UNIVERSIDADE FEDERAL DA PARAÍBA

CENTRO DE CIÊNCIAS APLICADAS A EDUCAÇÃO

DEPARTAMENTO DE CIÊNCIAS EXATAS

BACHARELADO EM SISTEMAS DE INFORMAÇÃO

TESTE DE SOFTWARE PARA APLICATIVOS ANDROID: UM MAPEAMENTO SISTEMÁTICO DA LITERATURA

LILIAN THAÍS BARROS DA SILVA

Profa. Dra. Ayla Débora Dantas de Souza Rebouças

(Orientadora)

RIO TINTO - PB

2014

LILIAN THAÍS BARROS DA SILVA

TESTE DE SOFTWARE PARA APLICATIVOS ANDROID: UM MAPEAMENTO SISTEMÁTICO DA LITERATURA

Monografia apresentada para obtenção do título de Bacharel à banca examinadora no Curso de Bacharelado em Sistemas de Informação do Centro de Ciências Aplicadas e Educação (CCAE), Campus IV da Universidade Federal da Paraíba.
Orientadora: Profa. Dra. Ayla Débora Dantas de Souza Rebouças.

RIO TINTO - PB
2014

S586t Silva, Lilian Thaís Barros da.

Teste de software para aplicativos Android: um mapeamento sistemático da literatura. / Lilian Thaís Barros da Silva. – Rio Tinto: [s.n.], 2014.

60 f.: il. –

Orientadora: Profa. Dra. Ayla Débora Dantas de Souza Rebouças.
Monografia (Graduação) – UFPB/CCAE.

1. Software - desenvolvimento. 2. Software – teste. 3. Sistema Android - software.

UFPB/BS-CCAE CDU: 004.4(043.2)

LILIAN THAÍS BARROS DA SILVA

TESTE DE SOFTWARE PARA APLICATIVOS ANDROID: UM MAPEAMENTO SISTEMÁTICO DA LITERARURA

Trabalho de Conclusão de Curso submetido ao Curso de Bacharelado em Sistemas de Informação da Universidade Federal da Paraíba, Campus IV, como parte dos requisitos necessários para obtenção do grau de BACHAREL EM SISTEMAS DE INFORMAÇÃO.

Assinatura do autor:_____

APROVADO POR:

_____

Orientadora: Profa. DSc. Ayla Débora Dantas de Souza Rebouças
Universidade Federal da Paraíba – Campus IV

_____

Prof. Msc. José Jorge Lima Dias Junior
Universidade Federal da Paraíba – Campus IV

_____

Profa. Dra. Yuska Paola Costa Aguiar
Universidade Federal da Paraíba

RIO TINTO - PB
2014

Aos amigos, colegas e professores, minha eterna gratidão por compartilharem comigo seus conhecimentos.

AGRADECIMENTOS

Agradeço primeiramente a Deus que permitiu que tudo isso acontecesse ao longo de minha vida, me dando forças e saúde para superar todos os desafios.

Aos meus familiares, em especial aos meus pais Maria de Fátima e Antonio Ferreira por acreditarem em minha capacidade, pelo cuidado, pelo apoio e pelo amor incondicional. Aos meus tios Paulo, Marlene, e Maria Ferreira por acreditarem em meu potencial e pelo apoio constante. Ao meu primo José Reginaldo pelo enorme carinho e por me compreender nos momentos em que não pude lhe dar atenção.

Ao meu querido namorado Gaoberto por estar comigo me apoiando e incentivando em todos os momentos, pela compreensão nos momentos que precisei estar ausente, e pela motivação e conselhos quando pensei que não conseguiria seguir em frente.

A todos os professores da UFPB que contribuíram para minha formação, em especial agradeço imensamente a minha orientadora Ayla Débora Rebouças que se dedicou bastante durante a orientação deste trabalho, pelos ensinamentos, pela paciência e pela grande disponibilidade em dar assistência. Ayla na verdade é sem dúvidas um referencial para mim, e admiro-a pela competência de administrar diversas atividades ao mesmo tempo. E ao meu querido professor Saulo Emmanuel pela confiança e amizade, que desde o início de minha formação acreditou em minha capacidade, é um exemplo de profissional e pessoa.

A minha grande amiga Simone Nóbrega, que na verdade é uma irmã de coração, que sempre esteve comigo em todos os momentos, me ajudando, me ouvindo, dando sugestões, que sem sombra de dúvidas é peça fundamental na realização dessa formação. As minhas grandes amigas Sherliana Leite e Aline Fernandes pelo total incentivo, pela força, pelos conselhos, pelo enorme apoio, e por me motivarem nos momentos que mais precisava e me mostrar que depois eu iria apenas colher os frutos desse empenho, que iria valer a pena toda distância, todo sofrimento e todas as renúncias. A minha amiga Giordania pelas dicas e ajudas quando precisei. E a minha amiga Keyla Nóbrega pela amizade, incentivo, generosidade e hospitalidade nos momentos que mais precisei.

A todos os colegas de classe que contribuíram direta ou indiretamente, em especial a Wamberto Ferreira, Vicente Ramos e Smith Ascari pelos ensinamentos e paciência em alguns momentos que necessitei de suas ajudas e a todas as outras pessoas que não me vieram à memória mas que contribuíram fortemente para a minha formação pessoal e profissional.

RESUMO

A quantidade de aplicativos Android tem crescido significativamente nos últimos anos, e é importante que estes sejam produzidos com qualidade. Devido a esse motivo, é importante para os indivíduos que estão envolvidos com o desenvolvimento de aplicativos Android ter uma visão de modo ágil em relação às ferramentas e técnicas existentes, as quais permitem que esses aplicativos sejam testados de forma qualificável. Uma das técnicas utilizadas para este fim é o teste de software. Este trabalho tem por objetivo apresentar os resultados e o processo de realização de um mapeamento sistemático da literatura sobre teste de software para aplicativos Android, buscando identificar os trabalhos da área, as ferramentas de teste citadas por estes trabalhos e os tipos de testes cobertos pelas ferramentas sugeridas pela literatura. Acredita-se que este mapeamento é útil principalmente para os indivíduos que estão trabalhando com o desenvolvimento de aplicativos para Android, possibilitando-lhes uma visão organizada e autocontida das muitas ferramentas e técnicas existentes para o teste de aplicativos Android. Além disso, pesquisadores que estão iniciando na área também podem se beneficiar deste trabalho, já que poderão ter uma visão geral do estado da arte e identificar áreas de pesquisa que precisam de mais investigação.

Palavras-chave: Teste de Software, Aplicativos Android, Mapeamento Sistemático da Literatura.

ABSTRACT

Several Android Applications have been quickly developed. It is important to take care of the quality of these products. One of the techniques used for this purpose is software testing. This work is intended to present the results of a Literature Mapping Study on software testing for Android applications, showing the main works in this area, the main software tools cited in the literature and the main types of tests covered by these tools. We believe that this mapping study is useful especially for individuals who are working in the development of Android apps, because it provides a quick overview of the many existing tools and techniques for testing these applications. Besides, researchers beginning to work in this area may also benefit from this work, as they will get an overview of the state of the art and it will be possible for them to analyze and identify areas of research that need further investigation.

Keywords: Software Testing, Android Apps, Mapping Study.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE QUADROS

LISTA DE GRÁFICOS

LISTA DE SIGLAS

ACM DL	*Association for Computing Machinery Digital Library*
APIs	*Application Programming Interfaces*
CPD	Centro de Processamento de Dados
GUI	*Graphical User Interface*
IEEE	*Institute of Electrical and Electronics Engineers*
OHA	*Open Handset Alliance*

SUMÀRIO

1 INTRODUÇÃO

Vivemos hoje cercados por sistemas. O mau funcionamento desses sistemas possui capacidades de acarretar prejuízos financeiros; operacionais; ou de segurança. Considerando isso, é normal que as exigências de confiabilidade e qualidade sejam cada vez maiores. Para se atingir níveis de qualidade em produtos de software, é essencial que as organizações desenvolvedoras de software adotem boas práticas de teste de software (CRESPO et al., 2009).

Segundo Bourque e Fairley (2014), o teste de software consiste na verificação do comportamento dinâmico de um programa considerando um conjunto finito de casos de teste, adequadamente selecionados a partir do domínio de execuções (geralmente infinito), contra o comportamento esperado. Para Bastos et al. (2007, p. 31), o objetivo principal do processo de teste é encontrar o maior número possível de defeitos no software.

Ao pesquisar trabalhos na literatura sobre testes, vê-se que eles podem se focar em ferramentas ou técnicas de teste. Segundo Bastos et al. (2007), um testador primeiramente precisa entender as técnicas de teste para depois entender quais ferramentas devem ser utilizadas em cada técnica. A técnica de teste é um processo que assegura o funcionamento correto de alguns aspectos do sistema ou de sua unidade. Algumas técnicas são manuais, e outras, automáticas; algumas fazem testes estáticos, e outras, os dinâmicos; algumas avaliam a estrutura do sistema, e outras, sua função. Já a ferramenta de teste é um recurso para o testador que, sozinha, é insuficiente para conduzir todo o restante do processo de teste. A seleção da ferramenta adequada é um aspecto importante e afeta a eficiência e a eficácia do teste (BASTOS et al, 2007, p. 87).

Algumas das técnicas e ferramentas de teste que vêm sendo propostas na literatura nos últimos anos têm se voltado a aplicativos para dispositivos Android. Segundo Score (2012 apud YAN et al. 2013, p. 411), dispositivos Android atualmente lideram o mercado de smartphones nos Estados Unidos e tendências similares podem ser vistas em outros países. Android também tem presença significativa em tablets (por exemplo, Google Nexus 7/10, Samsung Galaxy Tab/Note) e em dispositivos de entrega de mídia (por exemplo, Amazon Kindle Fire, Barnes & Noble Nook HD). O uso generalizado desses dispositivos móveis coloca exigências em qualidade de software (SCORE, 2012 apud YAN et al. 2013, p. 411).

Segundo Liu et al. (2014), com a popularidade de dispositivos Android, o número de aplicativos Android aumentou nos últimos anos. Tendo seu uso se tornando cada vez mais

onipresente, a necessidade de testar sistematicamente os aplicativos nesses dispositivos cresce cada vez mais (ZAEEM et al., 2014) .

O presente trabalho de conclusão de curso se propõe a identificar tais técnicas e ferramentas Para isso, foi utilizado um mapeamento sistemático dos trabalhos da literatura em teste de software para aplicativos Android. Acredita-se que este mapeamento será útil para as pessoas que estão trabalhando no desenvolvimento de aplicativos Android, para que tenham conhecimento e possam se valer das diversas ferramentas e técnicas utilizadas nos testes deste tipo de software e para saberem onde podem buscar mais informações sobre elas, visto que algumas são bem recentes. Além disso, a pesquisa pode ajudar pesquisadores a identificar áreas de pesquisa que precisam de maior investigação, bem como as instituições trabalhando nessa direção.

Segundo Ramos e Brasil (2011), um mapeamento sistemático da literatura é uma espécie de revisão sistemática que representa uma revisão mais ampla de uma área de pesquisa, a fim de identificar quais evidências estão acessíveis. Os resultados de um estudo de mapeamento podem identificar áreas adequadas para a realização de revisões sistemáticas da literatura e também áreas onde um estudo primário é mais apropriado. O mapeamento sistemático proporciona uma perspectiva geral de uma área de pesquisa, através da identificação da quantidade, os tipos de pesquisas executadas, os resultados disponíveis, e ainda a frequência das publicações ao longo do tempo para constatar tendências (RAMOS; BRASIL, 2011).

1.1 OBJETIVO GERAL

O objetivo geral deste trabalho é identificar as técnicas e ferramentas propostas mencionadas na literatura para o teste de aplicativos Android e os tipos de teste cobertos por tais ferramentas.

1.2 OBJETIVOS ESPECÍFICOS

Este trabalho apresenta os seguintes objetivos específicos.

- Identificar os trabalhos da literatura sobre teste de software para aplicativos Android classificando-os segundo os seus objetivos em trabalhos sobre ferramentas de teste e trabalhos sobre técnicas de teste;
- Identificar as ferramentas de teste citadas pelos trabalhos da área;
- Identificar os tipos de testes relacionados às ferramentas propostas;

- Identificar as instituições e empresas publicando na área;
- Identificar o número e tipo de publicações na área ao longo dos anos.

1.3 ESTRUTURA

Este trabalho está organizado da seguinte forma: O primeiro capítulo reporta-se à motivação do uso de ferramentas e técnicas para o desenvolvimento do sistema Android e seus conceitos básicos. O Capítulo 2 apresenta os conceitos que estão relacionados ao tema de teste de software para aplicativos Android e que são necessários para a compreensão do trabalho. No Capítulo 3 é apresentado o método de pesquisa utilizado no presente trabalho, a condução da pesquisa e os resultados obtidos. No capítulo 4 são apresentadas as conclusões do presente trabalho, principalmente apresentando as ferramentas mencionadas pela literatura e os testes cobertos por essas ferramentas, de acordo com o mapeamento sistemático da literatura que foi realizado. Além disso, nesse capítulo são sugeridos também possíveis trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresentará os conceitos sobre a área de teste de software e sobre aplicativos Android, com o propósito de facilitar a compreensão deste trabalho.

2.1 ANDROID

Android tem sido considerada uma plataforma líder na área de *smartphones* e é suportada por centenas de dispositivos de diversos fabricantes disponíveis no mercado hoje (MITCHELL et al., 2014). Ela foi desenvolvida com o intuito de conceder aos desenvolvedores a possibilidade de produzir aplicações móveis que possam aproveitar vários dos recursos de um aparelho portátil, como realizar chamadas, enviar mensagens de texto ou usar a câmera (PEREIRA et al., 2003). A plataforma Android foi criada pela Google e pela Open Handset Alliance e está dentro de milhões de telefones celulares e outros dispositivos móveis (BURNETTE, 2009). Ela é uma plataforma de código aberto (*open source*) para o desenvolvimento de sistemas móveis e que apresenta um sistema operacional Linux padrão, um ambiente de execução personalizado, um arcabouço (*framework*) para aplicações abrangente e um conjunto de aplicativos do usuário (MAJI et al., 2012). Como se baseia no Linux, ele apresenta um modelo de *drivers* robusto, recursos de segurança, gerenciamento de processos, gerenciamento de memória, auxílio à gerência de rede e *drivers* para um grande conjunto de dispositivos. O ambiente de execução (*runtime*) é composto por bibliotecas gerais e pelo Dalvik, uma máquina virtual baseada em registro e otimizada para rodar considerando os requisitos de memória escassos. O arcabouço de aplicações provê aos desenvolvedores APIs (*Application Programming Interfaces*) para a criação de aplicativos de usuário (popularmente conhecidos como *apps*).

Segundo Khomh et al. (2012), Android é também o nome deste sistema operacional que se tornou uma das adaptações mais populares do kernel do Linux, com aproximadamente 60 milhões de novos dispositivos móveis rodando com Android a cada ano.

Segundo Hu e Neamtiu (2011), as aplicações Android podem ser compostas de quatro categorias de componentes: atividade (*activity)*, receptor de transmissão *(broadcast receiver*), provedor de conteúdo *(content provider*) e serviço *(service).* As atividades são as janelas em que ocorre a interação do usuário; apenas uma atividade pode estar ativa por vez. Cada atividade é uma classe no código-fonte e deve executar de acordo com os eventos gerados pelo sistema ou pelos usuários. Serviços são executados em segundo plano (*background*). Por

exemplo, um cliente de e-mail pode estar verificando novos e-mails enquanto os usuários estiverem executando outro aplicativo. Um provedor de conteúdo gerencia os dados de um determinado aplicativo e controla a acessibilidade dos dados. Receptores de transmissão ouvem e reagem a eventos anunciados, como o aviso de que a bateria do dispositivo está fraca, assim como qualquer outro software. Os aplicativos desenvolvidos para Android precisam ser verificados e avaliados, portanto se faz necessário que estes sejam submetidos a testes.

2.2 TESTE DE SOFTWARE

Segundo Crespo et. al (2009), testar um software é executá-lo de maneira controlada com o intuito de analisar se este se comporta de acordo com o especificado. Vê-se, portanto, que é uma atividade importante para analisar se o software desenvolvido atende as especificações dos requisitos dos usuários. Para identificação dos defeitos, caso existam, os quais podem provocar falhas de software, é recomendada a realização de testes de maneira sistemática e cuidadosa, pois, além disso, testes permitem fornecer evidências da confiabilidade do software ou simplesmente da falta desta propriedade.

De acordo com Função e Oliveira (2011), teste de software pode ser definido como um conjunto de processos executados com o intuito de validar ou atribuir qualidade ao software adotando uma metodologia, conhecendo melhor o processo, adotando o conteúdo de engenharia de software, e fazendo uso de alguns modelos de engenharia. De maneira geral, o teste é um grupo de processos em que a execução será feita em cadeia, com o propósito de qualificar, minimizar *bugs*, por meio da apuração, mapeamento e análise rigorosa, que tem a finalidade de garantir a confiabilidade, integridade no desenvolvimento/modificação. O teste tem como atribuição a qualidade do software e diminui determinadas falhas críticas em produção.

O objetivo de testar um software é a validação. O objetivo de certas empresas especializadas em testes de software é efetuar o levantamento de algumas anomalias e erros que ocorrem no uso de suas aplicações (FUNÇÃO; OLIVEIRA, 2011).

Molinari (2003, p. 28) afirma o seguinte sobre testes de software:

> Testes de software são a mais popular estratégia de gerenciamento de risco. São usados para verificar o encontro dos requerimentos com o produto. A limitação de sua abordagem, entretanto, é o momento em que ocorre, pois pode ser tarde para construir um produto de qualidade. São bons conforme a abrangência, direcionamento e profundidade dos casos de teste, porém podem ser inspecionados de modo a garantir que todos os requerimentos sejam testados através de todas as combinações de inputs e estados do sistema (MOLINARI, 2003, p. 28).

2.3 TIPOS DE TESTE

De acordo com Bourque e Fairley (2014), o teste de software é normalmente realizado em diferentes níveis ao longo dos processos de desenvolvimento e manutenção. Ou seja, o alvo do teste pode variar: um único módulo, um grupo de tais módulos (relacionados por fim, o uso, comportamento, ou estrutura), ou um sistema inteiro. Três grandes estágios de testes podem ser conceitualmente distintos, ou seja, Unidade, Integração e Sistema. Nenhum modelo de processo está implícito, nem são de qualquer uma destas três fases tidos como tendo uma maior importância do que os outros dois. Na Figura 1 são mostrados alguns tipos de teste de acordo com o seu alvo.

Figura 1: Tipos de teste de acordo com o seu alvo.

Fonte: A autora.

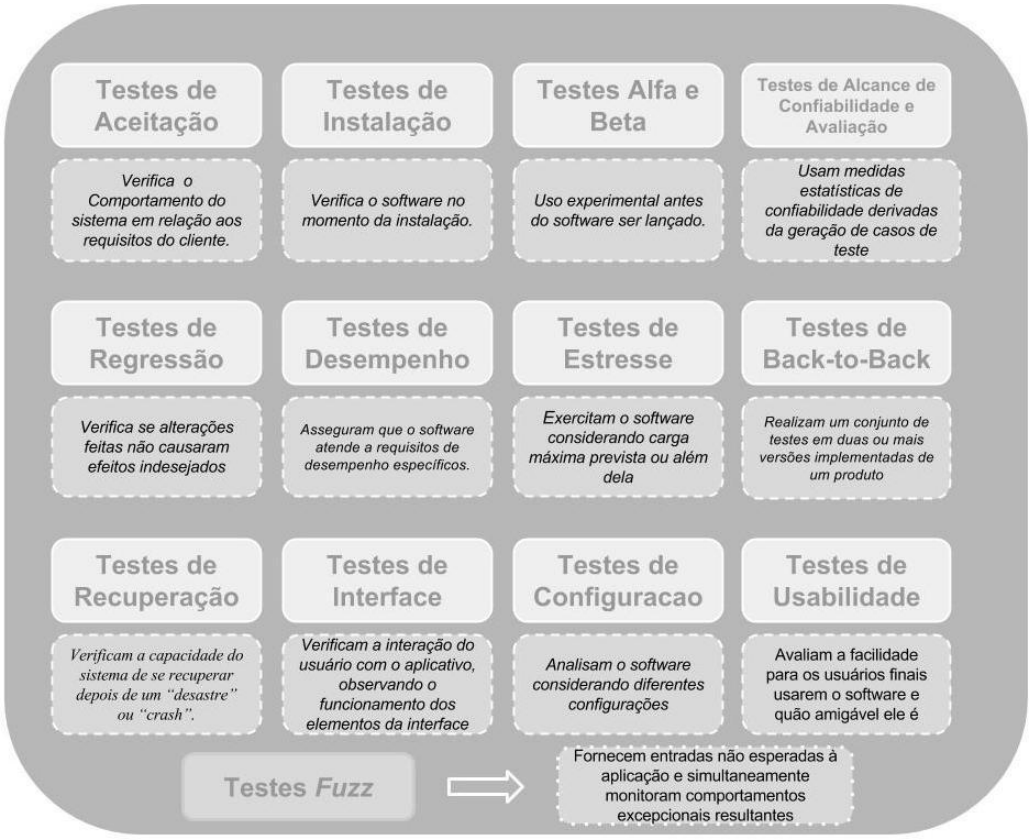
Segundo Bourque e Fairley (2014), o teste é realizado com vista a um determinado objetivo, o qual está apresentado mais ou menos de forma explícita, e com diferentes graus de precisão. Afirmando o objetivo em termos quantitativos precisos, permite o controle a ser estabelecido sobre o processo de teste.

O teste pode ser destinado a verificar propriedades diferentes. Os casos de teste podem ser projetados para verificar se as especificações funcionais são corretamente implementadas, o que é variadamente referido na literatura como testes de conformidade, teste de correção, ou

teste funcional. No entanto, várias outras propriedades não-funcionais podem ser testados, incluindo o desempenho, confiabilidade e usabilidade, entre muitos outros.

Outros objetivos importantes para os testes incluem (mas não estão limitados a) medição de confiabilidade, avaliação de usabilidade e aceitação, por que seria feita de diferentes abordagens. Na Figura 2 será mostrada uma breve descrição sobre tipos de teste de acordo com os objetivos, considerando os tipos previstos no SWEBOK (BOURQUE; FAIRLEY, 2014) acrescidos apenas dos Testes Fuzz (ilustrados na parte inferior da figura). As fontes para as definições utilizadas nesta figura foram baseadas no próprio SWEBOK e nos trabalhos de Delamaro et al. (2007), Molinari (2003), Bastos et al. (2007) e Ye et al. (2013).

Figura 2: Tipos de teste de acordo com o seu objetivo



Fonte: A autora

3 CONDUÇÃO DO MAPEAMENTO SISTEMÁTICO

Considerando o que é apresentado por Petersen et al. (2008), Kitchenham et al. (2008), Ramos e Brasil (2011), o processo de mapeamento sistemático da literatura é feito através de etapas, onde se define questões referentes à pesquisa, realizando a busca de trabalhos apropriados, escolha de trabalhos, palavras-chave de resumos, extração de dados e mapeamento.

Para que o mapeamento seja realizado, é necessário seguir etapas. De acordo com Silva (2012, p. 27) "O primeiro passo é definir o tema a ser mapeado, ou seja, definir o escopo sobre o qual se deseja obter dados relevantes e específicos já existentes na literatura da área e também justificar a realização do estudo". Em seguida, para identificar as questões de pesquisa relacionadas ao tema, deve-se defini-las (e.g.: mapear os principais meios de publicação de uma área). Segundo o mesmo autor, o próximo passo a seguir é a definição de um protocolo de busca, o qual deve possuir detalhes de todo o planejamento de estudo. Este deve conter uma introdução referente à área de conhecimento que será mapeada, para a realização do estudo, uma justificativa, a identificação das questões de pesquisa, a análise de como será realizada a busca (automática e/ou manual), os critérios de inclusão e exclusão e por fim a construção do mapa sistemático.

Silva (2012) mostra como etapa seguinte o processo de busca que é realizado através das buscas automáticas ou manuais. Nas buscas automáticas serão definidos quais engenhos de busca serão utilizados e nas buscas manuais devem ser definidas as principais conferências/jornais da área.

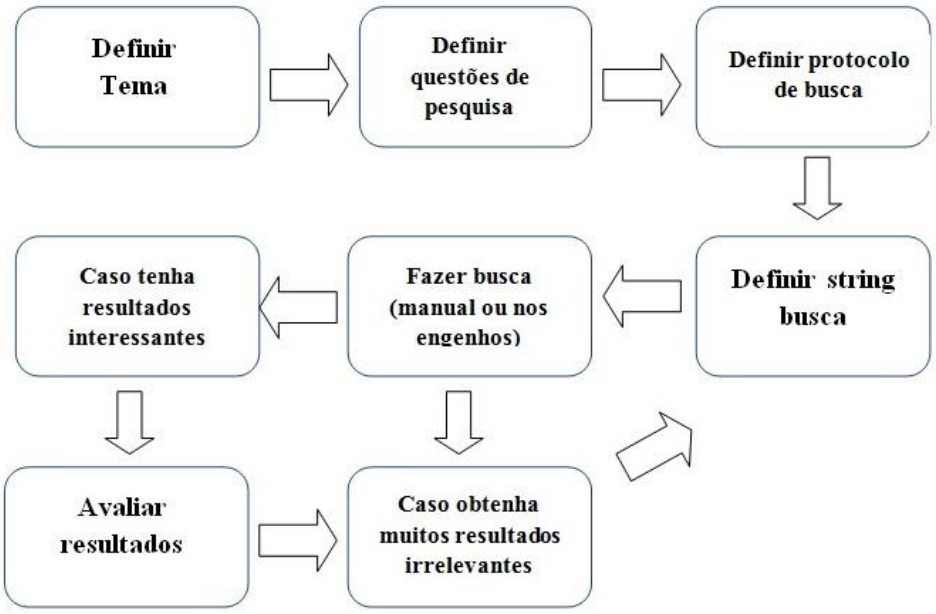
Neste trabalho, o tipo de busca escolhido foi a busca automática e tiveram de ser definidos alguns critérios de inclusão e de exclusão a serem utilizados nos resultados retornados pelos engenhos de busca para assim indicar quais trabalhos seriam incluídos ou excluídos do mapeamento.

Para não haver discordâncias nas avaliações dos pesquisadores participantes do mapeamento, os critérios precisam estar bem definidos. Neste trabalho, apenas uma pessoa participou do processo de análise dos artigos, sob a supervisão de sua orientadora em momentos de dúvida. Primeiramente, foi necessário fazer um tipo de triagem de todos os resultados retornados nos engenhos de busca utilizados para excluir as repetições. Após isso, pôde-se partir para a análise detalhada de documentos retornados para identificar e selecionar os mais adequados para responder às questões de pesquisa, respeitando os critérios de

inclusão e exclusão. Após a seleção dos trabalhos, eles foram analisados em detalhe, a fim de extrair dados para construir o mapa sistemático.

Seguindo afirma Silva (2012), para finalizar um mapeamento sistemático, deve-se decidir se os documentos serão lidos na íntegra ou se serão analisados apenas os seus resumos e também se os trabalhos que não foram publicados em eventos ou conferências devem ser analisados. Neste trabalho, os documentos foram selecionados de acordo com o seu resumo e posteriormente os selecionados foram lidos por completo com o objetivo de responder às questões de pesquisa. A Figura 3 mostra um resumo das etapas necessárias para a obtenção dos resultados em um mapeamento sistemático:

Figura 3: Etapas para a realização de um mapeamento sistemático (SILVA, 2012).



3.1 PROTOCOLO PARA O MAPEAMENTO SISTEMÁTICO DA LITERATURA

Para identificar os trabalhos sobre teste de software para aplicativos Android, é necessário seguir um protocolo e segundo Kitchenham (2007), os componentes necessários para a realização da pesquisa são: As questões de pesquisa, a definição das fontes de busca e suas estratégias, os critérios para inclusão e exclusão e, por fim, o procedimento da seleção dos artigos.

3.2 QUESTÕES DE PESQUISA

Algumas questões de pesquisa foram levantadas a fim de conduzir a realização deste trabalho. Algumas foram de cunho mais geral (as três primeiras) e as demais foram de cunho mais específico:

Q1. Em que anos estão sendo publicados os artigos na área de testes de aplicativos Android?

Q2. Quais os tipos dos artigos publicados na área quanto à forma de publicação? (Artigo publicado em evento / Artigo de periódico).

Q3. Que universidades/instituições ou empresas estão pesquisando sobre teste de aplicativos Android?

Q4. Quais os principais tipos de trabalhos sobre testes para aplicativos Android quanto aos seus objetivos (ex: propostas de ferramentas/ propostas de técnicas)

Q5. Quais as principais ferramentas citadas pela literatura para o teste de aplicativos Android?

Q6. Quais os principais tipos de teste cobertos pelas ferramentas de teste sugeridas pela literatura?

3.3 FONTES E ESTRATÉGIAS DE BUSCA

Foi realizado um levantamento bibliográfico nas bibliotecas digitais da ACM (*ACM Digital Library)*[1] *e IEEE (IEEE Xplore Digital Library*)[2]. Foram utilizados as strings de busca acrescidas dos operadores booleanos OR e AND. No caso da IEEE foi utilizada a busca simples, ou seja, acrescentando as palavras-chaves na caixa de busca da página principal dessa biblioteca digital considerando as restrições de tempo para realização desta pesquisa já que na opção de busca avançada apareceram muito mais artigos, muitos dos quais não eram diretamente relacionados ao tema pesquisado. No Quadro 1 são mostradas as bases de buscas e suas respectivas strings utilizadas para consulta.

[1] http://dl.acm.org/
[2] http://ieeexplore.ieee.org/

Quadro 1:Base de buscas e strings utilizadas.

Base	String
ACM Digital Library	((Title:(“System” or “Software” or “Sistema” or “Application” or “aplicativo”) and (“test” or “testing” or “Teste”) and (“Android”))) or (Abstract:(“System” or “Software” or “Sistema” or “application” or “aplicativo”) and (“test” or “testing” or “Teste”) and (“Android”)))
IEEE Xplore	("Software" OR "system" OR "application" OR "aplicativo") AND ("test" OR "testing" OR "teste") AND ("Android")

Fonte: a autora

3.4 CRITÉRIOS DE INCLUSÃO E EXCLUSÃO

Após realizar as buscas, foram encontrados diversos artigos relacionados com o tema proposto, porém nem todos são relevantes, como é característico de um mapeamento sistemático (HENRIQUE, 2013, p. 21). Portanto, foram definidos os critérios de inclusão e exclusão de artigos no mapeamento, apenas para selecionar os trabalhos que são considerados relevantes de acordo com as questões de pesquisa. No Quadro 2 são mostrados os critérios de inclusão e exclusão utilizados.

Quadro 2: Critérios de inclusão e exclusão.

Critérios de Inclusão	Critérios de Exclusão
Os trabalhos pesquisados devem estar escritos em inglês ou português.	Foram excluídos artigos que não têm relação direta com o tema proposto de acordo com o seu título ou seu resumo.
	Artigos resumidos dos mesmos autores nos casos em que há uma versão estendida do trabalho mais recente foram também excluídos.
Os trabalhos devem estar disponíveis para serem baixados através da rede da UFPB.	Artigos duplicados (que estavam presentes nos resultados retornados pelos dois engenhos de busca).
	Artigos curtos *(short papers)* com menos de três páginas.

Fonte: a autora

3.5 PROCEDIMENTO DA SELEÇÃO DOS ARTIGOS

O procedimento para a seleção dos artigos foi feito através da busca dos trabalhos utilizando as strings de busca mostradas no Quadro 1 nos engenhos de busca escolhidos. Em seguida, foi realizado um filtro dos trabalhos considerando os critérios de inclusão e exclusão considerados. Para identificar os trabalhos que tinham relação com o tema proposto, era feita a verificação do título e do resumo. Havendo dúvidas, a introdução e a conclusão foram lidas antes de excluir um trabalho. Após esse primeiro passo, os trabalhos selecionados foram analisados por completo com o objetivo de responder às questões de pesquisa propostas.

Para cada artigo selecionado, foram retiradas as seguintes informações:

- Título: Nome do trabalho investigado;
- Descrição: Descrição do trabalho examinado e dos assuntos que estão sendo explorados focando no que está sendo proposto pelo trabalho (seus objetivos) considerando o teste de sistemas Android;
- Referência: Referência do trabalho;
- Ano da publicação;
- Filiação dos autores;
- Tipos de teste de que trata a ferramenta discutida pelo artigo (quando este for o caso); e
- Ferramentas citadas nos artigos incluídos e relacionadas diretamente a estes trabalhos.

3.6 RESULTADOS GERAIS DO MAPEAMENTO

O mapeamento foi realizado entre o mês de maio de 2014 e o mês de Agosto de 2014. Através das buscas realizadas, resultaram 58 documentos na biblioteca da ACM (*ACM Digital Library*) e 17 na biblioteca da IEEE Xplore (*IEEE Xplore Digital Library*).

Depois disso, foram aplicados os critérios de inclusão e exclusão através da verificação do título e do resumo dos artigos resultando em 17 documentos selecionados dentre os que foram retornados na busca feita na biblioteca da ACM (*ACM Digital Library*) e 11 na biblioteca da IEEE Xplore (*IEEE Xplore Digital Library*). Conforme ilustrado pela Tabela 1.

Tabela 1: Resultados Gerais do Mapeamento.

Base		Quantidade de documentos			
	Busca inicial	Critérios de Exclusão			Resultante
		Sem relação	Duplicado	Artigo Curto	
ACM Digital Library [3]	**58**	33	0	8	**17**
IEEE Xplore[4]	**17**	5	0	1	**11**
Total	**75**				**28**

Fonte: a autora

Nesta fase foram incluídos 28 artigos, os quais foram examinados de maneira completa com a finalidade de responder às questões de pesquisa. Foram excluídos 47 artigos (Apêndice B). Foram excluídos 33 documentos dentre os que foram retornados na busca feita na biblioteca da ACM (*ACM Digital Library*) por não terem relação com o tema e 8 da ACM por serem artigos curtos. Dentre os resultados feitos na biblioteca da IEEE Xplore (*IEEE Xplore Digital Library*), foram excluídos 5 artigos sem relação com o tema proposto e 1 artigo por ser curto. Dos 75 artigos encontrados no total, 6 não foram possíveis de se baixar pela rede da UFPB, ou seja, estavam inacessíveis e também não estavam disponíveis quando buscados através do Google Acadêmico. A Tabela 1 resume os dados relativos aos artigos encontrados pelas buscas e os resultantes.

3.7 RESULTADOS REFERENTES ÀS QUESTÕES DE PESQUISA

Nesta subseção serão apresentados os resultados gerados pela análise dos 28 artigos incluídos quanto às respostas às questões de pesquisa apresentadas na subseção 3.2. Cada artigo recebeu um identificador iniciado pela letra P e sua citação completa se encontra no Apêndice A.

A seguir são apresentadas inicialmente as questões de cunho geral (Q1, Q2 e Q3) e posteriormente as de cunho específico (Q4, Q5 e Q6).

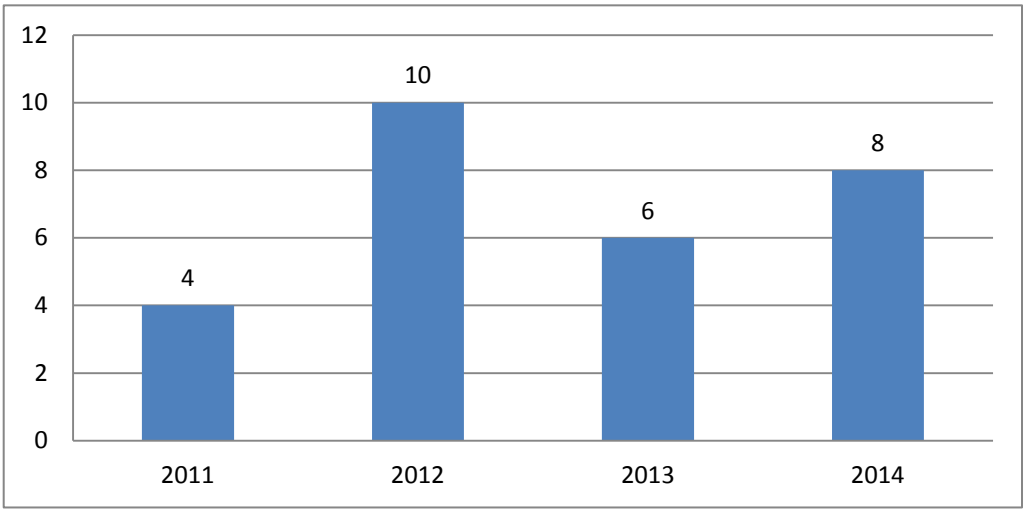
[3] ((Title:(“System” or “Software” or “Sistema” or “Application” or “aplicativo”) and (“test” or “testing” or “Teste”) and (“Android”))) or (Abstract:(“System” or “Software” or “Sistema” or “application” or “aplicativo”) and (“test” or “testing” or “Teste”) and (“Android”)))

[4] ("Software" OR "system" OR "application" OR "aplicativo") AND ("test" OR "testing" OR "teste") AND ("Android")

3.7.1 Q1: Em que anos estão sendo publicados os artigos?

Os artigos analisados na pesquisa são trabalhos publicados entre os anos de 2011 e 2014 (Gráfico 1). Isto mostra que o interesse em pesquisas na área de testes para aplicativos Android se iniciou em 2011. No ano de 2014, até o momento da realização da pesquisa, foram considerados 8 artigos, um número elevado por considerar que o mapeamento foi realizado até o mês de agosto de 2014 e que é provável que até o mês de dezembro outros artigos sejam publicados sobre o tema.

Gráfico 1: Quantidade de artigos publicados por ano.



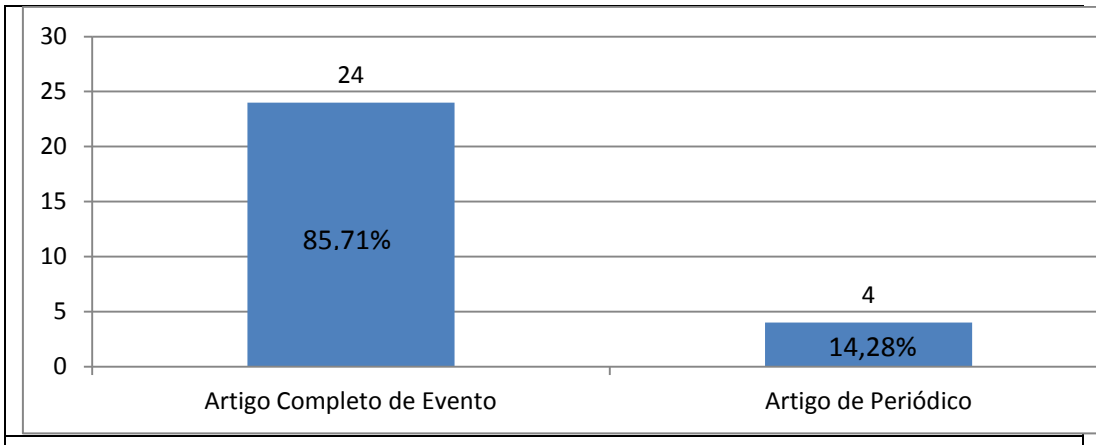
2014: [P1, P2, P4, P9, P11, P12, P21, P28]
2013: [P7, P10, P18, P22, P25, P26]
2012: [P3, P6, P13, P14, P16, P17, P20, P23, P24, P27]
2012: [P5, P8, P15, P19]

Fonte: a autora

3.7.2 Q2: Quais os tipos dos artigos publicados na área quanto à forma de publicação? (Artigo publicado em evento / Artigo de periódico).

Esta subseção mostra a questão referente aos tipos de artigos publicados na área considerando a forma de publicação destes artigos. No Gráfico 2 se pode observar os dados detalhados de cada tipo. Como *Artigo publicado em evento* foram encontrados 24 artigos, os quais representam 85,71% do total dos artigos incluídos. Em relação à categoria *Artigo de periódico*, foram encontrados apenas 4 artigos com a representatividade de 14,28% da totalidade de artigos incluídos.

Gráfico 2: Quantidade de artigos pesquisados por tipo.



Artigo publicado em evento:
[P1]; [P2]; [P3] [P4]; [P5]; [P6]; [P7]; [P8]; [P9]; [P10]; [P11]; [P12]; [P13]; [P15]; [P16]; [P19]; [P20]; [P22]; [P23]; [P24]; [P25]; [P26]; [P27]; [P28]

Artigo de periódico:
[P14]; [P17]; [P18]; [P21]

Fonte: a autora

A publicação dos artigos na categoria "Artigo publicado em evento" tiveram uma grande diferença em relação aos artigos publicados em periódicos, portanto conclui-se que o interesse maior foi em publicar artigos na primeira categoria. Além disso, percebeu-se que os eventos em que foram publicados os artigos foram bem variados, o que mostra que pesquisas bibliográficas na área não devem se focar em apenas um evento. Para esta pesquisa, por exemplo, foram encontrados artigos dos seguintes eventos:

- *International Symposium on Service Oriented System Engineering (SOSE)*;
- *Annual ACM Symposium on Applied Computing;*
- *International Workshop on Automation of Software Test*;
- *ACM conference on Computer and communications security*;
- *ACM SIGSOFT 20th International Symposium on the Foundations of Software Engineering*;
- *International Symposium on Software Testing and Analysis;*
- *International Workshop on Automation of Software Test*;
- *International Workshop on Automation of Software Test;*
- *International Conference on Advances in Mobile Computing & Multimedia*;
- *European Conference on Computer Systems*;

- *Joint International Workshop on Dynamic Analysis (WODA) and Software and System Performance Testing, Debugging, and Analytics (PERTEA);*
- *International Conference on Mobile and Ubiquitous Multimedia;*
- *ACM conference on Computer and communications security*;
- *IEEE/ACM International Conference on Automated Software Engineering*;
- *International Symposium on Quality Electronic Design (ISQED);*
- *International Symposium on Software Reliability Engineering (ISSRE);*
- *International Conference on Software Testing, Verification and Validation (ICST);*
- *World Congress on Engineering (WCSE);*
- *International Conference on Software Testing, Verification and Validation Workshops (ICSTW);*
- *Workshop on Automation of Software Test (AST);*
- *International Conference on Cloud Computing and Intelligent Systems (CCIS;*
- *International conference on Human-computer interaction with mobile devices and services.*

Os periódicos em que foram publicados artigos selecionados foram o *ACM SIGSOFT Software Engineering Notes*, um boletim informativo, *o* jornal *Mobile Networks and Applications* e a revista IEEE *Software*.

3.7.3 Q3: Que universidades/instituições ou empresas estão pesquisando sobre teste de aplicativos Android?

Esta questão analisa quais as universidades, instituições ou empresas que estão pesquisando sobre teste de aplicativos Android. O Quadro 3 apresenta os artigos pesquisados por universidade, instituição ou empresa e agrupados por países. De maneira geral, pode-se observar que para cada instituição/empresa relacionada aos autores dos artigos selecionados, houve no máximo dois artigos. Isso ocorreu com a *Università Federico II Napoli*, da Itália, relacionada aos artigos [P16] e [P25]; com a *Fujitsu Labs*, dos Estados Unidos, relacionada aos artigos [P7] e [P28] e também com a *Beijing University of Posts and Telecommunications*, da China, relacionada aos artigos [P1] e [P27].

Quadro 3: Artigos pesquisados por Universidade, Instituição ou Empresas.

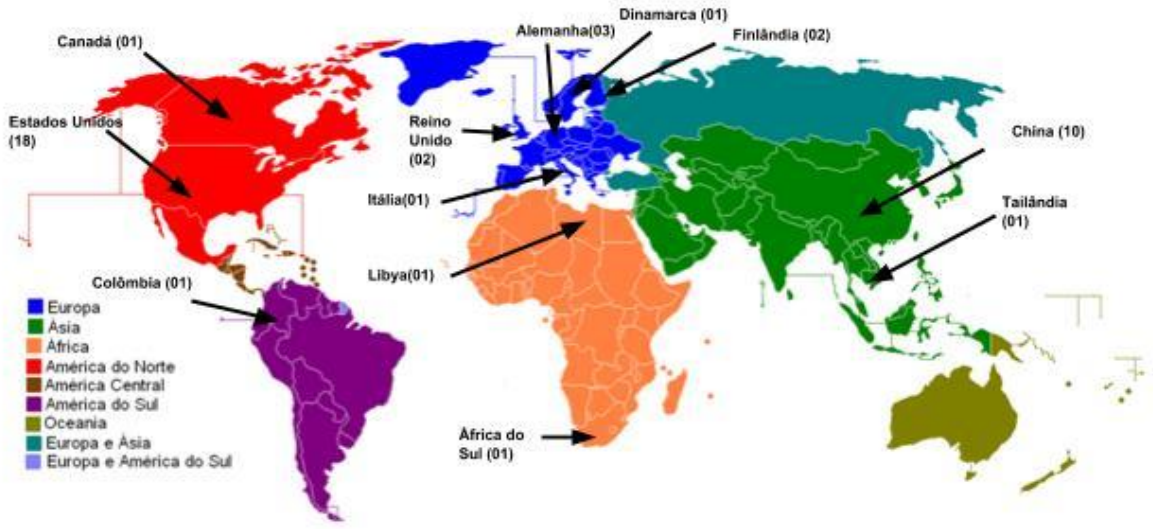
Distribuição de publicações por origem			
Continente	**Pais**	**Universidade, Instituição, Empresa**	**id**
América do Norte	Estados Unidos da América	Fujitsu Labs	[P7]; [P28]
		University of California, Santa Cruz	[P10]
		University of Massachusetts Lowell	[P18]
		University of California, Riverside	[P8]
		University of California, Berkeley	[P5]
		Georgia Tech	[P6]
		University of Maryland	[P16]
		George Mason University	[P14]
		NASA Ames Research Center	[P14]
		University of Washington	[P15]
		Microsoft Research, Redmond	[P15]
		Carnegie Mellon University	[P3]
		University of Utah	[P12]
		Arizona University State	[P20]
		Ohio State University	[P20]
	Canadá	University of Waterloo	[P9]
Europa	Alemanha	University of Duisburg-Essen	[P2]
		Embedded Software Laboratory	[P23]
		IVU Traffic Technologies AG	[P23]
	Finlândia	University of Oulu	[P13]
		University of Technology	[P19]
	Reino Unido	Bath Spa University	[P4]
		University of Oxford	[P6]
	Itália	Università Federico II Napoli	[P16];[P25]
	Dinamarca	Aarhus University	[P7]
África	Lybia	Aljabal Algharby University	[P9]
	Africa do Sul	University of Stellenbosch	[P17]
América do Sul	Colômbia	Columbia University	[P11]
Ásia	Tailândia	King Mongkut's University of Technology	[P23]
	China	Beijing University of Posts and Telecommunications	[P1];[P27]
		University of Science and Technology of China	[P10]
		Nanjing University, Republic of China	[P18]
		Nankai University, China	[P4]
		National Computer Network Emergency Response Technical Team	[P1]
		National Chiao Tung University	[P21]
		National Yunlin University of Science and Technology	[P21]
		National Taiwan University of Science and Technology	[P21]
		South China University of Technology	[P24]

		Guangdong Power Grid Corporation	[P24]

Fonte: a autora

Embora os países que estão pesquisando sobre testes de aplicativos Android não tenham sido uma questão de pesquisa, achou-se importante investigar esse ponto. A Figura 4 mostra os países a que se refere cada artigo de acordo com as universidades, instituições ou empresas à qual estão ligados os autores. Como se pode observar, os Estados Unidos é o país que concentra o maior número de universidades ou empresas (18) pesquisando sobre teste de aplicativos Android, estando uma delas relacionada a dois trabalhos. A China esteve envolvida com a participação de 10 universidades ou empresas, estando uma delas relacionada também a dois trabalhos. A Alemanha esteve representada por 3 universidades ou empresas. A Finlândia e o Reino Unido foram representados por 2 universidades. Já a Itália, Dinamarca, Canadá, Libya, África do Sul, Colômbia e Tailândia apresentaram cada um, apenas 1 universidade pesquisando sobre o assunto e com um trabalho cada.

Figura 4: Países participantes de pesquisas na área de acordo com as universidades, instituições ou empresas dos autores dos trabalhos selecionados.

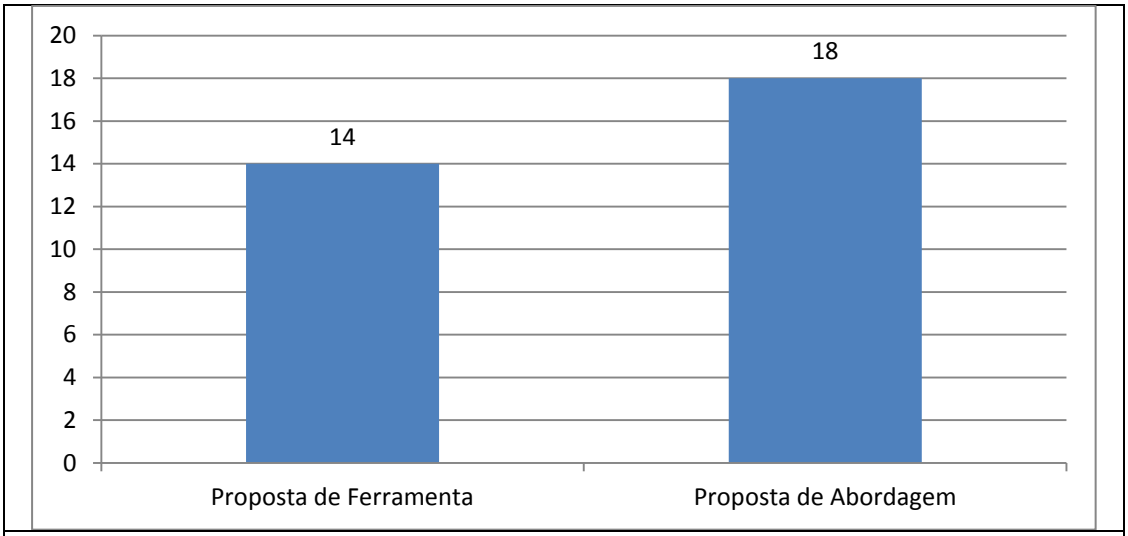


Fonte: Imagem retirada do site http://pt.wikipedia.org/wiki/Continente#mediaviewer/ e organizada de acordo com os dados da autora.

3.7.4 Q4: Quais os principais tipos de trabalhos sobre testes para aplicativos Android quanto aos seus objetivos (ex: propostas de ferramentas/ propostas de técnicas)

Nesta seção são apresentadas as duas categorias adotadas pelos trabalhos mapeados considerando as *propostas ou objetivos* destes trabalhos sobre teste de aplicativos Android e informações gerais sobre os trabalhos selecionados. A primeira categoria considera *proposta de ferramenta, toolkit, plataforma ou arcabouço (framework)* de teste, enquanto que a segunda, contempla *proposta de abordagem, metodologia, técnica ou método de teste*. Neste caso, o mesmo artigo também pode estar incluso nas duas categorias. O Gráfico 3 resume a classificação dos artigos de acordo com essas categorias. No Quadro 4 são apresentadas as categorias definidas para este mapeamento, considerando proposta de ferramentas e os artigos que nelas se enquadraram e no Quadro 5 são apresentadas as propostas de abordagens ou técnicas de teste.

Gráfico 3: Quantidade de artigos que se enquadram nas categorias definidas quanto aos objetivos dos trabalhos.



Proposta de Ferramenta [P3]; [P5]; [P7]; [P10]; [P11]; [P12]; [P13]; [P16]; [P17]; [P18]; [P19]; [P21]; [P26]; [P28]

Proposta de Abordagem [P1]; [P2]; [P4]; [P6]; [P8]; [P9]; [P10]; [P14]; [P15]; [P16]; [P19]; [P20]; [P22]; [P23]; [P24]; [P25]; [P27]; [P28]

Fonte: a autora

Quadro 4: Considerando proposta de ferramentas.

Proposta de ferramenta			
Id	**O quê?**	**Para quê?**	**Como?**
P3	Sistema de prototipagem *Wizard of Oz*.	Testes de usuário baseados em cenários.	Observar o usuário e simular funcionalidades não implementadas.
P5	Ferramenta de análise estática Stowaway.	Analisa privilégio demasiado (overprivilege) em aplicações Android.	---
P7	Arcabouço (framework)	Teste automático de aplicações orientadas a evento.	Execução concolic[5] e modelos da interface do usuário para geração de entradas de teste que alcancem pontos do código da aplicação difíceis de serem alcançados.
P10	Método e ferramenta DroidFuzzer para testar aplicativos Android.	Testar os aplicativos Android.	Uso de ferramenta para teste Fuzz chamada DroidFuzzer.
P11	AppDoctor, um sistema que testa os aplicativos Android.	Ajuda os desenvolvedores no diagnóstico dos relatórios de bugs.	Usa a técnica de *approximate execution*[6] para acelerar os testes e reduzir o esforço de diagnóstico. E utiliza *action slicing* para reduzir o tamanho dos rastreios (traces) de execução.
P12	Ferramenta chamada Intent Fuzzer.	Combina análise estática e geração aleatória de casos de teste.	---
P13	Testdroid, uma plataforma online.	Conduzir testes de interface do usuário.	Permite testes via scripts que podem ser executados em paralelo em uma variedade de dispositivos usando o sistema Android.
P16	AndroidRipper, uma técnica automatizada implementada em uma ferramenta.	Teste de aplicativos Android.	Permite testes através da interface gráfica do usuário (GUI).
P17	Ferramenta JPF-ANDROID.	Verificação de aplicativos Android.	---
P18	Conjunto de ferramentas (toolkit).	Incorpora em aplicações móveis a capacidade de automaticamente coletar eventos da interface de usuário (UI).	Coleta informações a medida que o usuário interage com as aplicações permitindo assim a realização de testes de usabilidade.
P19	Experiências no teste de interface gráfica. Descrição de uma ferramenta de automação de teste (TEMA *Tools*).	Ferramenta implementada para o emulador Android.	Estudo de caso no aplicativo BBC News.

[5] Técnica de verificação que combina execução simbólica com execução concreta de testes.

[6] Uma execução próxima da real mas que pode levar a falsos positivos.

P21	Ferramenta de teste chamada SPAG (*Smart Phone Automated Gui testing tool*).	Reproduzir operações de GUI e verificar resultados de teste.	---
P26	Solução de Testing-as-a-Service e plataforma de TaaS chamada *Cloud Testing of Mobile Systems* (CTOMS).	Desenvolvimento para dispositivos móveis.	Plataforma apresenta arquitetura escalável.
P28	Abordagem para gerar casos de teste. Ferramenta QUANTUM.	Gerar casos de teste para aplicativos móveis e inclui oráculos de teste, que têm o intuito de indicar a corretude de uma execução de testes.	Abordagem se fundamenta em um estudo abrangente de defeitos reais em aplicativos móveis.

Fonte: A autora

Quadro 5: Considerando proposta de abordagem.

Proposta de abordagem			
Id	**O quê?**	**Para quê?**	**Como?**
P1	Visão geral das tecnologias atuais.	Análise estática de códigos maliciosos Android.	Método que utiliza a assinatura das instruções.
P2	Abordagem baseada em modelo.	Para melhorar o teste de aplicações móveis sensíveis ao contexto.	Gerando casos de teste a partir de modelos.
P4	Abordagem composicional, incluindo técnicas de testes automatizados estáticos e dinâmicos.	Detectar as vulnerabilidades de segurança causadas por sistemas de mensagens entre componentes e para simular automaticamente os componentes do ataque.	----
P6	Algoritmo e um sistema.	Gerar eventos de entrada para aplicativos de smartphones.	Baseia-se em uma técnica de geração sistemática de entradas de teste de forma automática e sistemática.
P8	Abordagem de verificação de aplicativos Android	Detectar erros da GUI.	Por geração automática de casos de teste.
P9	Técnica.	Auxiliar nos testes de desempenho de energia de smartphones.	Baseada na categorização dos parâmetros de configuração dos telefones em três grupos para reduzir o número de configurações de teste.
P14	Abordagem.	A execução simbólica de aplicativos Android.	---
P15	Metodologia de testes	Avaliar efeitos colaterais de controles de privacidade para usuários de smartphones Android.	Metodologia baseada na gravação de telas da aplicação e identificação de diferenças em distintas execuções.

P20	Metodologia de projeto	Para o desenvolvimento e teste de hardware e software.	Baseada numa plataforma virtual e auxilia desde a fase de concepção do produto e até mesmo antes que sua arquitetura tenha sido finalizada.
P22	Abordagem inovadora e abrangente.	O teste sistemático de vazamentos de recurso (*resource* leaks) em software Android.	---
P23	Abordagem baseada em testes de unidade.	Testar propriedades dependents do ciclo de vida das aplicações Android.	Usa métodos de retorno (*call-back-methods*) de ciclo de vida com o fim de testar as propriedades.
P24	Método.	Facilitar o teste funcional automatizado de aplicativos móveis.	Faz uso das ferramentas de teste Robotium e MonkeyRunner.
P25	Abordagens de teste baseadas na definição de padrões de evento reutilizáveis.	Geração manual e automática de casos de teste.	Foca no problema de se testar um aplicativo móvel como um sistema orientado a eventos (*event-driven system*), levando em conta tanto os eventos da GUI quanto os eventos de contexto
P27	Método.	Avaliar a segurança de aplicações Android antes da instalação.	Baseado em técnicas de visualização de rede e em um algoritmo de clusterização *(clustering).*

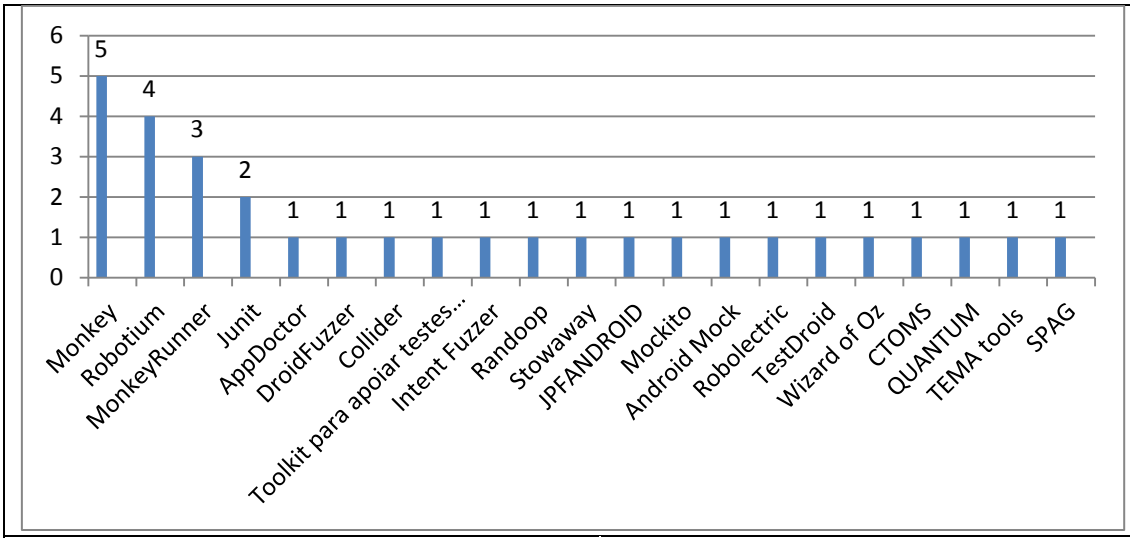
Fonte: A autora.

Considerando os objetivos dos trabalhos selecionados, vê-se que todos se enquadraram em pelo menos uma das duas categorias que foram definidas para este mapeamento sistemático: ou trabalhos para propor ferramentas de teste (de maneira geral), ou propostas de técnicas ou métodos de teste (algumas vezes usando ferramentas também). Além disso, observou-se que a distribuição dos trabalhos entre as duas categorias foi praticamente igual.

3.7.5 Q5: Quais as principais ferramentas citadas pela literatura para o teste de aplicativos Android?

Esta questão teve como objetivo identificar as principais ferramentas para o teste de aplicativos Android citadas nos artigos incluídos e relacionadas diretamente a eles. Através do Gráfico 4 é possível visualizar essas ferramentas, artigos e a frequência com que foram referenciadas nos trabalhos selecionados. Os artigos que citam estas ferramentas e o propósito de cada uma estão ilustrados pelo Quadro 6.

Gráfico 4:Ferramentas citadas nos artigos incluídos e relacionadas diretamente a eles..



Monkey [P7]; [P11]; [P16]; [P19]; [P26]
Robotium [P7]; [P17]; [P19]; [P28]
MonkeyRunner [P11]; [P17]; [P21]
JUnit [P17]; [P28]
AppDoctor [P11]
DroidFuzzer [P10]
Collider [P7]
Toolkit para apoiar teste de usabilidade [P18]
Intent Fuzzer [P12]
Randoop [P5]
Stowaway [P5]
JPF ANDROID [P17]
Mockito [P17]
Android Mock [P17]
Robolectric [P17]
TestDroid [P13]
Wizard Of Oz [P3]
CTOMS [P26]
TEMA Tools [P19]
SPAG [P21]
QUANTUM [P28]

Fonte: A autora.

A seguir serão apresentadas informações sobre as ferramentas citadas no Quadro 6.

Quadro 6: Ferramentas citadas nos artigos incluídos e relacionadas diretamente a eles quanto ao seu propósito..

Caracterização das Ferramentas		
Ferramentas	**Artigos Relacionados**	**Propósito**
Monkey	[P7]; [P11]; [P16]; [P19]; [P26]	Ferramenta de teste aleatório que dispara um grande número de eventos aleatórios e reinicia periodicamente a aplicação.
Robotium	[P7]; [P17]; [P19]; [P28]	Usada para simular interações do usuário com o aplicativo; uso do framework JUnit que vem com o Android.
MonkeyRunner	[P11]; [P17]; [P21]	Ferramenta para testes funcionais e que faz uso do framework JUnit que vem com o Android.
JUnit	[P17]; [P28]	Testa as unidades do sistema.
AppDoctor	[P11]	Testa os aplicativos Android de forma eficiente e eficaz considerando muitas ações do sistema e do usuário Essa ferramenta usa a técnica de *approximate execution* para acelerar os testes e reduzir o esforço de diagnóstico. Usa também *action slicing* para reduzir o

		tamanho dos rastreios (*traces*) de execução..
DroidFuzzer	[P10]	Apoia testes caixa-preta em aplicativos Android.
Collider	[P7]	Utilizada para avaliar a utilidade prática de uma abordagem que é proposta. Apoia essa abordagem por meio da geração de seqüências de eventos e de uma infra-estrutura de apoio.
Toolkit para apoiar testes de usabilidade	[P18]	Incorpora em aplicações móveis a capacidade de automaticamente coletar eventos da interface de usuário (UI) para permitir testes de usabilidade.
Intent Fuzzer	[P12]	Combina análise estática e geração aleatória de casos de teste.
Stowaway	[P5]	Analisa aplicações Android para determinar se não se utilizam de privilégio demasiado (*overprivilege*).
JPFANDROID	[P17]	Verificação de aplicativos Android.
Randoop		Geração de testes de unidade.
Mockito		Testes JUnit na máquina virtual Dalvik do Android usando classes Android mocks[7].
Android Mock		
Robolectric		Testes JUnit que executam sobre a JVM.
TestDroid	[P13]	Conduzir testes de interface do usuário via scripts em uma variedade de dispositivos usando o sistema Android.
Wizard of Oz	[P3]	Designer pode melhorar versões não funcionais geradas digitalmente ou desenhos em papel digitalizados com o uso de *widgets* (componentes de interface) interativos e transições de tela automatizadas.
CTOMS	[P26]	Plataforma de TaaS para auxiliar no desenvolvimento para dispositivos móveis.
TEMA tools	[P19]	Ferramenta de automação de teste de GUI baseada em palavra-chave.
SPAG	[P21]	Reproduzir operações de GUI e verificar resultados de teste.
QUANTUM	[P28]	Gerar os casos de teste que incluem os oráculos.

Fonte: A autora.

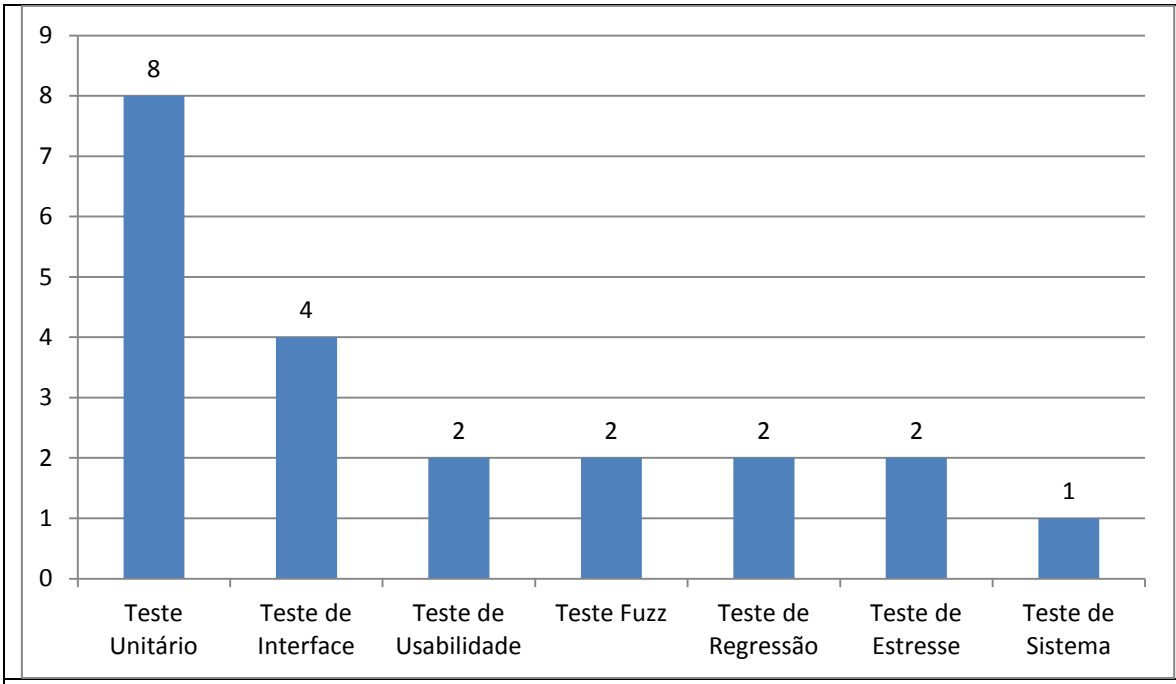
Podemos concluir que as ferramentas mais citadas nos artigos selecionados e relacionados diretamente a eles foram Monkey, Robotium e MonkeyRunner.

3.7.6 Q6. Quais os principais tipos de teste cobertos pelas ferramentas de teste sugeridas pela literatura?

Nesta seção são mostrados os principais tipos de testes (considerando diferentes tipos de classificação) cobertos pelas ferramentas de teste segundo os artigos que as referenciam. Através do Gráfico 5 é possível visualizar os tipos de teste cobertos pelas ferramentas, a quantidade de artigos que referenciam as ferramentas daquele tipo e a classificação dos tipos de teste quanto ao alvo ou quanto ao objetivo.

No Quadro 7 são apresentados a classificação dos testes cobertos pelas ferramentas quanto ao alvo ou objetivos e as próprias ferramentas.

Gráfico 5: Quantidade dos tipos de teste cobertos pelas ferramentas de acordo com seu alvo ou objetivo, e os artigos.



Quanto ao Alvo	**Quanto ao objetivo**
Teste Unitário: [P17]; [P28]; [P15]; [P5]; [P28] **Teste de Sistema:** [P7]	**Teste de Interface:** [P17]; [P28]; [P15]; [P5]; [P28] **Teste de Usabilidade:** [P18]; [P3] **Teste Fuzz:** [P10]; [P12] **Teste de Regressão:** [P11]; [P17]; [P21] **Teste de Estresse:** [P7]; [P11]; [P16]; [P19]; [P26]

Fonte: A autora.

Classificação dos testes		Ferramentas						
		Monkey CTOMS	AppDoctor MonkeyRunner	DroidFuzzer IntentFuzzer	Collider	Robotium TestDroid TEMA tools SPAG	Toolkit Wizard of Oz	JUnit Randoop Stowaway JPF ANDROID Mockito Android Mock Robolectric QUANTUM
Quanto ao alvo	Teste unitario							X
	Teste de integração							
	Teste de sistema				X			
Quanto aos objetivos	Teste de aceitação							
	Instalação							
	Alfa e Beta							
	Alcance de confiablidade e avaliação							
	Teste de Regresão		X					
	Teste de Desempenho							
	Teste de Estresse	X						
	Teste Back-to-Back							
	Teste de Recuperação							
	Teste de Interface					X		
	Teste de configuração							
	Teste de Usabilidade						X	
Outros	Teste Fuzz			X				

Quadro 7: Tipos de teste cobertos pelas ferramentas e as ferramentas.

Observando o Quadro 7, pode-se concluir que a maioria dos artigos que discutem ferramentas para o teste de sistemas Android exploram testes unitários que se enquadram na classificação dos testes quanto ao alvo e o teste de UI que se enquadra na classificação dos testes quanto aos objetivos. Sendo assim, pode ser interessante este resultado para pesquisadores que queiram identificar áreas que precisam de maior investigação.

4 CONSIDERAÇÕES FINAIS

4.1 CONCLUSÃO

Neste trabalho, foi realizado um Mapeamento Sistemático da Literatura sobre testes de software para aplicativos Android, com o intuito de identificar: as principais propostas ou objetivos dos artigos sobre teste de aplicativos Android, as principais ferramentas citadas pela literatura, os principais tipos de teste cobertos pelas ferramentas sugeridas em cada trabalho, o período em que esses trabalhos foram publicados, as instituições ou universidades que estão pesquisando sobre testes de aplicativos Android e os veículos de publicação dos principais artigos publicados na área.

Os 28 artigos que foram mapeados se enquadraram em uma das seguintes categorias quanto aos seus objetivos: Proposta de ferramenta/toolkit/plataforma/framework de teste ou em Proposta de abordagem/metodologia/técnica/método de teste. Nos artigos mapeados também foram encontradas diversas ferramentas que auxiliam no teste de aplicativos Android e foram levantados os tipos de teste cobertos por tais ferramentas. Testes de unidade e de UI foram os tipos de teste mais citados por serem relacionados às ferramentas encontradas.

Considerando a análise da autoria dos trabalhos sobre o teste de sistemas Android, viu-se que os Estados Unidos e a China são os países da maioria das publicações na área, que estão distribuídas entre diferentes universidades ou empresas. Para cada instituição isolada, porém, houve no máximo dois trabalhos dentre os que foram mapeados. Também foi possível observar que a maioria dos artigos encontrados pelo mapeamento feito foram publicados em Conferências Internacionais e bem variadas também.

Acredita-se que esse mapeamento dá uma visão geral dos tipos de teste mais cobertos pela literatura e dos principais trabalhos na área. Além disso, pode-se ter uma visão geral das ferramentas para apoiar no teste de aplicativos Android. Tal informação pode ser útil tanto para as pessoas que estão trabalhando com o desenvolvimento de aplicativos Android, que nem sempre têm ciência das ferramentas que podem lhe apoiar, quanto para pesquisadores, que podem com o estudo feito identificar áreas de pesquisa que necessitam de maior investigação, como alguns tipos de teste.

O mapeamento sistemático deste trabalho utilizou-se apenas das bibliotecas digitais da ACM (*ACM Digital Library), e IEEE (IEEE Xplore Digital Library*). Sugere-se estender esse mapeamento realizando pesquisas automáticas na biblioteca digital da *SCOPUS* e com buscas

manuais em eventos como o Simpósio Brasileiro de Engenharia de Software, para assim incluir também trabalhos em português.

É importante destacar que a aplicação dos critérios de inclusão e exclusão foi realizada por uma única pessoa, sob a supervisão de sua orientadora para dirimir algumas dúvidas encontradas no processo, o que se torna uma limitação deste trabalho. Sendo assim, um dos trabalhos futuros previstos é a repetição desse estudo por outros pesquisadores para confirmar seus resultados e aumentar sua confiabilidade. Outra sugestão de trabalho futuro seria analisar os trabalhos de teste também em relação à sua maturidade em termos de método científico, analisando os métodos utilizados por eles (estudo de caso, experimento, etc).

REFERÊNCIAS BIBLIOGRÁFICAS

ANAND, Saswat et al. Automated concolic testing of smartphone apps. In: **Proceedings of the ACM SIGSOFT 20th International Symposium on the Foundations of Software Engineering**. ACM, 2012. p. 59.

AMALFITANO, Domenico et al. Using GUI ripping for automated testing of Android applications. In: **Proceedings of the 27th IEEE/ACM International Conference on Automated Software Engineering**. ACM, 2012. p. 258-261.

AMALFITANO, Domenico et al. Considering context events in event-based testing of mobile applications. In: **Software Testing, Verification and Validation Workshops (ICSTW), 2013 IEEE Sixth International Conference on**. IEEE, 2013. p. 126-133.

BASTOS, Anderson; RIOS, Emerson; CRISTALLI, Ricardo; MOREIRA, Trayahú. **Base de conhecimento em teste de software**. Martins Fontes, 2007.

BURNETTE, Ed. Hello, Android: introducing Google's mobile development platform. Pragmatic Bookshelf, 2009.

CRESPO, Adalberto Nobiato et al. Teste de Software no Desenvolvimento Colaborativo. 2009.

DELAMARO, Márcio Eduardo; MALDONADO, José Carlos; JINO, Mario. **Introdução ao teste de software**. Elsevier, 2007.

DE FREITAS, Fabrício Gomes et al. Otimização em teste de software com aplicação de meta heurísticas. **Revista de Sistemas de Informação da FSMA**, n. 5, p. 3-13, 2010. Disponível em: http://www.fsma.edu.br/si/edicao5/FSMA_SI_2010_1_Estudantil_1.pdf >. Acessado em 02/09/2014.

DE SENA RAMOS, Erivan; BRASIL, Marcia Maria Albuquerque. Um Mapeamento Sistemático sobre Padrões de Software para Reengenharia de Sistemas.

FELT, Adrienne Porter et al. Android permissions demystified. In: **Proceedings of the 18th ACM conference on Computer and communications security**. ACM, 2011. p. 627-638.

FUNÇÃO, Cleomar Dias. Teste de Software: Desmembramento do Desenvolvimento. **Revista Eduf@ tima**, v. 2, n. 1, 2011.

FRANKE, Dominik et al. Testing Conformance of Life Cycle Dependent Properties of Mobile Applications. In: **Software Testing, Verification and Validation (ICST), 2012 IEEE Fifth International Conference on**. IEEE, 2012. p. 241-250.

FRANZEN, Marcelo Becker; BELLINI, Carlo Gabriel Porto. Arte ou prática em teste de software?. **Revista Eletrônica de Administração**, v. 11, n. 3, 2005.

GE, Hu et al. Malicious Code Detection for Android Using Instruction Signatures. In: **Service Oriented System Engineering (SOSE), 2014 IEEE 8th International Symposium on**. IEEE, 2014. p. 332-337.

GRIEBE, Tobias; GRUHN, Volker. A model-based approach to test automation for context-aware mobile applications. In: **Proceedings of the 29th Annual ACM Symposium on Applied Computing**. ACM, 2014. p. 420-427.

HENRIQUE, M. S. Identificação de objetos de aprendizagem para apoiar o Ensino de Programação Orientada a Objetos. Rio Tinto, 2013. Monografia (Graduação em Ciências da Computação) Universidade Federal da Paraíba.

HORNYACK, Peter et al. These aren't the droids you're looking for: retrofitting android to protect data from imperious applications. In: **Proceedings of the 18th ACM conference on Computer and communications security**. ACM, 2011. p. 639-652.

HU, Cuixiong; NEAMTIU, Iulian. Automating GUI testing for Android applications. In: **Proceedings of the 6th International Workshop on Automation of Software Test**. ACM, 2011. p. 77-83.

HU, Gang et al. Efficiently, effectively detecting mobile app bugs with AppDoctor. In: **Proceedings of the Ninth European Conference on Computer Systems**. ACM, 2014. p. 18.

JENSEN, Casper S.; PRASAD, Mukul R.; MØLLER, Anders. Automated testing with targeted event sequence generation. In: **Proceedings of the 2013 International Symposium on Software Testing and Analysis**. ACM, 2013. p. 67-77.

JIANG, Danyang et al. A security assessment method for Android applications based on permission model. In: **Cloud Computing and Intelligent Systems (CCIS), 2012 IEEE 2nd International Conference on**. IEEE, 2012. p. 701-705.

KAASILA, Jouko et al. Testdroid: automated remote UI testing on Android. In: **Proceedings of the 11th International Conference on Mobile and Ubiquitous Multimedia**. ACM, 2012. p. 28.

KITCHENHAM, Barbara A.;CHARTERS, Stuart. Guidelines for performing systematic literature reviews in software engineering. 2007.

KITCHENHAM, Barbara A.; BUDGEN, David; BRERETON, P. The value of mapping studies-a participant-observer case study. **Proceedings of Evaluation and Assessment of Software Engineering, EASE**, 2010.

KHOMH, Foutse; YUAN, Hao; ZOU, Ying. Adapting linux for mobile platforms: An empirical study of android. In: **Software Maintenance (ICSM), 2012 28th IEEE International Conference on**. IEEE, 2012. p. 629-632.

LINNELL, Natalie; BAREISS, Ray; PANTIC, Kristoffer. A wizard of Oz tool for Android. In: **Proceedings of the 14th international conference on Human-computer interaction with mobile devices and services companion**. ACM, 2012. p. 65-70.

LIU, Chien Hung et al. Capture-Replay Testing for Android Applications. In: **Computer, Consumer and Control (IS3C), 2014 International Symposium on.** IEEE, 2014. p. 1129-1132.

Lu Lu; Yulong Hong; Ying Huang; Kai Su; Yuping Yan. Activity Page Based Functional Test Automation for Android Application, In: **Software Engineering (WCSE), 2012 Third World Congress on** , vol., no., pp.37,40, 6-8 Nov. 2012

MA, Xiaoxiao et al. Design and Implementation of a Toolkit for Usability Testing of Mobile Apps. **Mobile Networks and Applications**, v. 18, n. 1, p. 81-97, 2013.

MAJI, Amiya Kumar et al. An empirical study of the robustness of inter-component communication in Android. In: **Dependable Systems and Networks (DSN), 2012 42nd Annual IEEE/IFIP International Conference on.** IEEE, 2012. p. 1-12.

MIRZAEI, Nariman et al. Testing Android apps through symbolic execution. **ACM SIGSOFT Software Engineering Notes**, v. 37, n. 6, p. 1-5, 2012.

MITCHELL, Michael; TIAN, Guanyu; WANG, Zhi. Systematic audit of third-party android phones. In: **Proceedings of the 4th ACM conference on Data and application security and privacy**. ACM, 2014. p. 175-186.

MOLINARI, Leonardo. **Testes de software: produzindo sistemas melhores e mais confiáveis: qualidade de software: soluções, técnicas e métodos**. Érica, 2003.

NAIK, Kshirasagar et al. Categorizing conFiguration parameters of smartphones for energy performance testing. In: **Proceedings of the 9th International Workshop on Automation of Software Test**. ACM, 2014. p. 15-21.

P. Bourque and R.E. Fairley, eds., Guide to the Software Engineering Body of Knowledge, Version 3.0, IEEE Computer Society, 2014.

PEREIRA, LUCIO CAMILO OLIVA; DA SILVA, MICHEL LOURENÇO. **Android para desenvolvedores**. Brasport, 2009.

PETERSEN, Kai et al. Systematic mapping studies in software engineering. In: **12th International Conference on Evaluation and Assessment in Software Engineering**. 2008. p. 1.

PRESSMAN, Roger S. Engenharia de Software – Uma Abordagem Profissional. 7ª Edição, 2011.

PRESSMAN, Roger S. Engenharia de Software – Uma Abordagem Profissional. (Versão Traduzida). 3ª Edição, 1992.

SASNAUSKAS, Raimondas; REGEHR, John. Intent fuzzer: crafting intents of death. In: **Proceedings of the 2014 Joint International Workshop on Dynamic Analysis (WODA) and Software and System Performance Testing, Debugging, and Analytics (PERTEA)**. ACM, 2014. p. 1-5.

SELLAN, Eduardo.Continentes do mundo em português, 2011. Disponível em:http://pt.wikipedia.org/wiki/Continente#mediaviewer/File:Continentes_do_mundo_em_portugu%C3%AAs.PNG Acessado em: 15/09/2014.

SILVA, G. S. Mapeamento Sistemático de Arquiteturas em Sistemas de Telemedicina. João Pessoa, 2012. Monografia (Graduação em Ciências da Computação) – Centro de Informática, Universidade Federal da Paraíba.

SHAH, Mohit et al. A top-down design methodology using virtual platforms for concept development. In: **Quality Electronic Design (ISQED), 2012 13th International Symposium on**. IEEE, 2012. p. 444-450.

STAROV, Oleksii; VILKOMIR, Sergiy. Integrated TaaS platform for mobile development: Architecture solutions. In: **Automation of Software Test (AST), 2013 8th International Workshop on**. IEEE, 2013. p. 1-7.

SUTTON, Michael; GREENE, Adam; AMINI, Pedram. Fuzzing: brute force vulnerability discovery. Pearson Education, 2007.

TAKALA, Tommi; KATARA, Mika; HARTY, Julian. Experiences of system-level model-based GUI testing of an Android application. In: **Software Testing, Verification and Validation (ICST), 2011 IEEE Fourth International Conference on**. IEEE, 2011. p. 377-386.

VAN DER MERWE, Heila; VAN DER MERWE, Brink; VISSER, Willem. Verifying android applications using Java PathFinder. **ACM SIGSOFT Software Engineering Notes**, v. 37, n. 6, p. 1-5, 2012.

YAN, Dacong; YANG, Shengqian; ROUNTEV, Atanas. Systematic testing for resource leaks in Android applications. In: **Software Reliability Engineering (ISSRE), 2013 IEEE 24th International Symposium on**. IEEE, 2013. p. 411-420.

YE, Hui et al. DroidFuzzer: Fuzzing the Android Apps with Intent-Filter Tag. In: **Proceedings of International Conference on Advances in Mobile Computing & Multimedia**. ACM, 2013. p. 68.

YING-Dar Lin; Chu, E.T.-H.; Shang-Che Yu; Yuan-Cheng Lai, "Improving the Accuracy of Automated GUI Testing for Embedded Systems," *Software, IEEE* , vol.31, no.1, pp.39,45, Jan.-Feb. 2014

ZAEEM, Razieh Nokhbeh; PRASAD, Mukul R.; KHURSHID, Sarfraz. Automated Generation of Oracles for Testing User-Interaction Features of Mobile Apps. In: **Software Testing, Verification and Validation (ICST), 2014 IEEE Seventh International Conference on**. IEEE, 2014. p. 183-192.

APÊNDICE A – Artigos incluídos

ID	Fonte (ACM/IEEE)	Referência Completa
P1	IEEE	GE, Hu et al. Malicious Code Detection for Android Using Instruction Signatures. In: Service Oriented System Engineering (SOSE), 2014 IEEE 8th International Symposium on. IEEE, 2014. p. 332-337.
P2	ACM	GRIEBE, Tobias; GRUHN, Volker. A model-based approach to test automation for context-aware mobile applications. In: Proceedings of the 29th Annual ACM Symposium on Applied Computing. ACM, 2014. p. 420-427.
P3	ACM	LINNELL, Natalie; BAREISS, Ray; PANTIC, Kristoffer. A wizard of Oz tool for Android. In: Proceedings of the 14th international conference on Human-computer interaction with mobile devices and services companion. ACM, 2012. p. 65-70.
P4	ACM	GUO, Chenkai et al. An automated testing approach for inter-application security in Android. In: Proceedings of the 9th International Workshop on Automation of Software Test. ACM, 2014. p. 8-14.
P5	ACM	FELT, Adrienne Porter et al. Android permissions demystified. In: Proceedings of the 18th ACM conference on Computer and communications security. ACM, 2011. p. 627-638.
P6	ACM	ANAND, Saswat et al. Automated concolic testing of smartphone apps. In: Proceedings of the ACM SIGSOFT 20th International Symposium on the Foundations of Software Engineering. ACM, 2012. p. 59.
P7	ACM	JENSEN, Casper S.; PRASAD, Mukul R.; MØLLER, Anders. Automated testing with targeted event sequence generation. In: Proceedings of the 2013 International Symposium on Software Testing and Analysis. ACM, 2013. p. 67-77.
P8	ACM	HU, Cuixiong; NEAMTIU, Iulian. Automating GUI testing for Android applications. In: Proceedings of the 6th International Workshop on Automation of Software Test. ACM, 2011. p. 77-83.
P9	ACM	NAIK, Kshirasagar et al. Categorizing conFiguration parameters of smartphones for energy performance testing. In:

		Proceedings of the 9th International Workshop on Automation of Software Test. ACM, 2014. p. 15-21.
P10	ACM	YE, Hui et al. DroidFuzzer: Fing the Android Apps with Intent-Filter Tag. In: Proceedings of International Conference on Advances in Mobile Computing & Multimedia. ACM, 2013. p. 68.
P11	ACM	HU, Gang et al. Efficiently, effectively detecting mobile app bugs with AppDoctor. In: Proceedings of the Ninth European Conference on Computer Systems. ACM, 2014. p. 18.
P12	ACM	SASNAUSKAS, Raimondas; REGEHR, John. Intent fuzzer: crafting intents of death. In: Proceedings of the 2014 Joint International Workshop on Dynamic Analysis (WODA) and Software and System Performance Testing, Debugging, and Analytics (PERTEA). ACM, 2014. p. 1-5.
P13	ACM	KAASILA, Jouko et al. Testdroid: automated remote UI testing on Android. In: Proceedings of the 11th International Conference on Mobile and Ubiquitous Multimedia. ACM, 2012. p. 28.
P14	ACM	MIRZAEI, Nariman et al. Testing Android apps through symbolic execution. ACM SIGSOFT Software Engineering Notes, v. 37, n. 6, p. 1-5, 2012.
P15	ACM	HORNYACK, Peter et al. These aren't the droids you're looking for: retrofitting android to protect data from imperious applications. In: Proceedings of the 18th ACM conference on Computer and communications security. ACM, 2011. p. 639-652.
P16	ACM	AMALFITANO, Domenico et al. Using GUI ripping for automated testing of Android applications. In: Proceedings of the 27th IEEE/ACM International Conference on Automated Software Engineering. ACM, 2012. p. 258-261.
P17	ACM	VAN DER MERWE, Heila; VAN DER MERWE, Brink; VISSER, Willem. Verifying android applications using Java PathFinder. ACM SIGSOFT Software Engineering Notes, v. 37, n. 6, p. 1-5, 2012.
P18	ACM	MA, Xiaoxiao et al. Design and Implementation of a Toolkit for Usability Testing of Mobile Apps. Mobile Networks and Applications, v. 18, n. 1, p. 81-97, 2013.
P19	IEEE	TAKALA, Tommi; KATARA, Mika; HARTY, Julian. Experiences of system-level model-based GUI testing of an Android application. In: Software Testing, Verification and Validation (ICST), 2011 IEEE Fourth International

		Conference on. IEEE, 2011. p. 377-386.
P20	IEEE	SHAH, Mohit et al. A top-down design methodology using virtual platforms for concept development. In: Quality Electronic Design (ISQED), 2012 13th International Symposium on. IEEE, 2012. p. 444-450.
P21	IEEE	Ying-Dar Lin; Chu, E.T.-H.; Shang-Che Yu; Yuan-Cheng Lai, "Improving the Accuracy of Automated GUI Testing for Embedded Systems," Software, IEEE , vol.31, no.1, pp.39,45, Jan.-Feb. 2014
P22	IEEE	YAN, Dacong; YANG, Shengqian; ROUNTEV, Atanas. Systematic testing for resource leaks in Android applications. In: Software Reliability Engineering (ISSRE), 2013 IEEE 24th International Symposium on. IEEE, 2013. p. 411-420.
P23	IEEE	FRANKE, Dominik et al. Testing Conformance of Life Cycle Dependent Properties of Mobile Applications. In: Software Testing, Verification and Validation (ICST), 2012 IEEE Fifth International Conference on. IEEE, 2012. p. 241-250.
P24	IEEE	Lu Lu; Yulong Hong; Ying Huang; Kai Su; Yuping Yan, "Activity Page Based Functional Test Automation for Android Application," Software Engineering (WCSE), 2012 Third World Congress on , vol., no., pp.37,40, 6-8 Nov. 2012
P25	IEEE	AMALFITANO, Domenico et al. Considering context events in event-based testing of mobile applications. In: Software Testing, Verification and Validation Workshops (ICSTW), 2013 IEEE Sixth International Conference on. IEEE, 2013. p. 126-133.
P26	IEEE	STAROV, Oleksii; VILKOMIR, Sergiy. Integrated TaaS platform for mobile development: Architecture solutions. In: Automation of Software Test (AST), 2013 8th International Workshop on. IEEE, 2013. p. 1-7.
P27	IEEE	JIANG, Danyang et al. A security assessment method for Android applications based on permission model. In: Cloud Computing and Intelligent Systems (CCIS), 2012 IEEE 2nd International Conference on. IEEE, 2012. p. 701-705.
P28	IEEE	ZAEEM, Razieh Nokhbeh; PRASAD, Mukul R.; KHURSHID, Sarfraz. Automated Generation of Oracles for Testing User-Interaction Features of Mobile Apps. In: Software Testing, Verification and Validation (ICST), 2014 IEEE Seventh International Conference on. IEEE, 2014. p. 183-192.

APÊNDICE B - Artigos excluídos

ID	Fonte (ACM/IEEE)	Referência Completa
P29	ACM	YANG, Wenbo et al. APKLancet: tumor payload diagnosis and purification for android applications. In: Proceedings of the 9th ACMsymposium on Information, computer and communications security.ACM, 2014. p. 483-494.
P30	ACM	LU, Long et al. Chex: statically vetting android apps for component hijacking vulnerabilities. In: Proceedings of the 2012 ACMconference on Computer and communications security. ACM, 2012.p. 229-240.
P31	ACM	LINNELL, Natalie et al. Context-aware technology for improvinginteraction in video-based agricultural extension. In: Proceedings of the 3rd International Conference on Human Computer Interaction.ACM, 2011. p. 10-19.
P32	ACM	KONSTANTINIDIS, Andreas et al. Demo: a programming cloud of smartphones. In: Proceedings of the 10th international conference onMobile systems, applications, and services. ACM, 2012. p. 465-466.
P33	ACM	LARKOU, Georgios et al. Demonstration abstract: sensor mockup experiments with SmartLab. In: Proceedings of the 13th international symposium on Information processing in sensor networks. IEEE Press, 2014, p. 339-340**.**
P34	ACM	JINDAL, Abhilash et al. Hypnos: understanding and treating sleepconflicts in smartphones. In: Proceedings of the 8th ACM European Conference on Computer Systems. ACM, 2013. p. 253-266.

P35	ACM	HAMMADI, Omran Al et al. Indoor localization and guidance usingportable smartphones. In: Proceedings of the The 2012 IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technology-Volume 03. IEEE Computer Society, 2012. p. 337-341.
P36	ACM	AGHAYEV, Abutalib; DESNOYERS, Peter. Log-structured cache: trading hit-rate for storage performance (and winning) in mobile devices. In: Proceedings of the 1st Workshop on Interactions of. NVM/FLASH with Operating Systems and Workloads. ACM, 2013. p.7.
P37	ACM	ZHOU, Wenming; ZHANG, Yuqing; LIU, Xuefeng. POSTER: A new framework against privilege escalation attacks on android. In: Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security. ACM, 2013. p. 1411-1414.
P38	ACM	FURUSHO, Hiroki et al. Poster: an energy profiler for android applications used in the real world. In: Proceedings of the 10th international conference on Mobile systems, applications, and services. ACM, 2012. P. 517 518.
P39	ACM	YEH, Chao-Chun; HUANG, Shih-Kun; CHANG, Sung-Yen. A blackbox based android GUI testing system. In: Proceeding of the 11th annual international conference on Mobile systems, applications, and services. ACM. 2013. P. 529-530.
P40	ACM	KAY, Matthew et al. PVT-touch: adapting a reaction time test for touchscreen devices. In: Pervasive Computing Technologies for Healthcare (PervasiveHealth), 2013 7th International Conference on. ACM, 2013. p. 248-251.

P41	ACM	MUDANYALI, Onur et al. Smart rapid diagnostics test reader running on a cell-phone for real-time mapping of epidemics. In: Proceedings of the Second ACM Workshop on Mobile Systems, Applications, and Services for HealthCare. ACM, 2012. p. 1.
P42	ACM	ROUNTEV, Atanas; YAN, Dacong. Static Reference Analysis for GUI Objects in Android Software. In: Proceedings of Annual IEEE/ACM International Symposium on Code Generation and Optimization. ACM, 2014. p. 143.
P43	ACM	DIEWALD, Stefan et al. Towards a holistic approach for mobile application development in intelligent environments. In: Proceedings of the 10th International Conference on Mobile and Ubiquitous Multimedia. ACM, 2011. p. 73-80.
P44	ACM	MAI, Haohui et al. Verifying security invariants in ExpressOS. In: ACM SIGPLAN Notices. ACM, 2013. p. 293-304.
P45	ACM	JOURDAN, Guy-Vincent; MESBAH, Ali. Workshop on Mobile and Rich Internet Application Model Generation. In: Proceedings of the 2012 Conference of the Center for Advanced Studies on Collaborative Research. IBM Corp., 2012. p. 265-266.
P46	ACM	BAO, Xuan et al. Your reactions suggest you liked the movie: automatic content rating via reaction sensing. In: Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing. ACM, 2013. p. 197-206.
P47	ACM	HASAN, Ragib et al. Sensing-enabled channels for hard-to-detect command and control of mobile devices. In: Proceedings of the 8th ACM SIGSAC symposium on Information, computer and communications security. ACM, 2013. p. 469-480.

P48	ACM	HINDLE, Abram et al. GreenMiner: a hardware based mining software repositories software energy consumption framework. In: Proceedings of the 11th Working Conference on Mining Software Repositories. ACM, 2014. p. 12-21.
P49	ACM	HICKS, John et al. Andwellness: an open mobile system for activity and experience sampling. In: Wireless Health 2010. ACM, 2010. p. 34-43.
P50	ACM	HEDGECOCK, Will et al. RegTrack: a differential relative gps tracking solution. In: Proceeding of the 11th annual international conference on Mobile systems, applications, and services. ACM, 2013. p. 475-476.
P51	ACM	DE LUCA, Alexander et al. Back-of-device authentication on smartphones. In: of the SIGCHI Conference on Human Factors in Computing Systems. ACM, 2013. p. 2389-2398.
P52	ACM	MBOGO, Chao; BLAKE, Edwin; SULEMAN, Hussein. A mobile scaffolding application to support novice learners of computer programming. In: Proceedings of the Sixth International Conference on Information and Communications Technologies and Development: Notes-Volume 2. ACM, 2013. p. 84-87.
P53	ACM	DAVIS, Benjamin; CHEN, Hao. Retroskeleton: Retrofitting android apps. In: Proceeding of the 11th annual international conference on Mobile systems, applications, and services. ACM, 2013. p. 181-192.
P54	ACM	NAKHIMOVSKY, Yelena et al. Behind the scenes of google maps navigation: enabling actionable user feedback at scale. In: CHI'10 Extended Abstracts on Human Factors in Computing Systems. ACM, 2010. p. 3763-3768.

P55	ACM	NADKARNI, Adwait; TENDULKAR, Vasant; ENCK, William. NativeWrap: Ad Hoc Smartphone Application Creation for End Users. In Proceedings of the 2014 ACM conference on Security and privacy in wireless & mobile networks (WiSec '14). ACM, New York, NY, USA, 13-24.
P56	ACM	SIMON, Laurent; ANDERSON, Ross. PIN skimmer: inferring PINs through the camera and microphone. In: Proceedings of the Third ACM workshop on Security and privacy in smartphones & mobile devices. ACM, 2013. p. 67-78.
P57	ACM	DSOUZA, Mary M. et al. Low power techniques for an android based phone. ACM SIGARCH Computer Architecture News, v. 39, n. 2, p. 26-35, 2011.
P58	ACM	MERRILL, Jamison et al. Pretesting an mHealth intervention for atrisk adolescent girls in Soweto, South Africa: studying the additive effects of SMSs on improving sexual reproductive health & rights outcomes. In: Proceedings of the Sixth International Conference on Information and Communications Technologies and Development: Notes-Volume 2. ACM, 2013. p. 96-99.
P59	ACM	JUNG, Wook Rak et al. Potential risks of WiFi-based indoor positioning and progress on improving localization functionality. In: Proceedings of the Fourth ACM SIGSPATIAL International Workshop on Indoor Spatial Awareness. ACM, 2012. p. 13-20**.**
P60	ACM	LIU, Yepang; XU, Chang; CHEUNG, Shing-Chi. Characterizing and detecting performace bugs for smartphone applications. In Proceedings of the 36th Internacional Conference on Software Engineering (ICSE 2014). ACM, New York,

		NY, USA, 1013-1024**.**
P61	ACM	BURGUERA, Iker; ZURUTUZA, Urko; NADJM-TEHRANI, Simin. Crowdroid: behavior-based malware detection system for android. In: Proceedings of the 1st ACM workshop on Security and privacy in smartphones and mobile devices. ACM, 2011. p. 15-26.
P62	ACM	HU, Cuixiong; NEAMTIU, Iulian. A gui bug finding framework for android applications. In: Proceedings of the 2011 ACM Symposium on Applied Computing. ACM, 2011. p. 1490-1491.
P63	ACM	FLOOD, Derek; HARRISON, Rachel; IACOB, Claudia. Lessons learned from evaluating the usability of mobile spreadsheet applications. In: Human-Centered Software Engineering. Springer Berlin Heidelberg, 2012. p. 315-322.
P64	ACM	Proceedings of the 2013 ACM workshop on Mobile development lifecycle. ACM, New York, NY, USA.
P65	ACM	ALLEVATO, Anthony; EDWARDS, Stephen H. RoboLIFT: simple GUI-based unit testing of student-written android applications. In: Proceedings of the 43rd ACM technical symposium on Computer Science Education. ACM, 2012. p. 670-670.
P66	ACM	Jian Zhu, Mohammad Oliya, Hung Keng Pung, and Wai Choong Wong. 2010. SOLE: context-aware sharing of living experience in mobile environments. In Proceedings of the 8th International Conference on Advances in Mobile Computing

		and Multimedia (MoMM '10). ACM, New York, NY, USA, 358-361.
P67	ACM	CHOI, Yoonjung; HONG, Ki-Hyung. User requirements for camera-based mobile applications on touch screen devices for blind people. Springer Berlin Heidelberg, 2012.
P68	ACM	Stephen H. Edwards. 2012. Web-CAT user group (abstract only). In Proceedings of the 43rd ACM technical symposium on Computer Science Education (SIGCSE '12). ACM, New York, NY, USA, 676-676.
P69	ACM	WANG, Chih-Sheng et al. A method-based ahead-of-time compiler for android applications. In: Compilers, Architectures and Synthesis for Embedded Systems (CASES), 2011 Proceedings of the 14th International Conference on. ACM, 2011. p. 15-24.
P70	IEEE	BYUN, Youngjoon et al. Wireless Broadband Measurement in California. In: Information Technology: New Generations (ITNG), 2013 Tenth International Conference on. IEEE, 2013. p. 505-509.
P71	IEEE	DING, Sun; ZHANG, Hongyu; BENG KUAN TAN, Hee. Detecting infeasible branches based on code patterns. In: Software Maintenance, Reengineering and Reverse Engineering (CSMR-WCRE), 2014 Software Evolution Week-IEEE Conference on. IEEE, 2014. p. 74-83.
P72	IEEE	CHUDÁCEK, Václav et al. Simple scoring system for ECG quality assessment on android platform. In: Computing in Cardiology, 2011. IEEE, 2011. p. 449-451.
P73	IEEE	BARETH, Ulrich. Simulating Power Consumption of Location Tracking Algorithms to Improve Energy-Efficiency of Smartphones. In: Computer Software and Applications Conference (COMPSAC), 2012 IEEE 36th Annual. IEEE, 2012. p. 613-622.

P74	IEEE	MARTINELLI, Fabio et al. A collaborative framework for generating probabilistic contracts. In: Collaboration Technologies and Systems (CTS), 2013 International Conference on. IEEE, 2013. p. 139-142.
P75	IEEE	YI, Won-Jae; GILLILAND, Spenser; SANIIE, Jafar. Mobile ultrasonic signal processing system using Android smartphone. In: Circuits and Systems (MWSCAS), 2013 IEEE 56th International Midwest Symposium on. IEEE, 2013. p. 1271-1274