue alterar a API para suas necessidades, criticando assim o código **fonte fechado**. Como é possível observar ao ler os seguintes relatos:

“It doesn't much flexibility in terms of their source code” – Respondente 62.

“Its not really open” – Respondente 50.

Como dito anteriormente, alguns respondentes criticam a documentação. Conforme pode-se observar nas citações a seguir:

“Modifying official (undocumented) dev tools (eg iOS simulator) to add missing features” – Respondente 54.

Linguagem de Programação e Ambiente de Desenvolvimento

Conforme apresentado, poucos respondentes relatam problemas quanto a linguagem de programação ou ambiente de desenvolvimento. Segue abaixo alguns relatos relacionados a esses dois aspectos.

“Objective-C is totally crazy” – Respondente 2.

“When i was developing augmented reality application on android, it was very hard to use both C++ and Java at the same time in android” – Respondente 13.

“The Eclipse IDE is tricky making the design process harder in Android” – Respondente 7.

“Xamarin is not that great... is really bad, lots of crashes” – Respondente 24.

Emuladores

Os emuladores foram relatados negativamente, este resultado converge com o que foi apresentado durante a fase de entrevistas. Abaixo é apresentada algumas citações que ilustram bem esta afirmação.

“Debugging can sometimes be hard. Emulators can't emulate for every device”
– Respondente 60

“Emulators. They are all bad” – Respondente 70.

“Accessing device resources on the dev computer” – Respondente 71.

Rentabilidade

A **rentabilidade** foi considerada como um aspecto negativo relatado por alguns respondentes. Inclusive, os resultados apontam que o ecossistema de software iOS possui maior rentabilidade, conforma pode ilustrar a citação abaixo:

“I feel like there is more money and a better market in the iOS app store” – Respondente 22.

Um dos respondentes relatou que construía aplicativos apenas para o ecossistema Android, embora muito experiente com 6 aplicativos desenvolvidos, considera que é difícil obter lucros financeiros.

“It is hard to make money” – Respondente 25.

Outro, apresenta o mesmo aspecto mas com uma abordagem diferente. Afirma que é difícil conseguir um grande público de usuários.

“It's hard to get users” – Respondente 16

Distribuição

Finalmente, o aspecto da **distribuição de aplicativos** foi considerada negativamente de acordo com alguns respondentes. Alegam que as vezes as lojas virtuais não aceitam seus aplicativos.

“It can be a challenge to make the app “acceptable” for app stores” – Respondente 47.

“iOS falls short with the burden of application provisioning and releasing the application to the public. This can become very frustrating” – Respondente 56.

Mesmo quando a aplicação é aceita, demora para que esta esteja disponível para que os usuários façam a instalação em seus dispositivos. Conforme os resultados apontaram, esta situação é mais agravante para o ecossistema de software iOS.

“Fall on the long waiting times from push to App Store” – Respondente 57.

“It took 3 tries to get it accepted to Apple's AppStore. Very discouraging process if I didn't know that my end users really wanted this app” – Respondente 16.

5. DISCUSSÃO

5.1 Visão Geral

O presente capítulo tem por objetivo apresentar uma discussão sobre os resultados deste estudo, procurando analisar e relacionar os aspectos identificados e descritos anteriormente. É objetivo também discutir quais aspectos foram mais relatados na primeira fase do estudo e seus motivos, bem como na segunda fase. Além disso, é apresentada a motivação para a utilização de duas formas diferentes para coleta de dados, no caso da primeira fase a aplicação de entrevistas e na segunda fase questionários. Ambas as fases realizadas com engenheiros de software de diferentes idades, formações e níveis de experiência no desenvolvimento de aplicativos para ecossistemas de software móveis.

5.2 Aspectos Positivos, Negativos e Ambíguos

A Tabela 17 apresenta uma visão geral da integração dos aspectos positivos, negativos e ambíguos relacionados aos resultados observados na primeira e na segunda fase da pesquisa.

Essas classificações surgiram durante a fase de análise dos dados, por exemplo, os aspectos positivos são aqueles em que todos os engenheiros relataram de forma positiva o aspecto em questão. Os aspectos negativos, diferentemente dos positivos, foram vistos unicamente de forma negativa. Pode-se citar como exemplo os emuladores: todos os engenheiros que trataram deste tema foram enfáticos em relatar problemas relacionados a criação de aplicativos no que se refere a emulação. Por outro lado, o engajamento da comunidade é tratado unicamente de forma positiva por todos os entrevistados e respondentes do questionário que trataram deste aspecto. Entretanto, durante a análise dos dados, foi revelado que ora um determinado aspecto era relatado como positivo ou como negativo, ou seja, dependendo do contexto em que o engenheiro de software se encontra tal aspecto é entendido como positivo ou negativo, daí surgiu a necessidade de classifica-los como aspectos *ambíguos*. Por exemplo, apesar da fragmentação se revelar como um dos piores pontos negativos, ela contribui de forma positiva com relação a adoção ou permanência de usuários finais.

Tabela 17 Visão geral dos aspectos positivos, negativos e ambíguos.

Aspectos Positivos	Aspectos Negativos	Aspectos Ambíguos
Engajamento da Comunidade	Emuladores	Fragmentação
Desenvolvimento Multiplataforma	Equipamentos de Hardware Específico	Linguagem de Programação e Ambiente de Desenvolvimento
Documentação	Recursos	Popularidade
Rápido Desenvolvimento	Interface	Distribuição de Aplicativos
Satisfação do Usuário Final	Código Fonte Fechado	Rentabilidade
	APIs	

5.3 Relações Entre Diferentes Aspectos

A Figura 14, ilustra diferentes relações entre alguns dos aspectos apresentados na Tabela 17. Observe que os aspectos podem influenciar positivamente com indicação da seta verde, e negativamente com a indicação da seta vermelha outros aspectos. Por exemplo, a rentabilidade, compatibilidade tecnológica, facilidade em distribuir aplicativos, a popularidade e documentação foram aspectos considerados positivos para a comunidade, a ideia de positivo está intimamente relacionada a adoção ou permanência do ecossistema de software móvel. Por outro lado, a necessidade de equipamentos específicos para desenvolvimento de aplicativos contribui de forma negativa para a comunidade, pois tende a diminuir seu contingente.



Figura 14 Relações entre diferentes os diferentes aspectos identificados.

No decorrer desta seção, será discutido cada aspecto positivo, negativo e ambíguo, bem como as explicações das relações apresentadas na Figura 14.

Dos pontos comuns que foram encontrados em ambos os resultados, o mais comum refere-se ao aspecto ambíguo **FRAGMENTAÇÃO** do ecossistema de software.

A fragmentação no contexto deste estudo refere-se a diferentes dispositivos de hardware com base no mesmo sistema operacional mas com atributos diferentes, disponíveis no mercado. Esses atributos podem ser: capacidade de memória interna, velocidade da unidade de processamento central (CPU), resolução, tamanhos de tela, versão do sistema operacional, entre outros. Um exemplo famoso de fragmentação são os dispositivos de hardware baseados no sistema operacional Android, uma vez que várias empresas lançam seus diferentes modelos de *smartphones* e *tablets* (hardware) com diferentes versões do sistema operacional (software) e passam a atualizar com grande atraso, deixando a versão do sistema desatualizada.

A fragmentação da plataforma é um aspecto negativo por dificultar o desenvolvimento de aplicativos em diversas perspectivas (diferentes tamanhos de interface, hardware, versões do sistema operacional, etc.). Portanto, quanto maior for a fragmentação maior terá que ser dada atenção para cuidados de **INTERFACE** com o usuário e **RECURSOS DE HARDWARE**. Os resultados apontam que um ecossistema altamente fragmentado permite que cada engenheiro de software faça o que desejar, ou seja, seguem seus próprios padrões e desenvolvem seus aplicativos do seu modo. O que acaba afetando a beleza da interface deixando com um ar um pouco de amadorismo. Não que isso seja culpa do ecossistema de software, mas sim da falta de controle que ele tem sobre seus parceiros e contribuidores. É importante observar que o aspecto **INTERFACE** com usuários poderia ter sido discutido como um aspecto positivo também, afinal o engenheiro de software não fica preso à interface padrão do sistema. Porém, ninguém relatou isto como vantagem, apenas apontado como um problema. Outro aspecto negativo é que uma vasta gama de dispositivos gera uma maior dificuldade aos engenheiros de software em otimizar um aplicativo é tarefa árdua, pois existem diferentes componentes de hardware (**RECURSOS DE HARDWARE**). Por exemplo, algum jogo ou aplicativo não funciona como deveria em alguns modelos.

Assim, os engenheiros de software, para garantir a satisfação de uma grande quantidade de usuários finais, devem necessariamente verificar manualmente que as funcionalidades do aplicativo são preservadas em diferentes dispositivos de hardware. Quando se trata de aplicações móveis, os usuários geralmente possuem pouca

paciência para um desempenho lento. De acordo com (Rapoza, 2012), “vinte e cinco por cento dos usuários abandonam um aplicativo móvel depois de três segundos de atraso”.

Outra característica comum aos dois resultados trata-se dos **EMULADORES** que foram bastante criticados por não apresentarem as mesmas qualidades durante testes em dispositivos reais. Novamente, embora os emuladores possam apresentar características positivas, por exemplo, não representam custo financeiro, mas nenhum dos entrevistados ou respondentes suportaram esta afirmação. Na verdade os emuladores foram relatados unicamente com aspectos negativos. Por exemplo, emuladores não possuem habilidade em testar para diferentes ecossistemas software e dispositivos de hardware (**FRAGMENTAÇÃO**), não conseguem emular interoperabilidade das redes de um dispositivo real, etc. Isto significa que mesmo que o aplicativo se comporte bem com o uso de emuladores, não se pode afirmar que realmente funcionará em um dispositivo real.

Para evitar o cenário de problemas críticos ao lançar um aplicativo com defeitos, assim evitar este problema significa que os desenvolvedores devem executar a maior parte dos testes em **EQUIPAMENTOS DE HARWARE ESPECÍFICOS** reais antes da disponibilidade comercial, pois como visto anteriormente, a percepção dos usuários finais quanto ao desempenho do aplicativo é de grande ajuda para uma maior permanência ou adoção por parte de novos usuários. É importante deixar claro a relação entre a fragmentação e emuladores. Os emuladores perdem sentido quando o ecossistema é muito fragmentado, pois os emuladores não conseguem representar a enorme quantidade de dispositivos no mercado. Portanto, quanto maior é a fragmentação, maior é a dificuldade para construir novos aplicativos, pois o processo de testes se torna muito longo, repetitivo e cansativo. Desta forma, prejudicando o rápido desenvolvimento de aplicativo, é justamente por isso que é apresentado uma correlação negativa entre fragmentação e tempo para construção de aplicativos.

Vocell (2013) relata que um aplicativo móvel deve ser rápido e funcional, a fim de oferecer a melhor e mais agradável experiência possível ao usuário. Para tanto, lista alguns fatores em que desenvolvedores devem levar em consideração, tais como: evitar muitas requisições HTTP, *download* de grandes imagens, neste caso as imagens devem ser utilizadas com base no tipo de dispositivo do usuário, ou seja, adequando a

dimensão para o tamanho da tela desse dispositivo, *download* de informações limitadas para obter uma maior performance, etc.

Quando uma aplicação não é devidamente preparada (**INTERFACE e RECURSOS DE HARDWARE**) para se comportar em diferentes dispositivos de hardware, a fragmentação pode influenciar negativamente na opinião dos usuários, o que conseqüentemente pode interferir na adoção por parte de novos usuários. Por exemplo, um aplicativo não compatível com determinado dispositivo poderá gerar comentários negativos a respeito do aplicativo em questão, assim, influenciando negativamente a adoção por parte de novos usuários. Com isto, cria-se uma imagem positiva ou negativa não apenas para a aplicação em si, mas também para os engenheiros de software criadores da aplicação. A Figura 15 ilustra diferentes experiências de usuários para um mesmo aplicativo. Note que enquanto alguns reclamam que o aplicativo “quebra” em seus *smartphones*, outros apontam que o aplicativo é muito bom e divertido. É interessante observar na Figura 15 que os dois usuários que não aprovaram o aplicativo, também informam o modelo de seus aplicativos. Talvez por reconhecerem o grande problema da fragmentação.

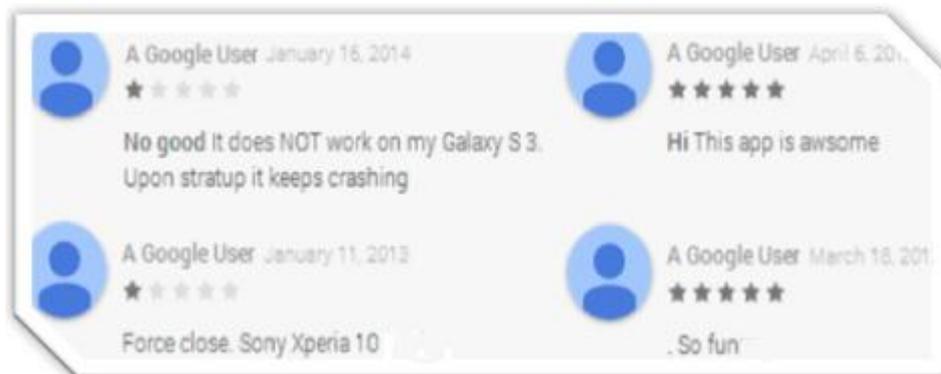


Figura 15 Diferentes experiências de usuários para um mesmo aplicativo baseado no sistema operacional Android, uma das maiores complexidades em desenvolver aplicativos segundo os resultados deste estudo.

Não é incomum que usuários da plataforma Android recebam uma mensagem de incompatibilidade ao tentar instalar algum aplicativo, se o seu dispositivo de *hardware* estiver desatualizado. A Figura 16 ilustra esta mensagem “*This app is incompatible with your device*”.

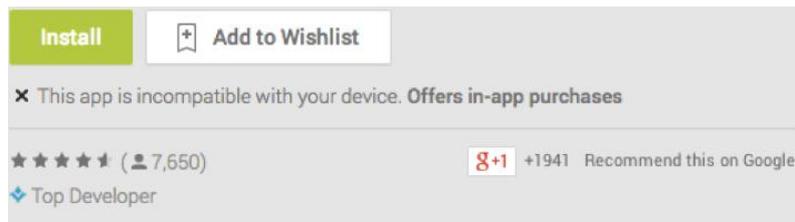


Figura 16 Mensagem que informa ao usuário que seu dispositivo de hardware é incompatível com o aplicativo.

Estudo corrobora com o que é apresentado em (Joorabchi et al, 2013), onde os autores buscam obter uma compreensão das principais práticas e desafios enfrentados pelos desenvolvedores para o desenvolvimento de aplicativos para diferentes dispositivos móveis em plataformas nativas. Realizaram um estudo exploratório, seguindo uma Abordagem da *Grounded Theory*, a mesma abordagem aplicada neste estudo. Para tanto, entrevistaram doze engenheiros de software com experiências em desenvolvimento móvel em diferentes plataformas, como Android, iOS, Windows Phone e Black Berry. Baseado nas análises das entrevistas, criaram um formulário e distribuíram para outros 188 desenvolvedores de software. Como resultado da pesquisa, revelaram que cada vez mais o desenvolvimento para plataformas móveis estão se fragmentando, isto é, quando o "mesmo" aplicativo é desenvolvido para múltiplas plataformas.

Joorabchi, et al (2013) aponta que os desenvolvedores móveis precisam de melhores ferramentas de análise para medir e monitorar seus aplicativos. Além disso, testar aplicações é um grande desafio, pois atualmente os teste são realizados manualmente por meio de celulares reais. Afirmam que o teste de unidade não são comuns dentro da comunidade móvel e frameworks de testes atuais não fornecem o mesmo nível de suporte para diferentes plataformas. Outro ponto interessante revelado, e que converge com que é apresento neste estudo, é que emuladores parecem não ter várias características reais de dispositivos móveis, tais como a mobilidade (por exemplo, mudar de rede conectividade), serviços de localização, sensores, ou gestos diferentes e insumos, o que faz análises e testes ainda mais desafiadora.

Conforme já mencionado, a fragmentação apresenta desvantagens e vantagens. Nos parágrafos acima, foi discutido os aspectos negativos da fragmentação e sua relação com alguns aspectos (Interface, Emuladores, Recursos de Hardware e Tempo de Desenvolvimento). Observe que são exatamente as relações negativas identificadas da

Figura 14. No parágrafo seguinte, é descrito em que aspecto a fragmentação é positiva, de acordo com os resultados apresentados neste estudo.

A grande vantagem da fragmentação é justamente a existência de uma infinidade de combinações de hardware e software possíveis à escolha do consumidor. A liberdade oferecida pelo ecossistema de software permite que surja uma grande quantidade de fabricantes, como é o caso do ecossistema de software Android que conta com a presença de diferentes fabricantes, tais como Samsung, Motorola, Sony, HTC, etc. Com o aumento da competição, o preço dos dispositivos de hardware caem, o que ajuda ao ecossistema de software obter uma grande quantidade de usuários finais, fortalecendo o mercado. Desta forma, os usuários finais conseguem comprar ou optar por smartphones de diferentes tamanhos, resoluções, etc. Variando entre hardware simples a hardware poderosos com recursos inovadores. De acordo com os resultados apresentados pelos entrevistados (primeira fase da coleta de dados) esse talvez seja o fator dominante que tornou o sistema tão **POPULAR**. A popularidade considerada neste contexto reflete o grau relativo ao número de indivíduos/usuários finais que utilizam a plataforma. Ela pode ser considerada um aspecto positivo para adoção da plataforma uma vez que o sucesso de uma aplicação depende de um grande número de usuários finais.

Para se ter uma noção sobre os atuais ecossistemas de software móveis quanto a popularidade, a Figura 17 apresenta o quão significante é a diferença entre os ecossistemas. Novamente, a fragmentação do ecossistema de software Android é talvez a maior responsável pela sua popularidade, que apesar de ter entrado no mercado um pouco depois do iOS (VisionMobile, 2013).

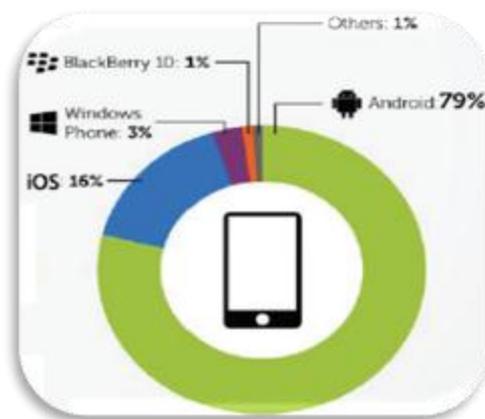


Figura 17 Smartphone Market Share (Fonte: Developer Economics. State of The Developer Nation Q3 2014)

A Figura 18 apresenta a porcentagem de cada continente em relação a preferência de plataformas por parte dos desenvolvedores de aplicativos móveis. Ao observar o continente sul-americano, percebe-se que o ecossistema Android é mais utilizado em detrimento de outros, porém tal realidade não é condizente com três continentes: América do Norte, Europa e Oceania. Portanto, a realidade observada quanto ao aspecto popularidade apontada no parágrafo anterior pode influenciar, mas depende do contexto em que o engenheiro de software se encontra.

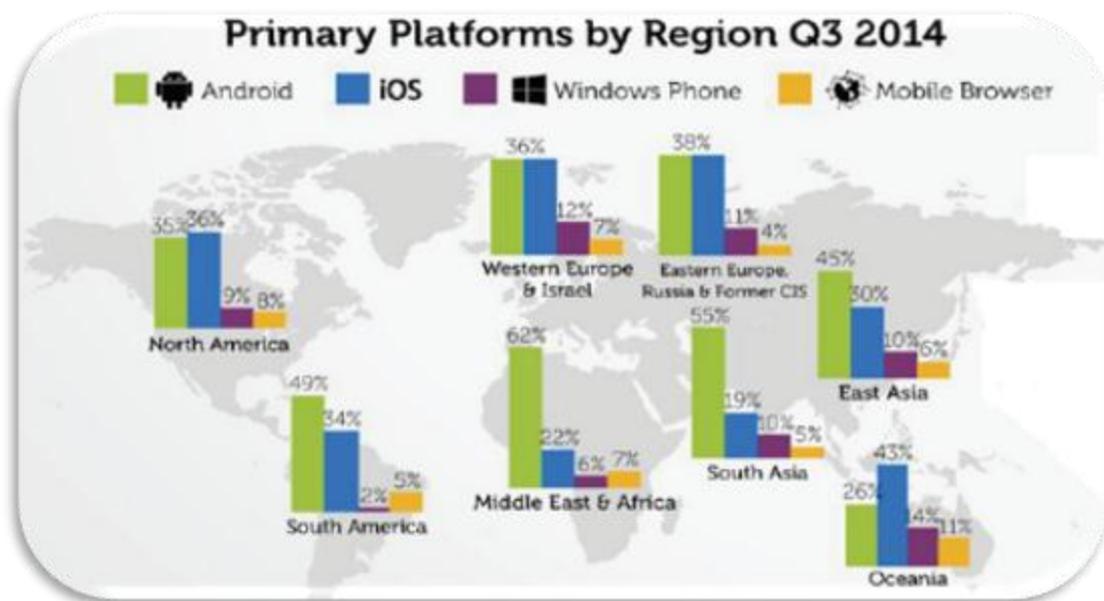


Figura 18 Principais plataformas utilizadas em diferentes continentes (Fonte: Developer Economics. State of The Developer Nation Q3 2014).

Apesar de ecossistemas de software altamente fragmentados serem mais populares e gerarem mais downloads para os aplicativos, isso não necessariamente reflete na **RENTABILIDADE** final dos mesmos, e conseqüentemente, na rentabilidade dos engenheiros de software. Porém, a rentabilidade está diretamente relacionada a adoção de engenheiros de software. A rentabilidade é definida neste contexto como o retorno financeiro esperado por um engenheiro de software a partir de um aplicativo. Este retorno pode ser obtido de diferentes maneiras, tais como: pagamento via *downloads* da aplicação, ou publicidade, por exemplo, utilizando o sistema Admob (Google) ou iAd (Apple).

Os resultados quanto ao aspecto da rentabilidade neste estudo são ambíguos, não fazendo relação nenhuma com fragmentação e muito menos com o número de usuários

disponíveis (**POPULARIDADE**). Alguns entrevistados afirmam que apesar da popularidade, e número de downloads para o ecossistema de software Android, a rentabilidade não segue em uma mesma proporção, eles sugerem que isto se deve ao fato de que há menos usuários dispostos a pagar por aplicativos, fazendo com que a média de ganhos por *download* seja a menor.

A partir da análise dos dados das entrevistas pôde-se concluir que a rentabilidade pode ser considerada como um aspecto importante que influencia na adoção de uma determinada plataforma. Embora o conjunto de dados das entrevistas apresentem aspectos tanto positivos quanto negativos em relação a este aspecto, no resultado da coleta de dados mediante questionário, não ocorreram relatos sobre aspectos positivos sobre isso, mas sim apenas relatos negativos. Porém, a análise desta discussão considerou a rentabilidade como um aspecto ambíguo. Vale ressaltar que o questionário aplicado na segunda fase do levantamento de dados não perguntava diretamente sobre o aspecto da rentabilidade, diferentemente do que foi feito nas entrevistas.

Muitos engenheiros de software acreditam no desenvolvimento de aplicativos para ecossistemas de software móveis como uma nova oportunidade de trabalho em um mercado relativamente novo. Além disso, é importante mencionar que para o crescimento de um ecossistema, todos os atores envolvidos devem ganhar, pois do contrário, o ecossistema não se mantém (Hoch, 2000). Isto foi percebido em ambas análises, tanto das entrevistas como dos questionários. Resumindo, quanto melhor for a rentabilidade maior será a comunidade de desenvolvedores, potencializando o item de engajamento da **COMUNIDADE**.

Alguns autores (e.g. Papas, 2013; Silito & Begel, 2013), buscam compreender o que motiva uma pessoa a desenvolver para um ecossistema de software móvel em específico. Os resultados convergem com os que são apresentados acima. Por exemplo, ao identificar que há, sem dúvida, uma forte correlação entre o valor da receita gerada e com a adoção de engenheiros de software.

O trabalho de Koch & Kerschbaum (2014) é um dos poucos que se concentra nas razões que levam os desenvolvedores de software a adotarem um ecossistemas de software específico. Estes autores relatam que a maior motivação para os desenvolvedores é a experiência de diversão e estímulo intelectual e o processo de aprendizagem de novas habilidades. Diferentemente do que é apresentado nesta

dissertação, a rentabilidade no desenvolvimento de aplicativos de um ecossistema de software é um fator que menos motiva estes desenvolvedores.

De acordo com os resultados, **EQUIPAMENTOS ESPECÍFICOS** são dispositivos de *hardware* necessários ao desenvolvimento de aplicativos para uma determinada plataforma, como por exemplo, para a criação de um aplicativo para o ecossistema de software iOS é necessário que se tenha um Macintosh (nome dos computadores pessoais fabricados e comercializados pela Apple). Neste sentido, quanto maior a necessidade de máquinas/equipamentos específicos para o desenvolvimento, menor será a adoção de novos engenheiros de software, diminuindo a participação fica evidente que este aspecto é prejudicial a **COMUNIDADE**. Isto é ainda mais agravado se o custo destes equipamentos for alto. A necessidade de obtenção de equipamentos específicos de hardware para o desenvolvimento de aplicações pode ser crucial para que novos desenvolvedores adotem a plataforma, principalmente para quem está começando ou aqueles com baixo poder aquisitivo. Isto reflete muito bem ao trabalho de Rogers (2013), indica como que tal necessidade é prejudicial a adoção de tecnologia, ele classifica que a utilização de equipamentos específico causa uma menor testabilidade e menor compatibilidade.

A necessidade em adquirir equipamentos de hardware como um aspecto negativo tem o viés da pesquisa ter sido feita na primeira fase apenas com brasileiros, sendo uma realidade que pode não condizer com demais países. Este, é uma das justificativas da utilização de um questionário eletrônico na segunda fase, assim pode-se obter opiniões de engenheiros de diferentes partes do mundo. Isto significa que enquanto a taxa de adoção depende do contexto de cada engenheiro de software, pois pouquíssimos relatos foram observados sobre este aspecto na segunda fase global.

A compatibilidade ou familiaridade de **LINGUAGENS DE PROGRAMAÇÃO** e **AMBIENTE DE DESENVOLVIMENTO** é um aspecto positivo para adoção, permanência do ecossistema de software uma vez que proporciona maior facilidade para o desenvolvimento de aplicativos. Os resultados mostraram que para muitos o ecossistema de software móvel iOS, que utiliza como linguagem de programação o Objective-C e o Xcode como ambiente integrado de desenvolvimento, não se mostraram familiar ou compatíveis tecnologicamente com a maioria dos entrevistados, já a combinação entre a linguagem de programação Java e Eclipse foi

relatada como mais família e que por isso se tornaram aspectos positivos a adoção, permanência no ecossistema de software Android. Conforme relato dos entrevistados, isto ocorre devido ao fato das universidades lecionarem disciplinas de programação utilizando com mais frequência a linguagem Java, por exemplo. Novamente, o contexto geral no qual um engenheiro de software está incorporado influencia sua decisão.

Os dois parágrafos acima convergem com o que é apresentado em (Goadrich & Rogers 2011), em que os autores apresentam uma análise comparativa de desenvolvimento entre as plataformas móveis Android e iOS. O objetivo é identificar qual a plataforma o setor acadêmico deve ensinar para alunos de graduação. Neste comparativo os autores levam em consideração requisitos de *software* e *hardware* de ambas as plataformas, por exemplo, interface gráfica, documentação, *software* Development Kit, a facilidade em desenvolver um simples “*Hello Word*”, etc. Como conclusão identificaram que o desenvolvimento de aplicativos para iOS em relação ao Android pode ser mais difícil devido a necessidade de *hardware* e ferramentas específicas, além disso apontam que a linguagem de programação para o iOS é menos comum. Enquanto que para o Android com qualquer computador modesto é possível seu desenvolvimento, e afirmam que estudantes estão mais acostumados com a combinação Java e eclipse que objetivo C e Xcode. Entretanto, o artigo afirma que conhecer novas linguagens de programação é fundamental para o ensino, assim, cabe ao setor acadêmico decidir qual plataforma melhor atende a suas necessidades.

A **COMUNIDADE** é definido como um conjunto de unidades (indivíduos, grupos informais, organizações e/ou subsistemas) inter-relacionadas que estão engajadas para resolver um problema e atingir um objetivo comum. De acordo com os resultados apresentados no capítulo anterior, os ecossistemas para qual os entrevistados desenvolvem aplicativos possuem uma grande participação da comunidade, seja para difusão de novas ideias ou sobre o soluções para problemas em comum. Além disso, os entrevistados relatam que ao participar efetivamente de comunidades, ajudará no processo de adoção da plataforma. Afirmam ainda que é de extrema importância que participem dessas comunidades para diminuir a curva de aprendizado e que este aprendizado é por vezes mais eficiente do que por meios mais tradicionais como livros e revistas. Poder se atualizar quanto as melhorias e inovações para o ecossistema de software também foi apontado como um aspecto positivo. A comunidade é representada por diferente níveis de experiência, buscam e compartilhar informações, desde

problemas mais básicos (iniciantes) quanto para problemas mais avançados (experientes). Este é portanto considerado um fator social.

Os resultados sugerem que quanto maior for a comunidade de desenvolvimento, mais será a disponibilidade de documentações. Isto reflete no tempo de desenvolvimento para construção de aplicativos, se de um lado observa-se a fragmentação como uma barreira para o tempo de desenvolvimento, por outro a comunidade é um aspecto positivo a isto. Em ambas as fases de coleta e análise de dados, os resultados mostram que o tempo de resposta para uma pergunta feita em fóruns como Stack Overflow é muito rápida, é como se existisse alguém esperando a pergunta, e que já possuísse a resposta.

Outro aspecto interessante que este trabalho atribui a comunidade, é que durante análise das entrevistas, pôde-se perceber a figura dos líderes de opinião. São estes líderes que influenciam a decisão dos outros engenheiros de software quando adoção, fornecendo conhecimento, informações, conselhos, etc. Desta forma, os indivíduos seguem os líderes que são percebidos como tecnicamente mais competentes que eles próprios (Amorim, 1999). Em analogia a isto temos o exemplo do craque de futebol que é um formador de opinião e influencia as crianças a serem torcedores do time para o qual ele joga.

Portanto, este resultado converge com o que é apresentado em (Koch & Kerschbaum, 2014), onde os autores buscam compreender se o número de desenvolvedores para determinado ecossistema móvel é um fator significativo para adoção de novos engenheiros de software, entre outras hipóteses.

A **DISTRIBUIÇÃO** de aplicativos refere-se as diferentes formas que os desenvolvedores possuem para disponibilizar seus aplicativos aos usuários finais. Conforme apresentado na Figura 14, quanto menos burocrático é a distribuição de aplicativos para os engenheiros de software, maior será a satisfação dos usuários finais.

Atualmente, a principal forma de distribuir um aplicativo é submetendo-o para uma App Store. A App Store é então o intermediário entre o desenvolvedor e o usuário final. Ela oferece um ambiente de distribuição onde o desenvolvedor pode, dentre outros (i) obter informações sobre o número de usuários que instalaram sua aplicação, de que país eles são, (ii) ler comentários dos usuários e (iii) receber os valores referentes à

venda do aplicativo, caso o mesmo seja pago. Os comentários dos usuários na App Store podem fornecer informações bastante úteis para os desenvolvedores, como falhas e pedidos de novas funcionalidades. Além disso, usar a App Store para buscar e fazer download de aplicativos é a maneira mais segura para os usuários, pois a plataforma avalia a aplicação antes de fato disponibilizar a aplicação.

Para alguns ecossistemas de software móvel, como o iOS, a única maneira de publicar um aplicativo é via *App Store*. Já nos outros ecossistemas, como o Android, não são tão restritivos: é possível instalar aplicações em dispositivos reais sem ter que publicar a aplicação na Play Store. Neste caso, basta que o desenvolvedor gere um executável de sua aplicação e distribua, por exemplo, via e-mail, pen-drive, ou disponibilize em algum site próprio na internet.

Conforme dito por vários entrevistados, para disponibilizar uma aplicação na loja virtual do Android (Google Play) o desenvolvedor precisa pagar uma taxa única de USD \$25 (vinte e cinco dólares), enquanto que o iOS não possui tanta liberdade. Para que o desenvolvedor possa distribuir sua aplicação, seja para testes ou publicação na *App Store* (Loja Virtual de Aplicativos da Apple), é necessário que o desenvolvedor crie uma conta e adquira uma licença para desenvolvimento que varia de USD \$99 à USD \$299 anuais, dependendo do tipo da conta. O processo para que um aplicativo seja aceito na Apple Store é mais burocrático por ter mais cuidados quanto a verificação da existência de possíveis *malwares* ou a análise da aderência aos padrões de interface da Apple. Isto precisa ser feito antes de disponibilizar os aplicativos na Apple Store, tornando o processo mais longo e burocrático.

Como discutido anteriormente, a fragmentação é algo desafiador para o desenvolvimento de aplicativos para dispositivos móveis. Tal situação encarece e complexa o desenvolvimento de uma aplicação que venha a atender a todos os dispositivos, pois é necessário conhecimentos específicos de cada plataforma, e também que seja desenvolvida uma aplicação diferente para cada plataforma. Neste contexto surgem as aplicações híbridas ou ditas **MULTIPLATAFORMAS**, em que se desenvolve apenas uma aplicação, e esta pode ser utilizada em vários dispositivos com diferentes sistemas operacionais. Isso é possível a partir da utilização de *frameworks* para desenvolvimento, que são responsáveis por empacotar o código-fonte para as diferentes plataformas (Android, iOS, Windows Phone, etc.), permitindo que elas sejam

instaladas no dispositivo, e possam acessar os seus recursos, como câmera, GPS e contatos. Por exemplo, o Ionic⁵ que é um framework criado no final de 2013 que visa a criação de aplicações híbridas para dispositivos móveis. Ele nada mais é do que uma pilha de componentes e outros frameworks. A possibilidade de criação para vários ecossistemas de software móvel por apenas um código fonte é apontada como um aspecto positivo para adoção de ecossistemas de software.

5.4 Limitações

Como em qualquer estudo empírico, esta dissertação possui limitações. Por exemplo, a maior parte dos entrevistados desenvolviam aplicativos para as plataformas Android ou iOS. Assim, os resultados encontrados neste trabalho podem não representar a total realidade enfrentada durante o processo de criação de aplicações, ou seja, muitos outros aspectos podem ter sido desconsiderados. Por exemplo, esta dissertação não obteve resultados expressados por desenvolvedores de ecossistemas de software tais como Windows Phone, Blackberry ou outras. De maneira similar, apesar dos resultados terem sido classificados de acordo com as experiências dos entrevistados, não foi realizado nenhuma separação ou distinção clara sobre os perfis dos engenheiros de software. Além disso, características como as idades ou regiões dos mesmos não foram utilizadas para compreensão de opinião de acordo de faixas etárias ou regiões. Sendo que estes fatores podem influenciar no resultados obtidos. Desta forma, dependendo do contexto do informante este resultado ou opinião pode ser diferente.

O número baixo de entrevistas (limitação) realizado na primeira fase, a local, para que fosse compreendido a totalidades de aspectos positivos ou negativos. É importante deixar claro também, que as entrevistas tiveram como origem o objetivo entender os motivos que levam um engenheiro de software a adotar, ou não, um determinado ecossistema de software. Apesar disso, esta dissertação utilizou a análise para a identificação dos aspectos positivos e negativos.

⁵ <http://ionicframework.com/>

6. CONCLUSÕES

6.1 Visão Geral

Esta dissertação descreveu um estudo qualitativo sobre aspectos positivos e negativos que influenciam na construção de aplicativos para dispositivos móveis por engenheiros de software. Os principais aspectos que podem ser relacionados a adoção, rejeição ou permanência em ecossistemas de software móveis foram identificados. Por exemplo, identificou-se que um dos principais desafios a ser tratado vem a ser a fragmentação, embora a fragmentação se apresente como um aspecto positivo, ou seja, aspectos ambíguos foram identificados, dependendo do objetivo o aspecto pode ser encarado negativamente ou positivamente. Esta dissertação também descreveu diferentes aspectos relacionados ao entendimento do que leva um desenvolvedor a adotar um ecossistema de software móvel, bem como os principais desafios na área para o desenvolvimento de aplicativos móveis, o que foi possível perceber que há uma grande semelhança entre os aspectos negativos considerados para a adoção de um ecossistema de software com os principais desafios revelados, entre outros. Este último capítulo traz primeiramente as considerações finais sobre este assunto, descrevendo depois as contribuições desta pesquisa e os trabalhos futuros.

6.2 Contribuições do trabalho

Esta dissertação contribui ao apresentar os principais aspectos positivos e negativos para a adoção de ecossistemas de software. Muito mais que isso, a identificação de aspectos chamados ambíguos, ou seja, que são positivos e negativos ao mesmo tempo. A fragmentação foi revelada como o maior problema, se mostrou um dos aspectos mais desafiadores para o desenvolvimento de aplicativos móveis. Nos resultados das duas fases desta dissertação a fragmentação se mostrou como principal característica negativa para um ecossistema de software móvel. Quando o "mesmo" app é desenvolvido para múltiplas plataformas, os desenvolvedores devem tratar o aplicativo para cada plataforma separadamente e realizar testes separadamente para confirmar que a funcionalidade é preservada em todas as plataformas.

Este estudo contribui ao identificar os relacionamentos entre os aspectos positivos e negativos. Por exemplo, quanto maior for a fragmentação do ecossistema de software móvel, maior será a adoção por parte de novos usuários finais, porém maiores

problemas técnicos de interface com o usuário, recursos, etc. Este relacionamento é expresso graficamente na Figura 14.

Os resultados relatados apresentados nesta dissertação também podem servir para as empresas governantes dos ecossistemas (Google, Apple, etc.) planejarem melhor suas estratégias de adoção e conseguirem aumentar sua base de engenheiros de software e usuários finais para o seu ecossistema.

6.3 Trabalhos Futuros

A quantidade de entrevistas realizadas neste exame de dissertação pode ter prejudicado a formação de conclusões, pois o número de entrevistados foi relativamente baixo, nove entrevistas exatamente. Além disso, todos os entrevistados eram limitados ao desenvolvimento de aplicativos para Android ou iOS. Em um trabalho futuro, é interessante que o número de desenvolvedores entrevistados aumente, bem como aumentar a gama de plataformas na pesquisa, por exemplo, Firefox OS e Windows Phone. Além disso, classificar os entrevistados quanto suas experiências, com isso possíveis lacunas entre o alinhamento da teoria com os dados obtidos podem ser analisadas. Também é interessante tentar voltar com os entrevistados deste estudo para obter mais informações de aspectos que podem não ter sido tão bem explorados, por exemplo, em nenhum momento foram feitas perguntas diretas a respeito do quanto a influência de amigos ou líderes de opinião ajuda na aceitação de tecnologias. Apesar de tudo tender que sim, é importante que isto seja observado a partir de novas entrevistas. Ou ainda, o que faz um ecossistema de software móvel ser mais rentável que outros, por exemplo.

No que se refere o conjunto de dados dos questionários, é interessante que se aprimore o questionário para que ele apresente questões que envolvem mais os aspectos sociais, pois a maioria dos aspectos observados são técnicos.

Esta dissertação levou em consideração aspectos positivos e negativos, embora tais aspectos possam ser compreensíveis para adoção, rejeição ou permanência de engenheiros de software, é interessante a aplicação de *frameworks* que tratam deste tema. Por exemplo, a teoria da difusão de inovações de Roger poderia explicar como engenheiros de software percebem determinado ecossistema de software móvel levando em consideração seus pilares tais como as características de uma inovação. O

CERTICS⁶ que é um modelo que tem vários critérios para definir inovação, possui avaliação técnica, de mercado, operacionais e de inovação. Outro exemplo muito utilizado para explicar aceitação da tecnologia é o “*Technology Acceptance Model*” (TAM) ou em português “modelo de aceitação de tecnologia”. A ideia do TAM é que a intenção da pessoa para utilizar uma nova tecnologia é baseada na facilidade de uso e na utilidade que o usuário viu nessa tecnologia.

⁶ www.certics.cti.gov.br

REFERÊNCIAS

- ANDRADE, R., ALVES, C. and ALBUQUERQUE, A. A Model Proposal for Characterizing Software Ecosystems. Centro de Informática – Universidade Federal de Pernambuco, 2011.
- AMORIM, M., Estratégias para difusão de um ambiente virtual para comércio eletrônico via internet : um estudo de caso - CIMM: Centro de Informação Metal Mecânica. 1999 (Mestrado em Engenharia de Produção e Sistemas) – Universidade Federal de Santa Catarina, Florianópolis.
- BANKS, S., LOUIE, E., EINERSON, M. Constructing Personal Identities in Holiday Letters. *Journal of Social and Personal Relationships*, Vol. 17(3): 299–327, 2000.
- BARBOSA, O., SANTOS, R.P., ALVES, C., WERNER, C., JANSEN, S. A Systematic Mapping Study on Software Ecosystems through a Threedimensional Perspective. In: JANSEN, S., CUSUMANO, M. & BRINKKEMPER, S. (eds.), *Software Ecosystems: Analyzing and Managing Business Networks in the Software Industry*. Cheltenham, UK, and Northampton, MA, USA, Edward Elgar Publishing, 2013, p. 59-81.
- BARBOSA, O., ALVES, C. A Systematic Mapping Study on Software Ecosystems. In: *Proceedings of the 3rd International Workshop on Software Ecosystems, 2nd International Conference on Software Business*. 2011, p. 15-26.
- BOSCH, J, From Software Product Lines to Software Ecosystem, In: *Proceedings of 13th International Software Product Line Conference*, San Francisco, CA, USA, August, p. 1-10, 2009.
- BOSCH, J. & BOSCH-SIJTSEMA, P. From Integration to Composition: On the Impact of Software Product Lines, Global Development and Ecosystems, In: *Journal of Systems and Software*. Vol. 83, 2010, p. 67-76.
- BUTLE, M. Android: Changing the Mobile Landscape. In: *IEEE Vol. 10*. 2010, p. 4-7.

- CATALDO, M., HERBSLEB, J.D. Architecting in Software Ecosystems: Interface Translucence as an Enabler for Scalable Collaboration. In: Proceedings of the 4th European Conference on Software Architecture, 2nd International Workshop on Software Ecosystems. 2010, pp. 65-72.
- CAMPBELL, P.R.J., AHMED, F. A Three-dimensional View of Software Ecosystems. In: Proceedings of the 4th European Conference on Software Architecture, 2nd International Workshop on Software Ecosystems. 2010, p. 81-84.
- CUSUMANO, M. How platform adoption can be an important determinant of product and technological success. In: Communications of the ACM, Vol. 53 No. 1, 2010, p. 32-34
- CUSUMANO, M.A. & GAWER, A. The elements of platform leadership, MIT Sloan Management Review, 2002, p. 51-58.
- DHUNGANA, D., GROHER, I., SCHLUDERMANN, E., BIFFL, S. Software Ecosystems vs. Natural Ecosystems: Learning from the Ingenious Mind of Nature. In: Proceedings of the 4th European Conference on Software Architecture, 2nd International Workshop on Software Ecosystems. 2010, p. 96-102.
- GIL, A. Métodos e Técnicas de Pesquisa Social. 5. ed. São Paulo: Atlas, 1999.
- GODOY, S. Introdução à Pesquisa Qualitativa e suas Possibilidades In Revista de Administração de Empresas, vol. 35, 1995.
- GLASER, B.G. & STRAUSS, A.L. The Discovery of Grounded Theory: Strategies for Qualitative Research. Chicago: Aldine Pub. Co. 1967.
- GOADRICH, M., & ROGERS, M. Smart smartphone development: iOS versus android. In: SIGCSE '11 Proceedings of the 42nd ACM technical symposium on Computer science education. 2011, p. 607-612.
- HANSEN, G.K. A Longitudinal Case Study of an Emerging Software Ecosystem: Implications for Practice and Theory. In: Journal of Systems and Software, Vol. 85, 2012, p. 1455-1466.

- HANSSEN, G.K., DYBA, T. Theoretical Foundations of Software Ecosystems. In: Proceedings of the 4th International Workshop on Software Ecosystems, 3rd International Conference on Software Business. 2012, p. 6-17.
- HOCH, D. Secrets of Software Success: Management Insights from 100 Software Firms around the World. Harvard Business School Press. 2000.
- IANSITI, M., LEVIEN, R. The Keystone Advantage: What the New Dynamics of Business Ecosystems Mean for Strategy, Innovation, and Sustainability. In: Harvard Business Review Press, 2004.
- INGEN, K., OMMEN, J., JANSEN, S. Improving activity in communities of practice through software release management. In: MEDES '11 Proceedings of the International Conference on Management of Emergent Digital EcoSystems. 2011, p. 94-98.
- JANSEN, S., BRINKKEMPER, S., FINKELSTEIN, A. Business Network Management as a Survival Strategy: A Tale of Two Software Ecosystems. In: Proceedings of the First International Workshop on Software Ecosystems, 11th International Conference on Software Reuse. 2009, p. 34-48.
- JANSEN, S., CUSUMANO, M., BRINKKEMPER, S. (eds.), 2013, Software Ecosystems: Analyzing and Managing Business Networks in the Software Industry. Cheltenham, UK, and Northampton, MA, USA, Edward Elgar Publishing.
- JOORABCHI, M.E, MESBAH, A., KRUCHTEN, P. Real Challenges in Mobile App Development. In: Empirical Software Engineering and Measurement. ACM/IEEE International Symposium. 2013, p. 15-24.
- KOCH, S., & KERSCHBAUM, M. Joining a smartphone ecosystem: Application developers' motivations and decision criteria. In: Information and Software Technology, 2014, p. 1423-1435.
- LUNGU, M. Towards reverse engineering software ecosystems. In: in: IEEE International Conference on Software Maintenance, ICSM. 2008, p. 428–431.

- MANSFIELD-DEVINE, S. Android Architecture Attacking The Weak Points. In: Network Security, 2012, p.5-12.
- MANIKAS, K., HANSEN, K.M. Software Ecosystems – A Systematic Literature Review. In: Journal of Systems and Software, Vol. 86, 2013, p. 1294-1306.
- MATTAR, F. N. Pesquisa de marketing: edição compacta. São Paulo: Atlas, 1996.
- MARCONI, M. de A.; LAKATOS, E. M. Fundamentos de metodologia científica. 5. ed. São Paulo: Atlas, 2003.
- MESSERSCHMITT, D.G., SZYPERSKI, C., Software Ecosystem: Understanding an Indispensable Technology and Industry. 1. ed., The MIT Press, 2003.
- MOORE, J. F. The death of competition: leadership and strategy in the age of business ecosystems: HarperBusiness New York. 1996.
- NEVES J, L., Pesquisa qualitativa características, usos e possibilidades Caderno de pesquisa em administração São Paulo v1 1996.
- ODUM, E. & BARRETT, G. Fundamentals of Ecology 5th Edition. Cengage, 2005.
- PAPPAS A., “How do developers prioritise platforms? iOS vs Android vs HTML5” (2013). Disponível em: <<http://www.visionmobile.com/blog/2013/12/developers-prioritise-platforms-ios-vs-android-vs-html5/>> Acesso em 20 de Outubro de 2015.
- PINSONNEAULT, A. e KRAEMER, K. L. Survey research in management information systems: an assesment. Journal of Management Information System, 1993.
- RAPOZA, J., First Class Mobile Application Performance Management. 2012 Disponível em: <<http://info.soasta.com/rs/soasta/images/aberdeen-first-class-mobile-application-performance-management.pdf>> Acesso em 05 de Setembro de 2015.
- RICHER, F. Android Dominates Outside of the United States, In: Statista, (2012). Retrieved January 2nd, 2014.

- ROGERS, Everret M. Diffusion of Innovations. 5ª. Ed. New York: Free Press, 2003.
- SANTOS, L., Entrevista: um instrumento de coleta de dados em pesquisa teórico-empírica. 2005. Disponível em: <<http://www.lcsantos.pro.br/blog.php>> Acesso em 30 de Outubro de 2014.
- STRAUSS, A., CORBIN, J. Pesquisa Qualitativa: Técnicas e Procedimentos para o Desenvolvimento de Teoria Fundamentada. Artmed, 2008.
- SEAMAN, C. Qualitative Methods. In: Guide to Advanced Empirical Software Engineering. Londres: Springer-Verlag, 35-62, 2008.
- SINGER, J., SIM, S., LETHBRIDGE, T. Software Engineering Data Collection for Field Studies. In: Guide to Advanced Empirical Software Engineering. Londres: Springer-Verlag, p. 9-34, 2008.
- SILLITO, J., & BEGEL, A., App-directed learning: An exploratory study. In: The 6th International Workshop on Cooperative and Human Aspects of Software Engineering, San Francisco, CA, USA. 2013.
- VISION MOBILE, Developer Economics Q3 2013: State of the Developer Nation. Disponível em: <<http://www.visionmobile.com/product/developer-economics-q3-2013-state-of-the-developer-nation/>> Acesso em 05 de Janeiro de 2015.
- VOCELL J., Five Mobile Site Performance Challenges (and how to fix them) 2013. Disponível em: <<http://www.sitespect.com/blog/jeffrey-vocell/5-mobile-site-performance-challenges-and-how-fix-them>> Acesso em 01 de Novembro de 2015.
- YU, L., RAMASWAMY, S., BUSH, J. Software Evolvability: An Ecosystem Point of View. In Proceedings of the 3rd Workshop on Software Evolvability, 2008.

APENDICE A - ROTEIRO DAS ENTREVISTAS

- 01°. Qual seu nome, idade e formação ?
- 02°. Para quais plataformas (iOS, Android, Mozilla OS, etc.) você desenvolve ? Há quanto tempo ?
- 03°. Como você conheceu a plataforma ?
- 04°. Por que você escolheu esta plataforma ?
- 05°. O quão simples ou complicado é a utilização da plataforma ?
- 06°. Quais são os recursos necessários para desenvolver *aplicações* para a plataforma ? Por exemplo, equipamentos específico, conhecimento de alguma linguagem específica ?
- 07°. Aproximadamente, quanto tempo você levou ou leva para desenvolver *aplicações para a plataforma* ?
08. Foi fácil ou difícil para você realmente começar a desenvolver para a plataforma, em outras palavras, como foi a curva de aprendizado ?
- 09°. O que você faz para que suas aplicações sejam conhecidas ?
- 10°. Você participa de qualquer comunidade que trate de questões que estão relacionadas a plataforma ? Que tipo de comunidade é essa ?
11. Você contribui para a comunidade ? Por exemplo, respondendo perguntas de outros desenvolvedores ? Se sim, com que frequência ?
- 12°. Quanto você acredita que a participação nessas comunidades ajudou ou ajuda a desenvolver aplicações ? Qual é o tempo de resposta na comunidade para esclarecer dúvidas ?

ANEXO A – QUESTIONÁRIO

Hi there!

We are researchers from Canada and Brazil. We want to find out **how** and **why** developers create apps for **mobile platforms** -- and which problems they may encounter.

The results of our research will be ANONYMIZED and then made PUBLIC for everyone to learn from! Filling the survey should take only about 10 minutes.

This is a purely academic research project with no commercial interests. We will openly publish the results so everyone can benefit from them, but will anonymize everything before doing so. We will handle your response confidentially. If at some point during this survey you want to stop, you're free to do so without any negative consequences.

Thanks a lot for participating!

Leif Singer (University of Victoria, Canada)

Cleudson de Souza (Federal University of Pará, Brazil)

Fernando Figueira Filho (Federal University of Rio Grande do Norte, Brazil)

Your GitHub username?

This will give us some context for your responses. Delete if you're uneasy about it!

Do you develop software?

Mark all that apply.

- Yes, professionally
- Yes, non-professionally-- e.g., pet projects, tinkering, ...
- Yes, I contribute to one or more open source projects (irrespective of size)

Do you develop software targeted at MOBILE devices?

Mark all that apply.

- Yes, professionally
- Yes, non-professionally -- e.g., pet projects, tinkering, ...
- Yes, I contribute to one or more open source projects (irrespective of size)

How long have you been developing mobile apps?

- I don't really do that
- Less than a year

- 1-2 years
- 3-5 years
- More than 5 years

How many mobile apps have you developed so far?

Consider any kind of project you can remember, e.g. from pet to professional ones.

- None
- 1
- 2-3
- 4-6
- More than 6

Which platforms do you develop for?

Mark all that apply.

- Android
- iOS
- Firefox OS
- Symbian
- J2ME
- Windows Phone
- Web Applications (e.g. mobile websites, Phonegap, ...)
- Cross-platform (e.g. Xamarin, Icenium, ...)
- Other:

Have you developed a new mobile app in the last two years?

- No
- Yes, alone
- Yes, with one or more people
- Yes, both alone and as a team

Your most important mobile app

For the following questions, please think of the most important app you have developed so far.

Is it available on some app store?

- Yes
- No

Could you provide us with the link to it? or the links in case it is available in more than one app store.

It will help us with some statistics e.g., number of downloads, categorization.

What tools did you use while developing this app?

E.g., emulators, IDEs, testing tools, analytics SDKs, build tools, etc.

What did you like best about developing the app for its platform (Android, iOS, ...) -- and why?

In what aspects does this app's platform fall short? Why?

Your thoughts on mobile development in general

What is the most crucial difference between mobile development and desktop or web development?

Answer as best as you know.

What is the hardest problem that you need to cope with in mobile development?

Which problems have actually kept you from succeeding, ones that you were unable to solve?

Closing

Would you be willing to do a short Skype / Hangout call with us so we can learn more about your response?

We'll contact you to make an appointment at your convenience.

Yes, I'd be up for it!

Would you like to receive an email when we publish the results of the survey?

Yes, please!

Please leave your email address if you chose "Yes" in one of the two preceding questions.

Thanks so much for getting this far! Any questions, comments or concerns you'd like to tell us about?