Machine Translated by Google

Tecnologia da Informação e Software 76 (2016) 92–117

Listas de conteúdo disponíveis em ScienceDirect

Tecnologia da Informação e Software

página inicial da revista: www.elsevier.com/locate/infsof

Quando e o que automatizar no teste de software? Uma revisão de literatura multivocal

Vahid Garousi a,b,ÿ, Mika V. Mäntyläc

a *Software Engineering Research Group, Department of Computer Engineering, Hacettepe University, Ancara, Turquia* b *Maral Software Engineering Consulting Corporation, Calgary, Canadá*

c *M3S, Faculdade de Tecnologia da Informação e Engenharia Elétrica, Universidade de Oulu, Oulu, Finlândia*

informações do artigo

Historia do artigo:
Recebido em 20 de outubro de 2015
Revisado em 25 de abril de 2016
Aceito em 28 de abril de 2016
Disponível on-line em 30 de abril de 2016

Palavras-chave:
Automação de testes de software
Apoio à decisão
Quando automatizar
O que automatizar
Revisão da literatura multivocal
Revisão sistemática da literatura
Estudo de Mapeamento Sistemático

abstrato

Contexto: Muitas organizações veem a automação de teste de software como uma solução para diminuir os custos de teste e reduzir o tempo de ciclo no desenvolvimento de software. No entanto, o estabelecimento de testes automatizados pode falhar se a automação de testes não for aplicada no momento certo, no contexto certo e com a abordagem apropriada.
Objetivo: As decisões sobre quando e o que automatizar são importantes, pois decisões erradas podem levar a decepções e grandes gastos errados (recursos e esforços). Para apoiar a tomada de decisão sobre quando e o que automatizar, pesquisadores e profissionais propuseram várias diretrizes, heurísticas e fatores desde os primeiros dias das tecnologias de automação de teste. À medida que o número de tais fontes aumentou, é importante categorizar sistematicamente o estado da arte e a prática atual e fornecer uma visão geral sintetizada.

Método: Para atingir o objetivo acima, realizamos um estudo Multivocal Literature Review (MLR) sobre quando e o que automatizar em testes de software. A MLR é uma forma de Revisão Sistemática de Literatura (SLR) que inclui a literatura cinzenta (por exemplo, postagens em blogs e white papers) além da literatura publicada (formal) (por exemplo, artigos de periódicos e conferências). Pesquisamos a literatura acadêmica usando o Google Scholar e a literatura cinzenta usando o mecanismo de busca regular do Google.
Resultados: Nossa MLR e seus resultados são baseados em 78 fontes, 52 das quais eram literatura cinzenta e 26 eram fontes formalmente publicadas. Usamos a análise qualitativa (codificação) para classificar os fatores que afetam as perguntas quando e o que automatizar em cinco grupos: (1) fatores relacionados ao software em teste (SUT), (2) fatores relacionados ao teste, (3) fatores relacionados à ferramenta de teste, (4) fatores humanos e organizacionais e (5) fatores transversais e outros. Os fatores individuais mais frequentes foram: necessidade de teste de regressão (44 fontes), fatores econômicos (43) e maturidade do SUT (39).

Conclusão: Mostramos que o suporte à decisão atual na automação de teste de software fornece conselhos razoáveis para a indústria e, como resultado prático desta pesquisa, o resumimos como uma lista de verificação que pode ser usada pelos profissionais. No entanto, recomendamos o desenvolvimento de abordagens sistemáticas de apoio à decisão validadas empiricamente, pois os conselhos existentes são muitas vezes não sistemáticos e baseados em evidências empíricas fracas.

© 2016 Elsevier BV Todos os direitos reservados.

1. Introdução

De acordo com uma pesquisa industrial de 2014 com 1.543 executivos de 25 países, testes e garantia de qualidade de sistemas com uso intensivo de software representam cerca de 26% dos orçamentos de TI [1], mas a falta de testes é ainda mais cara. Um estudo de 2013 da Universidade de Cambridge [2] afirma que o custo global de localizar e remover bugs de software subiu para US$ 312 bilhões anualmente e representa metade do tempo de desenvolvimento de um projeto médio.

O trabalho de teste pode ser dividido em testes automatizados e manuais. No teste manual, um testador humano assume o papel de um usuário final e executa os recursos de determinado software em teste (SUT) para garantir que seu comportamento seja o esperado. Para garantir a integridade do teste, o testador geralmente segue um plano de teste escrito que contém um conjunto de casos de teste. Teste de software automatizado significa a automatização das atividades de teste de software [3]. Em termos mais específicos, *"automação de teste é o uso de software especial (separado do software que está sendo testado) para controlar a execução de testes e a comparação dos resultados reais com os resultados previstos" [4].* A compensação entre testes automatizados e manuais depende de vários fatores. Normalmente, o primeiro instinto de adotar a automação de teste é apenas aplicá-la para fazer o que os testadores humanos estavam fazendo manualmente.

ÿ Autor correspondente.
Endereços de e-mail: vahid.garousi@hacettepe.edu.tr, vgarousi@gmail.com (V. Garousi), mika.mantyla@oulu.fi (MV Mäntylä).

http://dx.doi.org/10.1016/j.infsof.2016.04.015 0950-5849/© 2016 Elsevier BV Todos os direitos reservados.

Machine Translated by Google

No entanto, a automação não pode substituir o teste manual completamente ou eliminar custos com pessoal [5]. O estabelecimento de testes automatizados pode falhar se a automação de testes não for aplicada no contexto correto com a abordagem apropriada [3].

Quando implementada corretamente, a automação de testes pode diminuir o custo de teste e aumentar a qualidade do software. No entanto, de acordo com uma pesquisa recente chamada "World Quality Relatório 2014–2015" [1], realizado por uma empresa francesa chamada Sogeti, apenas 28% dos casos de teste são atualmente automatizados enquanto os gerentes desejam aumentar esse número no futuro. À medida que a automação de testes se torna cada vez mais comum na indústria de software, a decisões sobre quando e o que automatizar tornam-se muito importantes uma vez que decisões erradas neste contexto podem levar a decepções e grandes gastos errados (recursos e esforços).

Embora, no mundo ideal, muitas pessoas tenham uma visão de plena testes automatizados, apenas 6% dos profissionais apoiaram essa visão de acordo com uma pesquisa de 2012 [6]. Outras fontes como [7] e nosso pesquisas de práticas de teste no Canadá [8] e na Turquia [9] confirmam que praticantes pensam que não é prático (ou possível) automatizar todos os testes, devido a restrições de orçamento e tempo. Decidir quando e quais partes do sistema em teste devem ser testadas de forma automatizada maneira é uma pergunta amplamente feita [10]. A partir desta escrita (março 2016), uma busca no Google pela frase ("quando automatizar" teste de software) retorna mais de 17.500 resultados, por exemplo, há muitos de fóruns online, debates e troca de experiências sobre o tema. Além disso, a importância do tópico é destacada em um livro orientado ao profissional cujo título *"Just enough software test automation"* [11] reflete que automatizar o teste de software nem sempre é uma questão de sim ou não. Outras fontes como [12] fazem similar proposições: *"Como todas as atividades de teste, por trás da decisão de automatizar alguns testes está uma análise de custo e benefício. Se você receber a análise errado, você alocará suas fontes de forma inadequada".* Vários fatores foram discutidos tanto na literatura cinzenta (por exemplo, postagens em blogs e white papers) e a literatura publicada (formal) (por exemplo, e documentos de conferências) que impactariam o trade-off entre testes automatizados e manuais [3].

Muitos pesquisadores e profissionais abordaram a questão de quando e perguntas sobre o que automatizar em artigos técnicos, blogs online e fóruns. No entanto, nenhum estudo secundário (ou seja, 'revisão' ou artigos de pesquisa) foi publicado sobre este tópico, para revisar, reunir e sintetizar o conhecimento sobre os dois W's acima de testes automatizados (o que e quando). Para cobrir essa lacuna entre a literatura acadêmica e o desejo da indústria por conhecimento prático, realizamos um estudo Multivocal Literature Review (MLR) que abrange não só estudos acadêmicos e livros, mas também uma ampla área da literatura cinzenta (não publicada sem pesquisa) disponível online, ou seja, postagens em blogs, white papers, vídeos de apresentação e ferramentas. A MLR é uma forma de uma Revisão Sistemática da Literatura (SLR) que inclui fontes de tanto a literatura cinzenta quanto a publicada (formal). Assim, em suma, as necessidades (motivações) para o nosso estudo é apresentar um único fonte para pesquisadores e profissionais, resumindo todos os fatores, abordagens, diretrizes e opiniões baseadas na experiência wrt os dois W's de teste de software automatizado (o que e quando), um necessidade que muitos praticantes nos expressaram pessoalmente em nossas interações indústria-academia, por exemplo, [13-16].

O restante deste artigo está organizado da seguinte forma. Seção 2 apresenta os antecedentes e revisa o trabalho relacionado. Seção 3 descreve o objetivo, o método de pesquisa e as questões de revisão abordadas neste estudo. A Seção 4 discute o processo de seleção de papel e fonte. A seção 5 apresenta o mapa sistemático (ou seja, esquema de classificação) que foi desenvolvido iterativamente em nosso estudo. Seção 6 apresenta os resultados da MLR. A Seção 7 resume os resultados e as implicações da MLR para pesquisadores e profissionais e discute as limitações de nosso estudo e seus resultados. Por fim, a Seção 8 conclui este estudo e apresenta o trabalho futuro instruções. A seção de referência no final do estudo é dividida em duas partes: o conjunto das fontes analisadas no MLR é listado primeiro e depois as demais referências utilizadas neste estudo.

2. Histórico e trabalho relacionado

Nesta seção, fornecemos primeiro uma breve revisão da automação de teste e os problemas 'quando' e 'o que automatizar'. Nós então apresentar um breve histórico sobre revisões de literatura multivocal (MLRs) uma vez que é uma terminologia relativamente nova em engenharia de software. Em seguida, revisamos o trabalho relacionado.

2.1. Revisão da automação de teste e o 'quando' e 'o que fazer problemas de automação

Teste de software automatizado significa a automação de software atividades de teste geralmente conduzidas por humanos. A automação de testes é realizado usando ferramentas de software referidas como ferramentas de teste [3]. A automação de testes tem uma história de mais de duas décadas e meia, desde 1990 [17], e pode levar a muitos benefícios, por exemplo, economia de custos e maior qualidade de software [19].

Qualquer grande software comercial ou de código aberto hoje em dia inclui conjuntos de testes automatizados para verificar sua funcionalidade. Este é especialmente o caso de projetos de software que evoluem através de muitos versões, uma vez que o teste automatizado compensa mais no caso de regressão e testes repetitivos. Para muitos sistemas de grande escala, os conjuntos de testes automatizados aumentam constantemente em tamanho e complexidade. Por exemplo, a base de código da versão 2.1 do sistema operacional Android tinha 357.933 LOC de código de teste JUnit que representava 17,1% do a base de código Java do SO, ou seja, 2090.904 LOC (dados de [18]). Pode-se facilmente imaginar o grande esforço necessário para desenvolver tais um conjunto de testes em primeiro lugar, garantir sua integridade (qualidade) e co-mantê-lo junto com o código de produção.

Em uma Google Tech Talk em novembro de 2005 [19], Jeff Feldstein, em seguida, o Gerente de Desenvolvimento de Software na Cisco Systems, referido à extensão da automação de testes no contexto de roteadores construído pela Cisco e mencionou que: *"Nós projetamos um sistema de teste que provavelmente é tão complicado quanto o próprio sistema.",* referindo-se ao quantidade de conjuntos de testes automatizados desenvolvidos para um roteador específico modelo.

Como exemplo de scripts de teste automatizados, a Fig. 1(a) mostra o caso de teste de interface de usuário (UI) automatizado desenvolvido usando o Selenium ferramenta (www.seleniumhq.org) para testar um aplicativo da web de rastreamento de problemas chamado JIRA. Este caso de teste verifica o recurso de inserir um problema de JIRA. As quatro etapas conceituais de teste (configuração, exercício, verificação, e desmontagem) foram explicitamente destacados na Fig. 1(a).

Como outro exemplo de scripts de teste automatizados, a Fig. 1(b) mostra o código de teste de unidade para a plataforma Android na verificação JUnit se o número de emergência (911) está configurado corretamente. Nisso caso de teste, podemos ver que uma boa prática, o chamado *teste pattern* [20], tem sido usado na modularização do código de teste por colocando um conjunto de regras de verificação específicas em uma função chamada assertIsValidEmergencyCallerInfo() e chamando-o de este método de teste de exemplo. De acordo com muitos estudos, tem sido observou empiricamente que o uso de padrões de teste aumenta a qualidade de suítes de teste, por exemplo, [20].

Para avaliar a popularidade da automação de testes na comunidade, realizamos uma pesquisa de livros de teste de software no Google Livros motor de busca (books.google.com) e compilou uma lista de todos os testes livros e também apenas aqueles com foco em automação de testes, em uma planilha online https://goo.gl/rAbDbC. A fim de manter a carga de trabalho gerenciável, identificamos apenas a automação de testes livros publicados entre 2010 e 2014. A tendência é Fig. 2. No total, 247 livros de teste de software foram publicados em período de 1979-2014. Entre 2010 e 2014, 78 do total 113 livros de testes (69%) se concentraram na automação de testes. Isso denota a alta popularidade da automação de testes na comunidade.

Machine Translated by Google

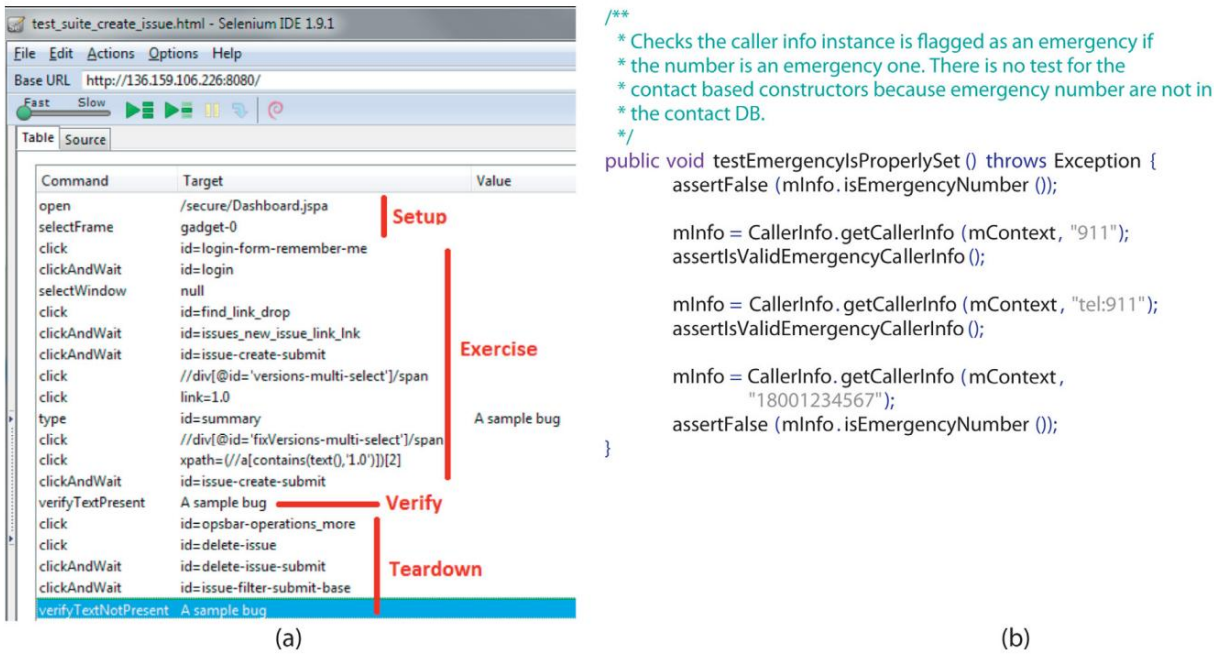


Fig. 1. Dois exemplos de casos de teste automatizados.

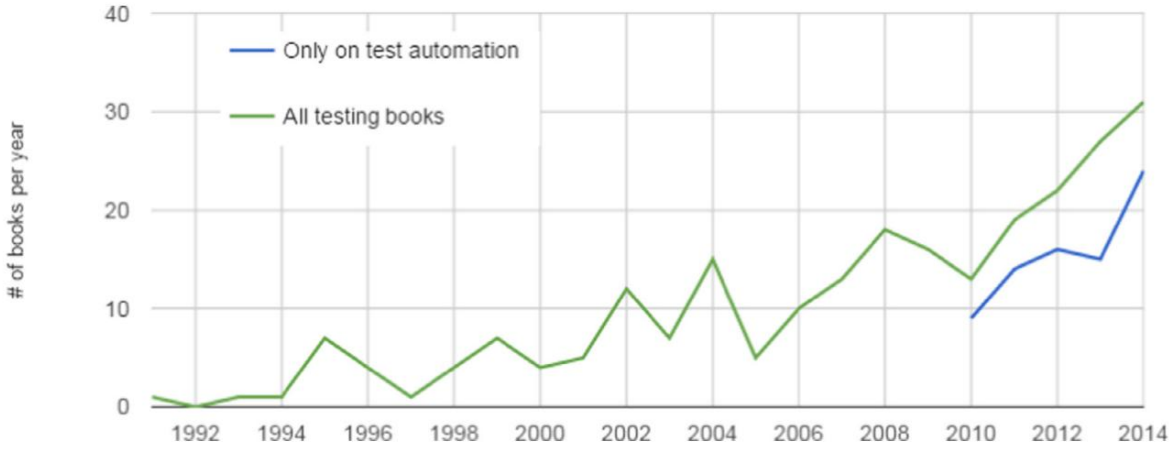


Fig. 2. Tendências de publicação de todos os livros de teste e apenas daqueles com foco em automação de teste.

Embora a automação de teste tenha começado principalmente com a automação de execução de teste, ele se expandiu para outras áreas de teste também, por exemplo, projeto de caso de teste automatizado, script de teste automatizado e relatórios de defeitos automatizados. Indo além da simples automação da execução de testes, grandes empresas como Microsoft e Google se beneficiaram muito com a automatização de diferentes atividades de teste, conforme relatado em dois livros recentes: *'How We Test Software at Microsoft' (2008)* [21] e *'Como o Google testa o software' (2012)* [22].

Normalmente, um processo de teste consiste em várias etapas, desde o planejamento do teste até a especificação do teste (projeto do caso de teste), execução e relatório [23], cada uma das quais pode ser feita manualmente ou automatizada. Para entender melhor como a automação é usada durante o processo de teste (e não apenas durante a execução), com base em muitos fontes de automação de testes, apresentamos abaixo seis atividades de teste onde um grande potencial de automação pode ser visto:

1. Projeto de caso de teste: designar uma lista de casos de teste ou requisitos de teste para satisfazer critérios de cobertura, outros objetivos de engenharia ou com base na experiência humana (por exemplo, testes exploratórios).
2. Script de teste: documentar casos de teste em scripts de teste manual ou código de teste automatizado
3. Execução de teste: execução de casos de teste no software em teste (SUT) e registrando os resultados
4. Avaliação do teste: avaliação dos resultados do teste (aprovado ou reprovado), também conhecido como veredicto de teste
5. Relatório de resultados de teste: relatar veredictos de teste e defeitos aos desenvolvedores, por exemplo, sistemas de rastreamento de defeitos (bugs)
6. Gerenciamento de teste e outras atividades de engenharia de teste: Teste o gerenciamento inclui atividades como planejamento, controle, monitoramento e estimativa de esforço. Outras atividades de teste incluem teste minimização de suítes e seleção de teste de regressão.

Uma visão geral dessas etapas é mostrada na Fig. 3, que mostra um visão geral da automação em todo o processo de teste de software.

Muitos pesquisadores e profissionais propuseram abordagens de apoio à decisão para 'quando' e 'o que automatizar', que este A MLR visa revisar, sintetizar e apresentar em uma única fonte. Nós reveja três fontes de exemplo [24–26] desse conjunto a seguir.

Ao decidir quais partes do sistema devem (não) ser automatizadas, vários fatores precisam ser considerados. Testadores de praticantes da Microsoft recomendou três fatores principais a serem considerados [24]: "(1) taxa de mudança do que estamos testando: quanto menos estável, mais custos de manutenção de automação, (2) frequência de teste

Machine Translated by Google

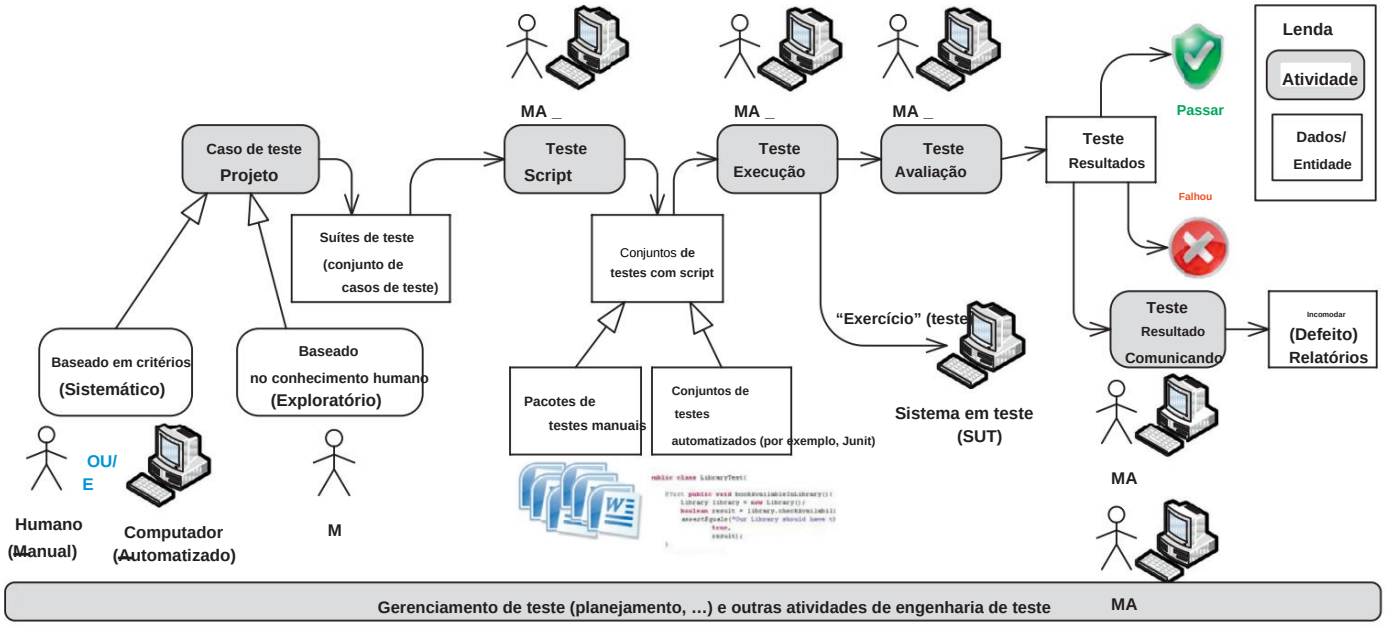


Fig. 3. Uma visão geral da automação em todo o processo de teste de software.

execução: Qual é a importância de cada resultado de teste e quão caro é conseguir?, e (3) utilidade da automação: Faça testes automatizados têm valor contínuo para encontrar bugs ou para se mostrarem importantes aspectos sobre o seu software, como cenários?"

Dentro de um livro sobre testes [25], dicas genéricas para abordar o "o que para automatizar" foram fornecidos. De acordo com este livro, certos tipos de teste, como estresse, confiabilidade e teste de regressão, são tipos passíveis de serem automatizados. Devido à natureza repetitiva do teste de regressão, a automação pode economizar tempo e esforço significativos e o tempo ganho pode ser utilizado de forma eficaz para testes ad-hoc e outras vias mais criativas [25]. Semelhante ao desafio em nosso contexto, encontrar a combinação certa de casos de teste a serem automatizados, [25] sugeriu que, ao iniciar a automação, o esforço deve se concentrar em áreas onde bons cenários em termos de ROI existem [25], no entanto, nenhuma abordagem sistemática foi forneceu.

Uma lista de verificação abrangente para apoiar a tomada de decisão para problemas 'quando-' e 'o que automatizar' foram fornecidos por outro livro sobre automação de testes [26], que considerou quase todos os fatores mencionados acima. Número de execuções, abrangendo caminhos críticos, áreas propensas a erros, recursos orientados a dados, número de hardware e software suportados e também com ROI promissor são os principais fatores discutidos neste livro [26].

2.2. Revisão de literatura multivocal

Revisão Sistemática da Literatura (SLR) e Mapeamento Sistemático (SM) estudos tornaram-se bastante populares na engenharia de software. Embora os estudos SLR e SM sejam bastante valiosos, os pesquisadores relataram que *"os resultados de um estudo SLR ou SM podem fornecer uma corpo de conhecimento estabelecido, focando apenas em contribuições de pesquisa" [27].* Como esses estudos não incluem a literatura "cinzenta" (fontes de informação não publicadas, nem revisadas por pares), produzidas constantemente em grande escala por profissionais, esses estudos não fornecem muita visão sobre o "estado da prática".

Para um campo prático (orientado ao praticante) como engenharia de software, sintetizando e combinando o estado da arte e – prática é muito importante. Infelizmente, é uma realidade que uma grande maioria dos profissionais de software não publica em fóruns acadêmicos [28], e isso significa que a voz dos profissionais é limitada se não considerarmos a literatura cinzenta além da literatura acadêmica em estudos de revisão.

2.2.1. MLRs em outros campos

SLRs que incluem tanto o acadêmico (formal) quanto o cinza literatura foram denominados no início de 1990 em outros campos, por exemplo, educação, como Multivocal Literature Reviews (MLR), por exemplo, [29]. O principal diferença entre um MLR e um SLR ou um SM é o fato de que, enquanto SLRs e SMs usam como entrada apenas artigos acadêmicos revisados por pares, em MLRs, literatura cinzenta, como blogs, white papers e páginas da web, também é considerado como entrada [27]. Uma síntese multivocal é sugerida como uma ferramenta apropriada para investigações em um campo *"caracterizado por uma abundância de documentos diversos e uma escassez de investigações sistemáticas" [30].* Os pesquisadores também acreditam que: *"outro potencial o uso de revisões de literatura multivocais está em fechar a lacuna entre a pesquisa acadêmica e a prática profissional" [29].*

Embora as noções de "MLR" e "multivocal" tenham sido usadas em a comunidade, ainda muitas fontes usam a terminologia da literatura "cinza" e se/ como incluí-los em SLRs, por exemplo, [31-33]. Por exemplo, [31] discute as vantagens e desafios de incluir literatura cinzenta em revisões de estado de evidência, no contexto de enfermagem baseada em evidências. [32] discute os desafios e benefícios pesquisa de literatura cinzenta em SLRs.

Um artigo de 1991 [29] discutiu o rigor em MLRs e propôs um método baseado em estudo de caso exploratório para conduzir MLRs com rigor. Hopewell *et ai.* [34] realizaram uma revisão de cinco estudos, em área da medicina baseada em evidências, comparando o efeito da inclusão ou exclusão de literatura 'cinza' em meta-análises de ensaios clínicos randomizados. A questão da literatura cinzenta é tão importante que existe até uma Revista Internacional sobre o tema Literatura Cinza (www.emeraldinsight.com/toc/ijgl/1/4).

2.2.2. MLRs em engenharia de software

A terminologia 'multivocal' só recentemente começou a aparecem nas SLRs em engenharia de software, ou seja, desde 2013 em [35]. Encontramos apenas três SLRs em engenharia de software que usavam explicitamente a terminologia 'multivocal': [27,35,36]. [27] é um 2015 MLR sobre o aspecto financeiro da gestão da dívida técnica. [35] é um MLR 2013 sobre dívida técnica. [36] é um MLR de 2015 em aplicativos iOS teste.

Machine Translated by Google

tabela 1

Uma lista selecionada de 15 dos 102 estudos secundários em teste de software (a lista completa pode ser encontrada em [40]).

Tipo de estudo secundário	Área de estudo secundário	Ano de publicação	Referência
SMS	Testes baseados em pesquisa para propriedades de sistema não funcionais	2008	[41]
	Testes de linhas de produtos	2011	[42]
	Teste de interface gráfica do usuário (GUI)	2013	[43]
	Priorização de casos de teste	2013	[44]
	Engenharia de código de teste de software	2014	[45]
SLRs	Testes baseados em modelos	2007	[46]
	Teste de aceitação automatizado	2008	[47]
	Teste de mutação para programas Aspect-J	2013	[48]
	Testes de aplicativos da web	2014	[49]
	Testando software científico	2014	[50]
Pesquisas regulares	Testando máquinas de estado finito	1996	[51]
	Minimização, seleção e priorização de testes de regressão: uma pesquisa 2012		[52]
	Testes em SOA 2013		[53]
	Geração de casos de teste a partir de modelos comportamentais UML 2013		[54]
	Teste oráculos 2014		[55]

Muitos outros SLRs também incluíram a literatura cinzenta em seus revisões e não usaram a terminologia 'multivocal', por exemplo, [37]. UMA Tese de mestrado de 2012 intitulada *"Sobre a qualidade da literatura cinzenta e seu uso na síntese de informações durante revisões sistemáticas da literatura" [38]* explorou o estado de inclusão da literatura cinzenta em SLRs em software Engenharia. Duas das perguntas de pesquisa (RQs) nesse estudo foram: (1) Qual é a extensão do uso da literatura cinzenta em SLRs? e (2) Como podemos avaliar a qualidade da literatura cinzenta? O estudo descobriu que a proporção de evidência cinza nas SLRs era apenas cerca de 9%, e a literatura cinzenta incluída concentrada principalmente nos últimos passado (~48% entre os anos de 2007-2012).

Um trabalho recente em andamento [39], no qual os autores estão envolvidos, visando aumentar a necessidade de (mais) MLRs na engenharia de software. Fizemos isso levantando e abordando dois RQs: (1) Que tipos de conhecimento são perdidos quando uma SLR não inclui a literatura multivocal em um campo de SE? e (2) O que nós, como comunidade, ganho quando incluímos a literatura multivocal e realizamos MLRs? Para responder a esses RQs, amostramos vários exemplos de SLRs e MLRs, e identificou o conhecimento perdido e adquirido devido à exclusão ou inclusão da literatura multivocal.

2.3. Revisão de estudos secundários em teste de software

Não foram publicados estudos secundários sobre quando e quando perguntas sobre o que automatizar. Assim, como o trabalho relacionado, apresentamos brevemente revisar os estudos secundários em teste de software. Por uma literatura pesquisa, conseguimos identificar um grande número (102) de estudos secundários em teste de software, que listamos em uma planilha on-line [40]. Os estudos secundários são geralmente de três tipos: estudos SM, e SLRs e pesquisas regulares. Como um instantâneo, mostramos uma lista selecionada aleatoriamente de 15 dos 102 estudos secundários em testes de software na Tabela 1, cinco em cada uma das três categorias acima.

Nenhum estudo secundário foi relatado no escopo exato de quando e o que automatizar no teste de software. Um trabalho um tanto relacionado é [6] que relatou uma SLR e uma pesquisa de praticantes sobre os benefícios e limitações do teste de software automatizado. Além disso, outra grande diferença entre nosso trabalho e [6] é que [6] não incluiu literatura cinzenta, enquanto o fazemos em nosso trabalho.

Ao ver uma grande lista de 102 estudos secundários em teste de software, pode-se perguntar sobre o "valor" (benefício) de tais estudos secundários. estudos. Analisando e discutindo o uso e a utilidade de SLRs em engenharia de software, em geral, está fora do escopo do nosso trabalho, mas tocamos brevemente neste tópico. Kitchenham *et ai.* [56] discutiram o valor educacional do SM na literatura de engenharia de software para os alunos. A utilidade de SLRs para profissionais tem sido estudou em vários campos não-SE, como na pesquisa de deficiência [57], em pesquisa em educação [58] e em saúde e assistência social [59].

3. Objetivo e método de pesquisa

A seguir, uma visão geral do nosso método de pesquisa e, em seguida, o objetivo e as questões de revisão do nosso estudo são apresentados.

3.1. Perguntas sobre metas e revisões

O objetivo deste estudo é mapear (classificar), revisar sistematicamente e sintetizar o estado da arte e a prática para responder às questões de quando e o que automatizar no teste de software, para descobrir as tendências e direções recentes neste campo, para identificar oportunidades para pesquisas futuras, do ponto de vista de pesquisadores e praticantes da área. Com base no objetivo acima, levantamos as seguintes questões de revisão (RQs). Observe que anterior Os estudos SM e SLR chamaram esses tipos de perguntas como "pesquisa" questões, mas com base na terminologia estabelecida em outros campos, como na pesquisa em educação [58], nos referimos a eles como "revisão" perguntas em vez disso.

- RQ 1-mapeamento de fontes por contribuição e facetas de pesquisa:
  - ÿ RQ 1.1- mapeamento por faceta de contribuição: Quantos estudos apresentar métodos, técnicas, ferramentas, modelos, métricas ou processos para o quando/o que automatizar perguntas? Mapeamento de estudos por faceta de contribuição é uma prática usual em muitos Estudos SM, por exemplo, [60-62], como proposto por Petersen et al. [63,64]. Responder a este QR nos permitirá avaliar se a comunidade como um todo teve mais foco no desenvolvimento de novas técnicas, ou mais foco no desenvolvimento de novas ferramentas, etc.
  - ÿ RQ 1.2-mapeamento por faceta de pesquisa: Que tipo de pesquisa métodos têm sido utilizados nos estudos nesta área? Algum estudos apenas propõem soluções sem validações extensas, enquanto alguns outros estudos apresentam uma avaliação aprofundada de sua abordagem (por exemplo, estudos empíricos rigorosos). Petersen et al. [63,64] também propôs diretrizes para classificar a abordagem de pesquisa dos estudos, que usaremos para responder a isso RQ. A lógica por trás deste RQ é que conhecer a divisão da área de pesquisa em relação aos tipos de facetas de pesquisa (wrt) nos fornecerá a maturidade do campo em usando abordagens empíricas.
- RQ 2-fatores considerados para decidir quando/o que automatizar (a contribuição central deste estudo): Que fatores são considerados nas perguntas quando/que? Este RQ é a principal contribuição deste estudo em particular para os praticantes, pois sintetiza todos os fatores discutidos nas fontes primárias. Nós visam agrupar e calcular a frequência de tais fatores.
- RQ 3-tools: Quais ferramentas foram propostas para apoiar o quando/que perguntas?

Machine Translated by Google

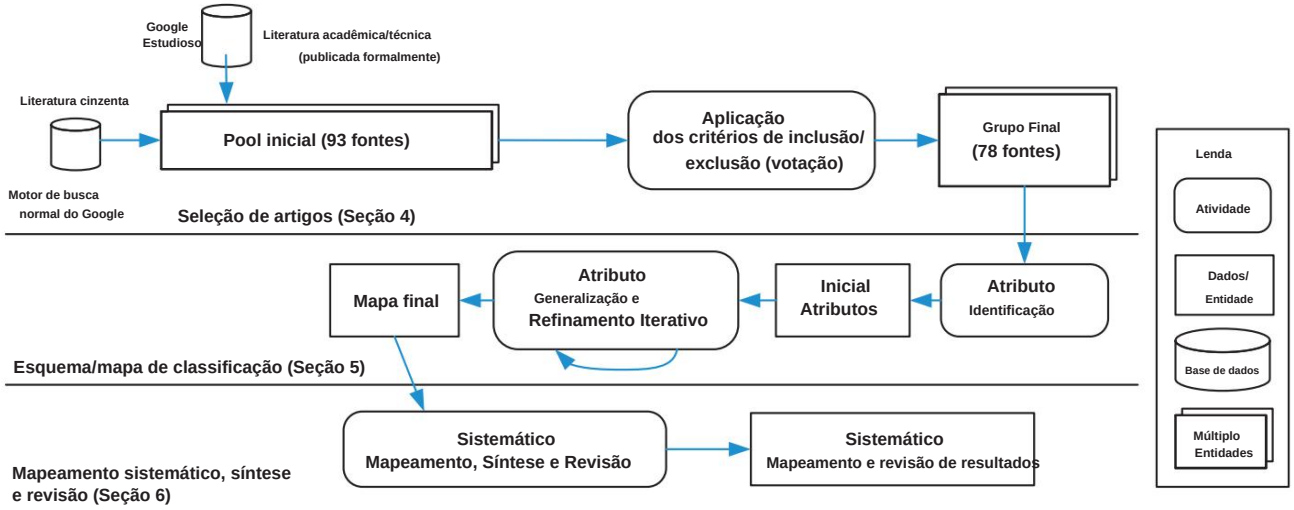


Fig. 4. Uma visão geral do processo de pesquisa usado para conduzir este estudo.

- Sistemas de software RQ 4 em teste ou projetos em estudo: Quais são os atributos desses sistemas e projetos?
  - ÿ Sistemas ou projetos de software RQ 4.1 em análise: Como muitos sistemas de software ou projetos em análise foram usado em cada fonte? Seria de esperar que um determinado papel ou artigo aplica a ideia proposta a pelo menos um sistema para mostrar sua eficácia.
  - ÿ RQ 4.2- domínios e tipos de sistemas de software ou projetos em análise: Quais são os domínios dos sistemas de software ou projetos em análise que foram estudados nas fontes (por exemplo, incorporado, crítico de segurança e software de controle)? Além disso, que tipos de sistemas de software houve estudos nas fontes (ou seja, sistemas experimentais de código aberto, comercial ou acadêmico)?
  - ÿ RQ 4.3-tipos de medições: Que tipos de medições, no contexto dos sistemas de software em análise, para apoiar as questões de quando/que foram fornecidas? Queríamos saber a proporção de fontes que realizaram medições quantitativas para avaliar as abordagens propostas.

3.2. Método de pesquisa

Como discutido acima, este estudo é realizado com base na orientações fornecidas por [65,66]. Ao desenhar a metodologia, métodos de vários outros estudos SM e SLR, como [67-69] também foram incorporados. O processo que está na base da este estudo é descrito na Fig. 4, que consiste em três fases:

- Seleção de artigos (Seção 4).
- Desenvolvimento do mapa sistemático (Seção 5).
- Mapeamento sistemático, síntese e revisão (Seção 6).

O processo começa com a seleção de artigos de diversos fontes. Em seguida, um mapa sistemático é desenvolvido sistematicamente. o mapa sistemático é então usado para realizar mapeamento sistemático e os resultados são então sintetizados e relatados. Detalhes do acima fases são descritas nas Seções 4, 5 e 6.

4. Seleção de artigos publicados formalmente e fontes do literatura cinzenta

Lembremos do nosso processo de MLR (Fig. 4) que a primeira fase do nosso estudo é a seleção de artigos. Para esta fase, seguimos o seguintes passos em ordem:

- Seleção de fonte e palavras-chave de pesquisa (Seção 4.1).
- Aplicação de critérios de inclusão e exclusão que foi realizado por meio de votação entre os dois autores (Seção 4.2).
- Finalizando o pool de artigos e o repositório online (Seção 4.3).

4.1. Seleção de fonte e palavras-chave de pesquisa

Este estudo seguiu as diretrizes sistemáticas para a realização de revisão sistemática da literatura e estudos de mapeamento [63,64,66]. Usando as diretrizes e também com base em nossa experiência anterior [6,43,70], primeiro definimos o conjunto de questões de revisão (RQs), o escopo do estudo e as strings de pesquisa.

Em seguida, realizamos as buscas no mecanismo de busca Google (para a literatura cinzenta) e em um mecanismo de busca acadêmico popular (ou seja, Google Scholar) para os artigos formalmente publicados, artigos incluídos/excluídos e, finalmente, extraímos, analisamos e sintetizamos os fatores dos artigos relevantes. Os dois autores conduziram todas as etapas em equipe.

Nossas strings de busca foram:

(a) “quando automatizar” testes.
(b) teste “o que automatizar”.
(c) teste de software automatizado de decisão.
(d) automação de teste de software de decisão.

Detalhes de nossa seleção de fontes e abordagem de palavras-chave de pesquisa foram os seguintes. Ambos os autores fizeram buscas independentes com o strings de busca, e durante esta busca, ambos os autores já aplicaram critério de inclusão/exclusão para incluir apenas aqueles que abordavam explicitamente qualquer uma das duas questões de W. Como resultado da fase inicial de pesquisa, acabamos com um conjunto inicial bastante pequeno de 93 artigos, dos quais 15 foram posteriormente removidos durante a análise e fases de exclusão.

Normalmente em estudos SM e SLR, uma equipe de pesquisadores inclui todos os resultados da pesquisa no pool inicial e, em seguida, separadamente, realiza a inclusão/exclusão como uma etapa separada. Isto resulta em enormes volumes de papéis irrelevantes. Por exemplo, em uma pesquisa de SLR e praticante em que o segundo autor estava envolvido [6], o equipe de pesquisadores começou com um conjunto inicial de 24.706 artigos mas desses apenas 25 foram considerados relevantes no final. Isso significa alto esforço devido à seleção e filtragem muito relaxada no primeira fase. No entanto, por outro lado, em dois outros estudos de SM em que o primeiro autor estava envolvido, ou seja [43,70], a filtragem inicial foi mais rigorosa e a redução dos conjuntos de papéis foi

Machine Translated by Google

como segue: (1) de um pool inicial de 230 artigos para um pool final de 136 artigos em [43], e (2) de um pool inicial de 72 artigos para um pool final de 60 artigos em [70]. Nesses últimos estudos, as equipes de pesquisadores consideraram o processo mais eficaz e eficiente, ao mesmo tempo em que a qualidade da seleção e os resultados não foram impactados. Assim, também seguimos a mesma abordagem usada em [43,70] neste estudo.

Também utilizamos o ranking de relevância dos mecanismos de busca (por exemplo, algoritmo PageRank do Google) para restringir o espaço de busca. Por exemplo, se alguém aplicar a string de pesquisa (c) acima ao mecanismo de pesquisa do Google, 1.330.000 resultados serão exibidos no momento da redação deste artigo (abril de 2015), mas, de acordo com nossas observações, os resultados relevantes geralmente aparecem apenas nas primeiras páginas. Assim, verificamos as primeiras 10 páginas (ou seja, um pouco de efeito de "saturação" da pesquisa) e só continuamos se necessário, por exemplo, quando os resultados na 10ª página ainda pareciam relevantes.

Normalmente, na maioria dos estudos SM e SLR [63,64,66], as pesquisas são direcionadas apenas para artigos na literatura acadêmica [63,64]. Como a grande maioria dos praticantes de software não publica em fóruns acadêmicos [28], isso significa que a voz dos praticantes é limitada se não considerarmos a literatura cinza além da literatura acadêmica. Assim, incluímos postagens em blogs e outros tipos de literatura cinzenta (por exemplo, white papers e até vídeos confiáveis do YouTube). Embora os blogs geralmente não se qualifiquem como evidência científica, nós os vemos como importantes canais através dos quais a voz dos profissionais pode ser estudada. Acreditamos que a inclusão da literatura cinzenta aumenta a relevância e utilidade deste estudo [71] para pesquisadores e profissionais.

A utilização da literatura cinzenta em outros campos, como as ciências da saúde, tem sido explorada, mas o campo da SE parece ser bastante marginal nesta edição e poucos estudos de SLR/SM ou artigos regulares não citam ou obtêm material da literatura cinzenta de SE.

4.2. Critérios de inclusão/exclusão e sua aplicação

Definimos cuidadosamente os critérios de inclusão e exclusão para garantir a inclusão de todas as fontes relevantes e não as fontes fora do escopo. Os critérios de inclusão foram os seguintes:

- Incluímos fontes na área de cálculos de ROI de testes automatizados, uma vez que poderiam ser usados como mecanismos de apoio à decisão para equilibrar e decidir testes de software manuais versus automatizados • Fontes que fornecem suporte à decisão para as duas perguntas "o que automatizar" e "quando automatizar"

As fontes que não atendiam aos critérios acima foram excluídas. Alguns exemplos de fontes excluídas são [3,72–74]. O título de [3] é *"Trade-off entre testes de software automatizados e manuais",* que parece muito relacionado à primeira vista, mas não aborda nenhuma das duas questões W&W, portanto, foi excluído. Intitulado *"Erros comuns na automação de testes",* [72] também não teve como alvo nenhuma das duas perguntas W&W. Incluído no primeiro grupo de candidatos, [73] relatou algumas observações empíricas sobre automação de teste de software, mas não teve como alvo nenhuma das duas perguntas W&W. Como o último exemplo, intitulado *"Sete passos para o sucesso da automação de testes",* embora [74] apresente recomendações interessantes sobre automação de testes, também não tocou em nenhuma das duas questões.

Para garantir a inclusão de todas as fontes relevantes, tanto quanto possível, realizamos uma bola de *neve* para frente e para trás [75], conforme recomendado pelas diretrizes de revisão sistemática, no conjunto de artigos já no pool. Snowballing, neste contexto, refere-se ao uso da lista de referência de um artigo (backward snowballing) ou das citações do artigo para identificar artigos adicionais (forward) [75].

4.3. Conjunto final de fontes e o repositório online

Nosso conjunto final de fontes incluiu um total de 78 fontes, que tinham: 17 fontes técnicas e científicas [Fontes 1–17], 9 livros e capítulos de livros [Fontes 18-26], 46 artigos da internet e white papers [Fontes 27–72], 2 vídeos do YouTube [Fontes 73, 74] e 4 ferramentas [Fontes 75-78] auxiliando os tomadores de decisão nas questões de quando/o quê. Para garantir a transparência, nosso repositório final pode ser encontrado em http://goo.gl/zwY1sj.

Relatamos a seguir um resumo das tendências no conjunto final de fontes, com base nestes aspectos:

- Número de fontes por ano (crescimento da atenção nesta área) por tipo de literatura (publicada formalmente versus literatura cinzenta) • Número de fontes por tipo de fonte • Número de fontes por tipo de contribuidores (contribuições de academia versus indústria)

4.3.1. Número de fontes por ano (crescimento da atenção nesta área)

A Fig. 5 mostra o número acumulado de fontes por ano por tipo de literatura (publicada formalmente versus literatura cinzenta). Podemos ver que as fontes em ambas as categorias de literatura têm apresentado uma tendência de aumento constante. A literatura cinzenta nesta área parece ter superado a literatura formalmente publicada a partir do ano de 2006, denotando a maior atenção dos profissionais sobre este importante tópico na automação de testes. Uma explicação alternativa pode estar relacionada à popularidade do Blogging que vem aumentando nos últimos anos, pois muitos profissionais de teste o estão usando para comunicação e desenvolvimento profissional. Também o fato de que a literatura cinzenta mais antiga pode não estar mais disponível devido à falta de arquivamento sistemático de tais obras provavelmente afetou a baixa quantidade de literatura cinzenta mais antiga. Podemos ver a partir da figura que o tema deve receber maior atenção nos próximos anos também.

4.3.2. Número de fontes por tipo de fonte

A Fig. 6 mostra o número de fontes por tipo de fonte em cada uma das duas categorias: formalmente publicada versus literatura cinzenta. Dividimos a literatura formalmente publicada e a literatura cinzenta, respectivamente, em cinco e três categorias, conforme mostrado.

Na categoria literatura publicada formalmente, houve 9 trabalhos de conferência, 9 livros ou capítulos de livros, 5 artigos de periódicos, 2 trabalhos de oficina e 1 tese. Na categoria literatura cinzenta, foram 46 artigos online e white papers, 4 ferramentas e 2 vídeos do YouTube.

Conforme discutido na Seção 4, uma vez que a grande maioria dos profissionais de software não publica em fóruns acadêmicos [28], isso significa que a voz dos profissionais é limitada se não considerarmos a literatura cinza além da literatura acadêmica. Este artigo dá um passo nessa direção ao incluir e revisar a base de conhecimento e a voz dos profissionais de software conforme retratada pela literatura cinza.

4.3.3. Número de fontes por tipo de contribuidores (contribuições da academia versus indústria)

A Fig. 7 mostra o número de fontes por tipo de contribuidores, em que há três categorias: (1) um artigo com todos os autores acadêmicos, (2) todos os autores da indústria e (3) uma mistura de autores acadêmicos e industriais (trabalho colaborativo). A lógica por trás dessa análise é avaliar como a academia e a indústria têm sido ativas ao abordar as questões de W&W. Principalmente devido ao elevado número de fontes da literatura cinzenta, as fontes escritas por autores da indústria são a maioria. Havia 6 fontes de autores acadêmicos [Fontes 2, 4, 6, 7, 11, 15] e apenas 3 trabalhos colaborativos [Fontes 1, 3, 17]. Estes últimos denotam a necessidade de mais colaborações acadêmico-industriais nesta área.

Machine Translated by Google

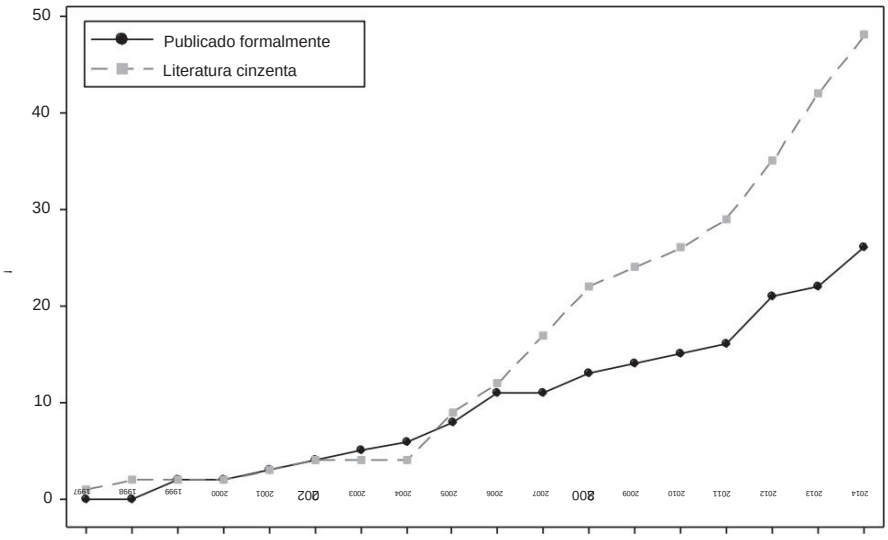


Fig. 5. Número acumulado de fontes por ano.

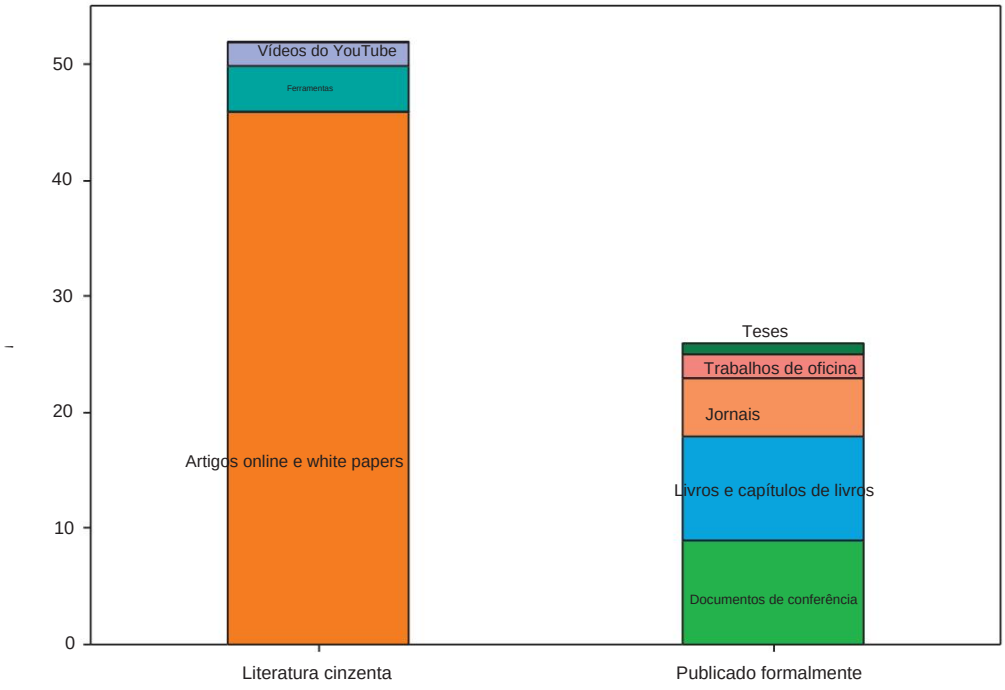


Fig. 6. Número de fontes por tipo.

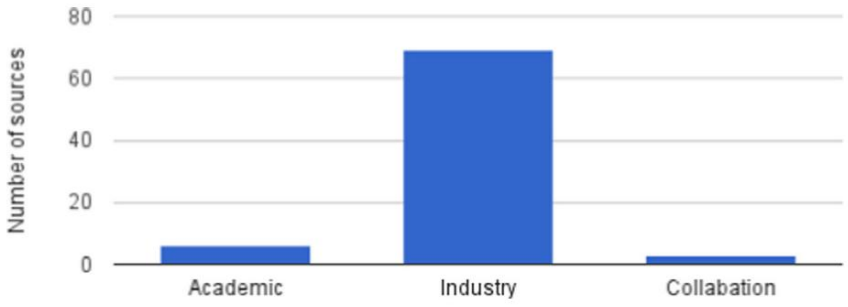


Fig. 7. Número de fontes por tipo de contribuinte.

Machine Translated by Google

mesa 2

Mapa sistemático desenvolvido e utilizado em nosso estudo.

Atributo/Aspecto RQ		Categorias	(M)múltiplo/ (S)único
1	Tipo de contribuição	Heurística/diretriz, método (técnica), ferramenta, métrica, modelo, processo, apenas resultados empíricos, outros	M
	Tipo de pesquisa	Proposta de solução, pesquisa de validação (fraco estudo), pesquisa de avaliação (estudo empírico forte), estudos de experiência, estudos filosóficos, estudos de opinião, outro	S
2	Fatores considerados para decidir quando/o que automatizar	Uma lista de categorias predefinidas (Maturidade do SUT, Estabilidade de casos de teste, 'Custo, benefício, ROI' e Necessidade de regressão testes) e uma categoria 'outra' cujos valores foram posteriormente codificado qualitativamente (aplicando codificação 'axial' 'aberta')	M
3	Ferramentas de apoio à decisão	Nome e características	M
4	Atributos dos sistemas de software em teste (SUT)	Número de sistemas de software: inteiro	M
		Nomes SUT: array de strings	
		Domínio, por exemplo, sistemas embarcados	
		Tipo de sistema(s): Acadêmico experimental ou código simples exemplos, código aberto real, comercial	
		Medições de custo/benefício de automação de teste: numéricas valores	

5. Desenvolvimento do mapa sistemático e extração de dados plano

O desenvolvimento iterativo do nosso mapa sistemático é discutido em Seção 5.1. A seção 5.2 apresenta o mapa sistemático final. Seção 5.3 discute nossa abordagem de extração e síntese de dados.

5.1. Mapa sistemático

Para desenvolver nosso mapa sistemático, conforme mostrado na Fig. 4, analisamos os estudos no pool e identificou a lista inicial de atributos. Em seguida, usamos generalização de atributos e refinamento iterativo para derivar o mapa final.

Como os estudos foram identificados como relevantes para o nosso estudo, registramos em uma planilha compartilhada (hospedada no Google Docs on-line planilha [76]) para facilitar a análise posterior. Nosso próximo objetivo era categorizar os estudos para começar a construir um quadro completo da área de pesquisa. Embora não tenhamos desenvolvido a priori um esquema de categorização para este projeto, estávamos amplamente interessados em: (1) fatores considerados para as perguntas W&W, e (2) tipos de sistemas em teste que foram usados em estudos. Nós refinamos esses interesses amplos em um mapa sistemático usando uma abordagem iterativa.

A Tabela 2 mostra o esquema de classificação final que desenvolvemos após a aplicação do processo descrito acima. Na tabela, coluna 1 é a lista de RQs, a coluna 2 é o atributo/aspecto correspondente. A coluna 3 é o conjunto de todos os valores possíveis para o atributo. Finalmente, a coluna 4 indica para um atributo se várias seleções podem ser aplicado. Por exemplo, no RQ 1 (tipo de pesquisa), o correspondente valor na última coluna é 'S' (Single). Indica que uma fonte podem ser classificados em apenas um tipo de pesquisa. Em contraste, para RQ 1 (tipo de contribuição), o valor correspondente na última coluna é 'M' (Múltiplo). Indica que um estudo pode contribuir mais de um tipo de opções (por exemplo, método, ferramenta, etc.).

Acreditamos que a maioria das categorias na Tabela 2 são autoexplicativas, exceto aquelas para contribuição e tipos de pesquisa (RQ 1) que são explicados nas próximas duas subseções.

Petersen *et ai.* [65] propuseram a classificação das contribuições em: método/técnica, ferramenta, modelo, métrica e processo. Esses tipos foram adaptados em nosso contexto. Também adicionamos outro tipo: pesquisa ou resultados empíricos, pois constatamos que muitos estudos contribuem com esses resultados. Se um estudo não puder ser categorizado em qualquer tipos acima mencionados, seria colocado em “Outros”.

O segundo conjunto de categorias em nosso esquema trata da natureza do método de pesquisa utilizado em cada fonte. Essas categorias foram influenciados pelas categorias descritas por Petersen *et al.* [65] al embora não sejam uma réplica exata. Nosso objetivo é fornecer insights na base empírica que está sendo desenvolvida pelo corpo de pesquisa. As categorias de “tipo de pesquisa” incluem:

- Proposta de solução: Um estudo nesta categoria propõe uma solução a um problema. Os benefícios potenciais e a aplicabilidade do solução são mostradas apenas por um pequeno exemplo ou uma boa linha de argumentação.
- Pesquisa de validação (estudo empírico fraco): Um estudo nesta categoria fornece evidências empíricas preliminares para a proposta técnicas ou ferramentas. Métodos empíricos formais (por exemplo, hipótese testes, estudos de caso, pesquisa de ação técnica, experimentos controlados) não são usados ou seu uso não é suficiente (por exemplo, experimentos com baixo número de sujeitos ou conduzidos de forma inadequada estudos de caso)
- Pesquisa de avaliação (estudo empírico forte): Esses estudos vão além de estudos do tipo "pesquisa de validação" usando e métodos experimentais formais (por exemplo, teste de hipótese, caso estudos, experimento controlado) na avaliação de novas técnicas ou ferramentas na prática ou prática como configurações. Nós interpretamos essencialmente a pesquisa de validação (estudo empírico fraco) e a pesquisa de avaliação (estudo empírico forte) como uma “rubrica”. Usando os exemplos dados (usando métodos empíricos formais ou não), fomos capazes de classificar um determinado estudo empírico em fraco ou forte estudo empírico.
- Estudos de experiência: estudos de experiência explicam como algo foi feito na prática, com base na experiência pessoal de Os autores).
- Estudos filosóficos: Esses estudos esboçam uma nova forma de olhar para as coisas existentes, estruturando a área em forma de taxonomia ou quadro conceitual.
- Estudos de opinião: Esses estudos expressam a opinião pessoal de o(s) autor(es) sobre se uma determinada técnica é boa ou ruim, ou como as coisas deveriam ser feitas. Eles não significativamente confiar em trabalho relacionado ou metodologias de pesquisa.
- Outros: uma categoria abrangente no caso de o trabalho ser relatado em um estudo não se encaixa em nenhum dos tipos de pesquisa acima.

5.2. Extração de dados e síntese qualitativa

Para extrair dados, os estudos em nosso pool foram revisados com o foco de cada RQ. Também incorporamos o máximo possível de ligações explícitas de 'rastreabilidade' entre nosso mapeamento e os estudos primários. A Fig. 8 mostra um instantâneo do repositório online (planilha hospedada no Google Docs) no qual as facetas de contribuição são

Machine Translated by Google

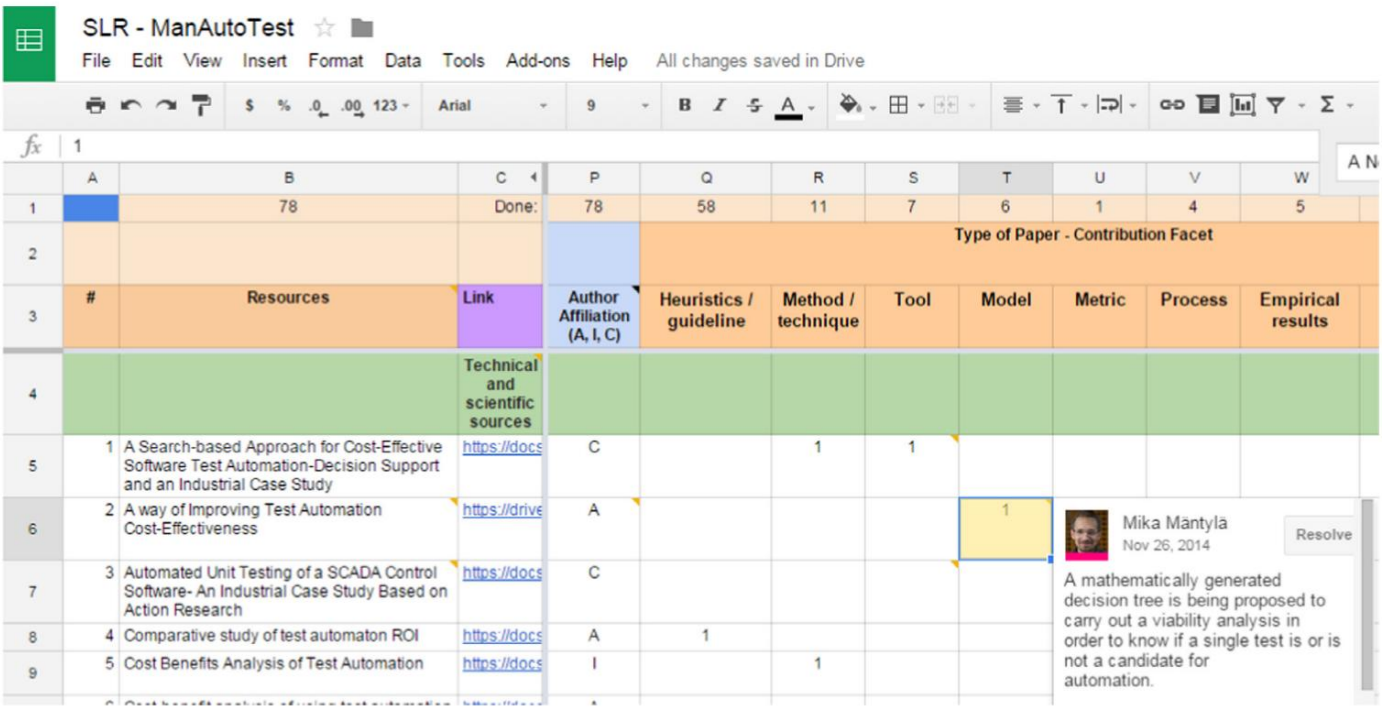


Fig. 8. Um instantâneo da planilha disponível publicamente hospedada no Google Docs.

mostrado e uma classificação de 'Modelo' para a faceta de contribuição de [Fonte 2] é mostrada, juntamente com o texto literal de copiar/colar de a fonte atuando como o link de 'rastreabilidade' correspondente.

Durante a análise, cada um dos dois pesquisadores extraiu e analisou dados de metade das fontes (atribuídas a ele), então revisaram os resultados das análises uns dos outros e também fizeram uma votação independente para incluir ou excluir cada fonte. Dentro do estojo de desacordos, discussões foram conduzidas. Isso foi realizado para garantir a qualidade e validade de nossos resultados.

Para escolher nosso método de síntese para RQ 3 (fatores considerados nas perguntas quando/o que), revisamos cuidadosamente as diretrizes de síntese de pesquisa em SE, por exemplo, [77-79], e também outras SLRs que realizou síntese de resultados, por exemplo, [80,81]. De acordo com para [77], o principal objetivo da síntese da pesquisa é avaliar a incluíram estudos para heterogeneidade e selecionaram métodos apropriados para integrar ou fornecer explicações interpretativas sobre eles [82]. Se os estudos primários forem semelhantes o suficiente em relação às intervenções e variáveis quantitativas de resultado, pode ser possível para sintetizá-los por meta-análise, que usa métodos estatísticos para combinar tamanhos de efeito. No entanto, no SE em geral e no nosso domínio focado em particular, os estudos primários são muitas vezes muito heterogêneos para permitir um resumo estatístico. Especialmente para qualidade e estudos de métodos mistos, diferentes métodos de síntese de pesquisa, por exemplo, análise temática e síntese narrativa são necessários [77].

Os fatores e sua categorização apresentados no artigo foram os resultados de uma síntese formal e sistemática feita em colaboração entre os dois pesquisadores seguindo um abordagem qualitativa de análise de dados [83]. Tínhamos alguns pré-definidos fatores (baseados em nosso conhecimento anterior da área), a saber, “teste de regressão”, “maturidade do SUT” e “ROI”. Durante o processo, nós descobrimos que nossa lista pré-determinada de fatores era muito limitante, assim, o restante dos fatores emergiram das fontes, por condução “aberta” e “codificação axial” [83]. A criação do novo fatores na fase de “codificação” foi um processo iterativo e interativo em que ambos os pesquisadores participaram. Basicamente, primeiro coletamos todos os fatores que afetam as questões de W&W das fontes.

Em seguida, procuramos encontrar fatores que representassem com precisão todos os itens extraídos, mas ao mesmo tempo não seja muito detalhado para que ainda fornecesse uma visão geral útil, ou seja, escolhemos o nível de “abstração” mais adequado, conforme recomendado por diretrizes de análise de dados [83].

A Fig. 9 mostra as fases de extração de dados qualitativos para o fatores, em que o processo partiu de uma lista de categorias pré-definidas: maturidade do SUT, estabilidade dos casos de teste, 'custo, benefício, ROI, e necessidade de testes de regressão) e um grande número de fatores 'brutos' expresso na categoria 'outro'. Por um processo iterativo, aqueles frases foram codificadas qualitativamente (aplicando 'axial' e 'aberto' abordagens de codificação [83]) para produzir o resultado final, ou seja, um conjunto coeso de fatores bem agrupados (a ser discutido na Seção 6.3).

6. Resultados

Os resultados do nosso estudo são apresentados da Seção 6.1 a 6.4.

6.1. RQ 1-mapeamento por facetas de contribuição e pesquisa

6.1.1. RQ 1.1-mapeamento de estudos por faceta de contribuição

A Fig. 10 mostra a repartição cumulativa anual e o número total de estudos primários por tipos de facetas de contribuição. Exato referências de estudo também foram fornecidas sob a figura. o As três principais facetas de contribuição são: heurística e diretrizes, métodos (também chamados de técnica ou abordagem) e métricas, que apareceram em 41 estudos (68%), 39 estudos (65%) e 12 estudos (20%), respectivamente. Observe que a maioria dos gráficos de tendência temporal em este estudo, como o mostrado na Fig. 10 , são pilha cumulativa gráficos, mostrando o número acumulado de estudos em cada ano, como acumulado de anos anteriores.

Observe ainda que, uma vez que muitos estudos foram classificados sob mais de uma faceta de contribuição, o gráfico de pilha da Fig. 10 não pode ser usado como a tendência anual do número de estudos (essa tendência foi apresentado na Seção 4.3.1). Podemos ver isso em diferentes anos, diferentes facetas de contribuição foram estudadas, e nenhuma mudança de tendência é observável.

58 fontes propuseram heurísticas e diretrizes. Conforme discutido na Seção 5.2, heurísticas e diretrizes neste contexto são

Machine Translated by Google

AH	AI	AJ	AK	AL
39	38	7	44	57
FACTORS considered (in when/what to automate)				
Stability of SUT	ROI	Test stability	Regression testing	Other
	1			
			1	Test reuse, revelevance, automation effort, resource (money, hw), manual complexity, automation tool quality, test portability, manual exec effort
1		stable enviroment and data	1	design the test case first, test engineer's skill, first manual then automatte, SUT known, risk and complexity low
	1			reduce costs and improve quality because of more testing in less time, but it causes new costs in, for example, implementation Generic and independent products facilitate and customized and complex products hinder testing automation High reusability facilitates and low reusability hinders testing automation
1			1	Easiness to automate, boringness of the test, manual test must exits first, Automate non-time dependent tests, Automate tests that have been written:
	1			

(a): initial phase of qualitative coding

SLR - ManAutoTest

File Edit View Insert Format Data Tools Add-ons Help All changes saved in Drive

High reusability facilitates and low reusability hinders testing automation

	A	B	C	D	E	F	G
1	78	Done:	78	32	17	14	18
2	Paper Title	Link	Who	Other			
3	Technical and scientific sources	.			Test reuse/ repeatability	Manual test effort	automatability
13	Pragmatic approach to software test automation	0B6dKdxaNj	v	design the test case first, first manual then automatte, SUT known, low test risk and complexity			
14	Software Test Automation in Practice: Empirical Observations	0B6dKdxaNj	v		High reusability facilitates and low reusability hinders testing automation		consider automation costs
15	Software test automation–Developing an infrastructure designed for success	lt/files/article	m			boringness of the test	automatability
16	Surviving the Top 10 Challenges of Software Test Automation	0B6dKdxaNj	v		Test repeatability		automatability

(b): intermediate phase of qualitative coding

SLR - ManAutoTest

File Edit View Insert Format Data Tools Add-ons Help All changes saved in Drive

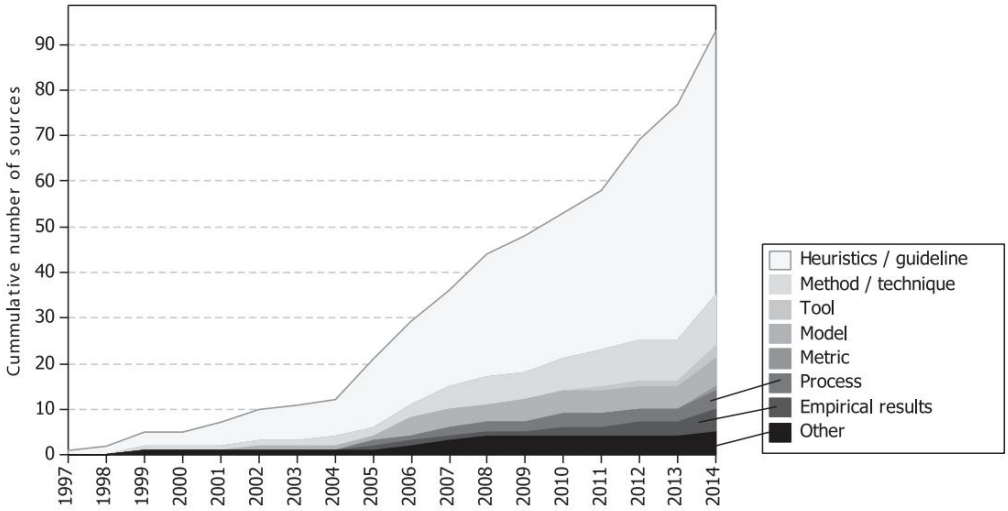
Decision support

	A	B	C	AJ	AK	AL	AM	AN
1		78	Done:	39	6	44	30	17
2				Factors considered (in when/what to automate)				
3	#	Resources	Link	Stability of SUT	Other SUT aspects	Need for regression testing	Test Type	Test reuse/ repeatability
14	10	Pragmatic approach to software test automation	https://docs	1		1	low test risk and complexity	
15	11	Software Test Automation in Practice: Empirical Observations	https://docs		SUT features (Generic and independent products facilitate and customized and complex products hinder testing automation)			High reusability facilitates and low reusability hinders testing automation
16	12	Software test automation–Developing an infrastructure designed for success	https://drive	1		1	Automate tests that have been written (manual test must exits first), Automate tests with no timing issues	
17	13	Surviving the Top 10 Challenges of Software Test Automation	https://docs		SUT criticality/risk			Test repeatability
18	14	The Return on Investment (ROI) of Test Automation	https://docs					
	15	The When & How of Test Automation	https://docs	1				

(c): the final result after qualitative coding

Fig. 9. Fases de extração de dados qualitativos para fatores considerados para decidir quando/o que automatizar.

Machine Translated by Google



Contribution facet types	Number of sources	References
Heuristics / guideline	58	Too many to be listed. Refer to the spreadsheet (http://goo.gl/zwY1sj)
Method / technique	11	[Sources 1, 5, 7, 8, 17, 43, 51, 56, 57, 61, 62]
Tool	7	[Sources 1, 17, 61, 75 -78]
Model	6	[Sources 2, 7, 17, 21, 22, 38]
Metric	1	[Sources 6]
Process	4	[Sources 15, 17, 47, 51]
Empirical results	5	[Sources 3, 6, 9, 11, 16]
Other	5	[Sources 5, 7, 31, 48, 62]

Fig. 10. Mapeamento dos estudos por faceta de contribuição.

recomendações informais sobre W&W para automatizar e geralmente são menos maduros do que métodos e técnicas que foram categorizados de forma diferente. 11 métodos/técnicas propostas de fontes abordando as questões W&W. Por exemplo, [Fonte 1] propôs um abordagem baseada em pesquisa (usando algoritmos genéticos) para suporte à decisão de automação de teste de software e um estudo de caso de teste industrial de suporte. Encontramos também quatro ferramentas online para o quando e quais perguntas na automação de teste de software [Fontes 75, 76, 78, 77]. Essas ferramentas serão brevemente revisadas quando uma resposta RQ 4 na Seção 6.4. Seis fontes propuseram modelos de apoio às questões de W&W. Por exemplo, [Fonte 2] propôs uma árvore de decisão gerada matematicamente para realizar uma viabilidade análise para saber se um caso de teste é ou não um candidato para automação. [Fonte 7] propôs um modelo de custo de oportunidade. Dentro Em termos de métricas usadas para responder às perguntas W&W, muitos estudos usaram a métrica ROI convencional. Quatro fontes [Fonte 15, 17, 47, 51] propuseram processos abordando as questões de W&W. Fig. 11 mostra o processo de referência adotado de [Fonte 17]. As contribuições de cinco fontes [Fontes 3, 6, 9, 11, 16] relacionadas ao nosso contexto foram apenas resultados empíricos e não puderam ser categorizadas sob os outros tipos de categorias de contribuições. [Fonte 6] foi uma tese de mestrado que apresentou um estudo de caso sobre a rentabilidade do teste automação no desenvolvimento de software embarcado em ambiente industrial. Cinco fontes [Fontes 5, 7, 31, 48, 62] contribuíram 'outros' tipos de contribuições. [Fonte 5] apresentou um conjunto de benefícios esperados (mitos) na automação de testes como: *"Todos os testes ser automatizado: Isso não é prático ou desejável.*

Em seguida, queríamos comparar a proporção de diferentes contribuições tipos nesta área com uma outra subárea representativa em teste. A Fig. 12 mostra o mapeamento dos estudos por faceta de contribuição em área de engenharia de código de teste de software (STCE) e neste artigo. Os dados STCE são retirados de outro estudo SM [70]. Exceto por caso de grande número de heurísticas apresentadas por fontes no MLR atual que se deve ao grande número de fontes na literatura cinzenta, as outras proporções são bastante comparáveis.

6.1.2. RQ 1.2-mapeamento de estudos por faceta de pesquisa

Com base no esquema descrito na Seção 5, classificamos o estudos em seis categorias de facetas de pesquisa. A Fig. 13 mostra a classificação de todas as fontes de acordo com o tipo de método de pesquisa eles seguiram e relataram. Lembre-se que, para a faceta de pesquisa tipo, cada estudo pode ser categorizado em uma ou mais categorias.

Uma grande proporção de fontes se enquadra em categorias baseadas em experiência e opinião, o que novamente se deve ao grande número de fontes fontes formam a literatura cinzenta em nosso pool. Há apenas um estudo empírico rigoroso (pesquisa de avaliação) [Fonte 16] e seis estudos empíricos fracos (pesquisa de validação) [Fontes 1–3, 6, 17, 61]. Na [Fonte 11] (categorizada em 'Outros' na Fig. 13), foi realizado um estudo qualitativo baseado em pesquisa em 55 especialistas em testes em 12 organizações de desenvolvimento de software selecionadas. Nós interpretamos o baixo número de fontes usando Pesquisa de Avaliação (forte estudo empírico) devido a duas possibilidades: (1) baixo, pois principalmente pesquisador conduzir estudos empíricos fortes, tem havido pouca atenção dos a comunidade de pesquisa neste domínio; (2) complexidade de realizar estudos empíricos fortes nesta área.

Também achamos que seria útil comparar a pesquisa quebra de facetas deste SM para quatro outros estudos SM em SE que utilizaram a mesma classificação [60–62,70]: aplicação web teste [60], teste de GUI [61], engenharia de código de teste de software (STCE) [70] e Linhas de Produto de Software Dinâmico (DSPL) [62]. A comparação é visualizada na Fig. 14 e mostra que as ações das facetas de pesquisa nos outros quatro estudos SM são bastante semelhantes entre si com proposta de solução, pesquisa de validação e

Machine Translated by Google

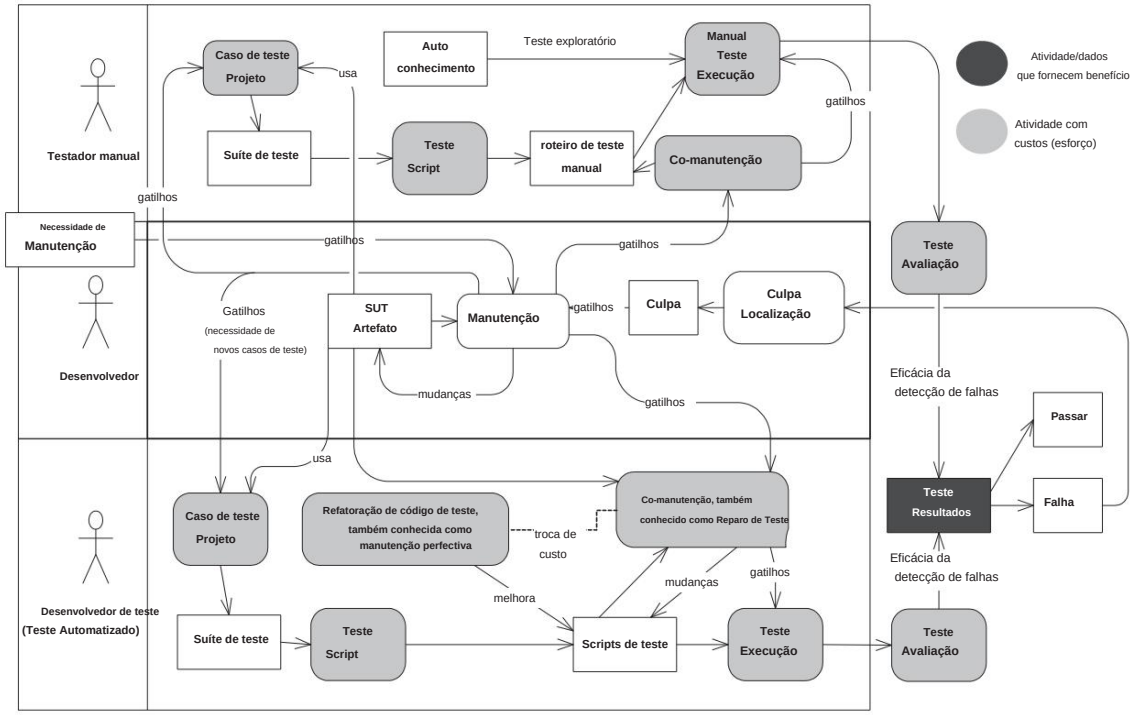


Fig. 11. Processo de referência de teste de software (adotado de [Fonte 17]).

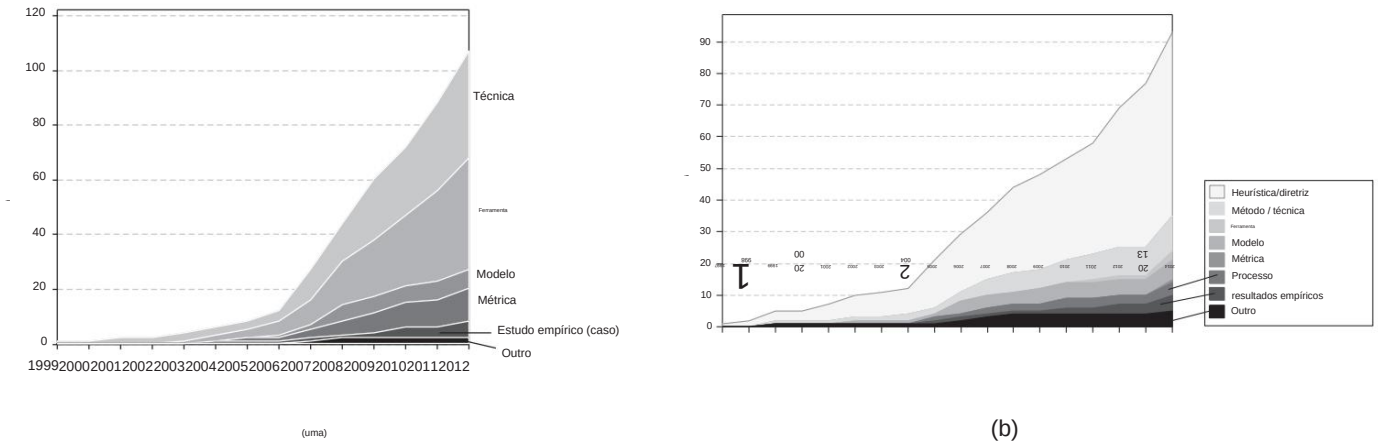


Fig. 12. Mapeamento de estudos por faceta de contribuição na área de engenharia de código de teste de software (a) (dados são de outro estudo SM [70]), e neste estudo (b).

pesquisa de avaliação com as maiores participações. No entanto, neste artigo, os índices de fontes baseadas em experiência e opinião são altos uma vez que um grande número de literaturas cinzentas (por exemplo, postagens em blogs) foram incluído.

6.2. RQ 2-fatores considerados para decidir quando/o que automatizar

Nesta Seção, apresentamos a classificação qualitativa do fatores a serem considerados ao decidir quando e o que automatizar. Achamos que esta Seção tem o maior valor para os praticantes, enquanto outras seções são voltadas mais para acadêmicos público.

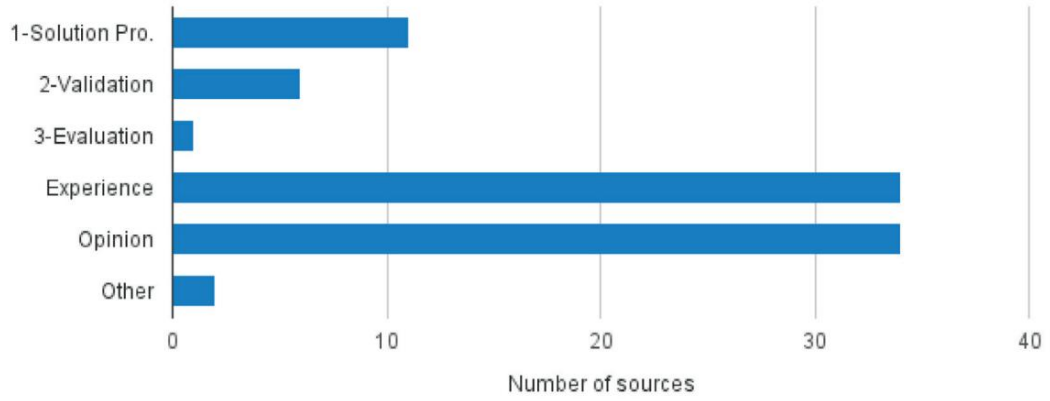
6.2.1. Uma visão simplificada da automação de teste de software

Na automação de teste, existem quatro componentes básicos (consulte Fig. 15): testadores (engenheiros de teste), ferramenta (automação de teste), casos de teste e suítes de teste e sistema em teste (SUT). Os engenheiros de teste interagem com as ferramentas de automação de teste e desenvolvem os casos de teste. Os casos de teste são então executados usando a automação de teste escolhida ferramenta. Os testes exercitam o SUT e a ferramenta fornece os relatórios de teste para humanos interpretarem. De acordo com o conjunto de fontes analisado em nosso estudo, o sucesso ou fracasso da automação de teste depende em todos esses fatores e em suas interações, como discutimos em as próximas seções. Além disso, encontramos uma série de fatores e os colocaram em uma categoria chamada “fatores transversais” (rotulado como #5 na Fig. 15) que abrange as outras quatro categorias. Exemplos de tais fatores transversais são fatores econômicos, automatização de testes, esforço de teste manual e processo de desenvolvimento, que serão discutidos posteriormente neste artigo.

6.2.2. Categorização de fatores que afetam quando e o que automatizar

Durante nossa revisão, extraímos os fatores mencionados em cada fonte e classificou-os em 15 tipos de fatores sob os cinco categorias como mostrado na Fig. 4. A frequência de diferentes fatores tipos conforme discutido nas fontes é mostrado na Fig. 2 para ilustrar o nível de atenção em cada fator. Observe que esses números não são significava ser um indicativo de importância, pois o que é importante varia

Machine Translated by Google



R esearch facet types	Number of sources	References
Solution proposal	11	Refer to the spreadsheet (http://goo.gl/zwY1sj)
Validation research	6	[Sources 1 -3, 6, 17, 61]
Evaluation research	1	[Source 16]
Experience studies	34	Refer to the spreadsheet
Opinion studies	34	Refer to the spreadsheet
Other	1	[Source 11]

Fig. 13. Mapeamento dos estudos por faceta de pesquisa.

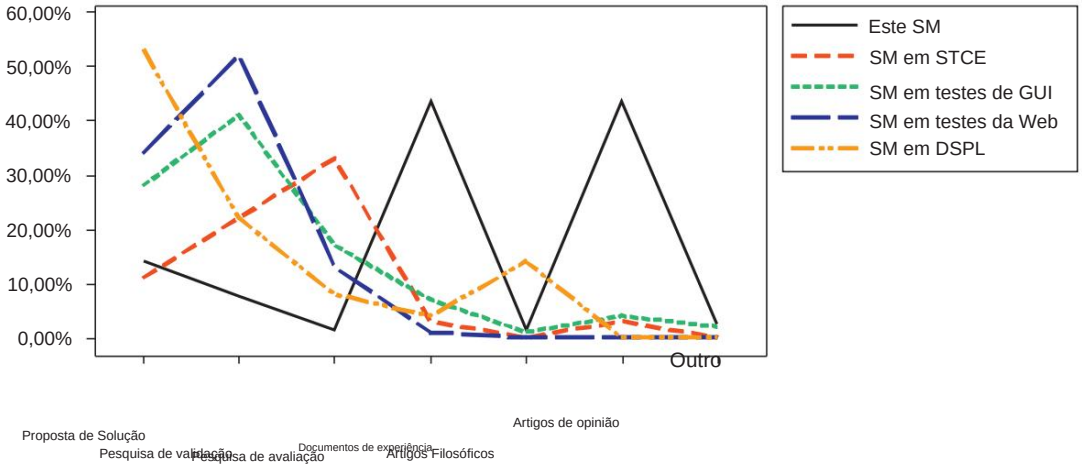


Fig. 14. Comparando o detalhamento das facetas de pesquisa deste SM com outros estudos representativos do SM [60–62,70].

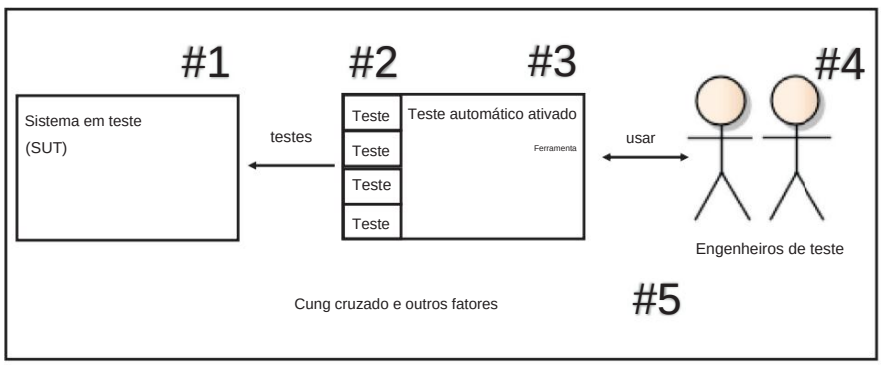


Fig. 15. Uma visão simplificada da automação de teste de software.

Machine Translated by Google

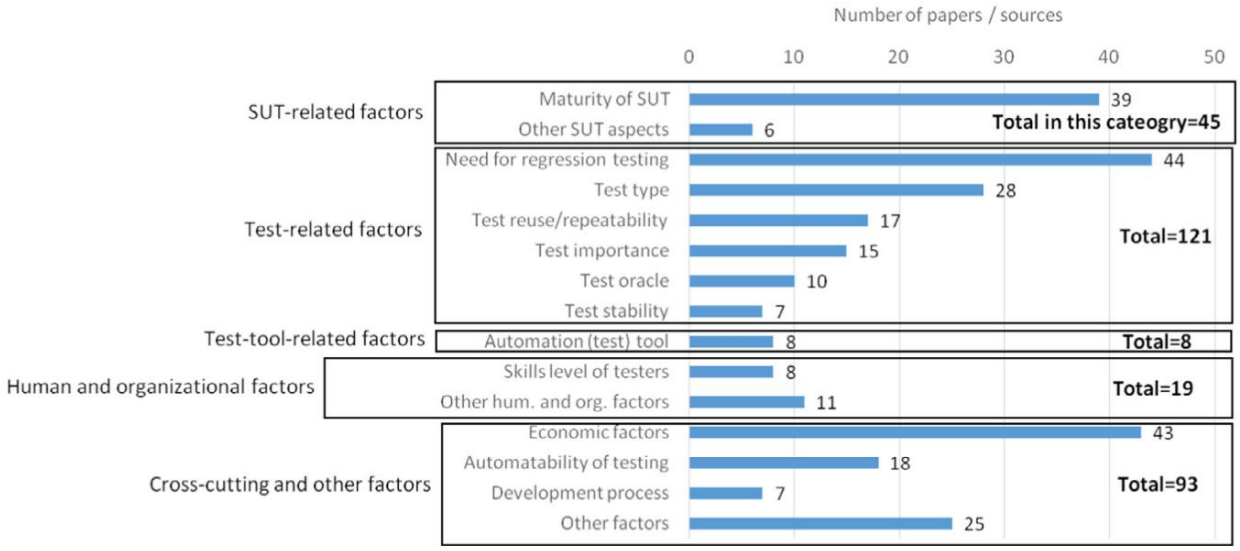


Fig. 16. Fatores considerados ao decidir quando automatizar os testes e quais partes do SUT devem ser automatizadas.

de caso para caso. Por exemplo, oito fontes que tivemos no pool mencionou que o nível de habilidades dos testadores deve ser cuidadosamente considerado ao decidir quando e o que automatizar. Observe que uma única fonte frequentemente menciona vários fatores e, portanto, alguns dos totais na figura, por exemplo, 121 para os fatores relacionados ao teste, são mais que 78 (número de fontes no pool).

Como nossa análise revelou, as duas perguntas "o que automatizar" e "quando automatizar" estão intimamente relacionados. O "o que automatizar?" foi interpretada em nossas fontes como quais casos de teste para automatizar, por exemplo, quais tipos de teste, quais casos de teste e quais recursos. O "quando automatizar?" pergunta foi interpretada em nosso fontes de duas maneiras: (1) quando iniciar a automação durante o ciclo de vida de um projeto de software e (2) sob quais condições, é preferível implementar um caso de teste de forma automatizada em vez de manual, por exemplo, quando se tem pessoal tecnicamente capacitado, quando a viabilidade de automação é alta, quando são encontrados testes adequados. A segunda interpretação de "quando" tem uma sobreposição muito alta com "o que automatizar?" Assim, nossa decisão de analisá-los globalmente foi baseada em nossos artigos de origem que revisamos nesta investigação, ou seja, queríamos ser o mais compatíveis com eles que possível. Em seguida, apresentamos nossas descobertas em relação a cada um dos as cinco categorias mostradas na Fig. 2.

6.2.3. Fatores relacionados ao SUT

As propriedades do sistema em teste (SUT) têm grandes impactos nas decisões de automação (Fig. 16). 45 fontes discutiram fatores nesta categoria. Se o SUT não estiver maduro o suficiente (discutido em 39 fontes), por exemplo, devido à reimplementação de novos recursos ou grandes mudanças em curso, haverá um grande impacto negativo na testes automatizados (também chamados de "testes quebrados" [84]) como o número de relatórios de defeitos falso-positivos de testes automatizados aumentariam e o esforço necessário para consertar (reparar) testes automatizados e a análise de defeitos falso-positivos reduziria os benefícios dos testes automatizados. Em geral, o esforço para manter os testes atualizados com últimas alterações no SUT, muitas vezes referido como reparo de teste [84] ou teste co-manutenção, é uma grande preocupação na área de testes automatizados. Algumas citações diretas das fontes nesta categoria de fatores são:

- *"A automação falha quando o aplicativo atual tem um design instável":* em um artigo escrito por engenheiros de teste em uma empresa com sede na Índia empresa chamada United Health Group [Fonte 10]
- *"Automatizar testes para aplicativos estáveis":* em um artigo sobre desenvolvimento uma infraestrutura projetada para o sucesso da automação de testes [Fonte 12]
- *"Seleção de casos de uso (corretos) antes de começar a automatizá-los pode evitar grandes quantidades de retrabalho em termos de manutenção de teste atividades no projeto.":* em um estudo de caso industrial [85] em que o primeiro autor estava envolvido [Fonte 1]
- *"O SUT deve ter atingido um certo nível de maturidade para VGT (teste de GUI visual) para ser aplicável":* em um artigo sobre práticas desafios do teste de GUI visual [Fonte 16]

Outros fatores do SUT, como duração da vida útil, alta capacidade de personalização, complexidade e dependência de aplicativos de terceiros, são fatores que afetariam a decisão de automatizar os testes.

- *"A automação de testes é uma solução eficaz quando a vida útil do aplicativo lançado é longa":* em um white paper da Infosys [Fonte 48]
- *"Aplicativo Principal tem muita interdependência com outros Aplicativos que por sua vez não podem ser automatizados.":* em uma apresentação por Engenheiros da IBM [Fonte 50]

Resumo: Para tomar a decisão adequada com relação ao SUT fatores, é preciso levar em conta a maturidade e estabilidade do SUT e prever se haveria muitas mudanças futuras futuro que pode exigir um grande esforço de co-manutenção de teste.

6.2.4. Fatores relacionados ao teste

As características dos casos de teste e suítes de teste (Fig. 16) também impactam nas decisões de automação, ou seja, quais testes (não) automatizar? Descobrimos que a necessidade de testes de regressão foi o fator mais mencionado nas decisões de automação de testes (mencionado em 44 fontes, mais da metade de todo o pool). Parece que a importância deste fator na tomada de decisão correta (quando e o que) continuará a aumentar à medida que o contínuo e rápido lançamentos de software tornam-se mais frequentes.

- *"Uma vez automatizados, os testes de regressão podem ser eficientes e eficazes. Assim, a ABB decidiu concentrar sua tentativa de estabelecer testes automatizados em testes de regressão de compilação, que são mais adequados para automação e onde os benefícios podem ser alcançados":* em um relato de experiência industrial de dois engenheiros da ABB Corporation [Fonte 8]

Machine Translated by Google

- Em uma postagem no blog intitulada *"Para aqueles que sonham com o sonho de automação 100%... por favor, acordem!" [Fonte* 37], um engenheiro da Microsoft cita outro praticante de testes chamado James Han cock da seguinte forma: *"James Hancock estimou que um teste deve ser executado 15-17 vezes para equilibrar o custo de desenvolvimento aquele teste".*
- Há também sugestões extremas como: "se você vai para executar um teste mais de uma vez, ele deve ser automatizado" [Fonte 39].

28 fontes consideraram os tipos de teste como fator para a tomada de decisão. Certos tipos de testes são bons candidatos para automação de testes enquanto outros não. Por exemplo, pesquisadores e praticantes relataram que os testes de desempenho e carga geralmente são muito difíceis devem ser executados manualmente e normalmente devem ser automatizados.

- *"Quando automatizar os testes? … Teste de carga e desempenho: simular centenas ou mesmo milhares de usuários virtuais em vários dispositivos":* em uma postagem no blog da Borland Corporation [Fonte 41]

Da mesma forma, testes que humanos não gostam de realizar foram sugeridos para automação. Por outro lado, os candidatos a testes manuais estão relacionados à experiência do usuário (UX) e testes de usabilidade, mas também testes que não são estáveis devido a problemas de tempo, por exemplo (por exemplo, em sistemas de tempo real). Esses testes oferecem pouco retorno para automação.

A reutilização e a repetibilidade do teste (mencionadas em 17 fontes) são outras fatores que estão intimamente relacionados ao teste de regressão. Eles podem se referir a casos em que o mesmo teste pode ser reutilizado como parte de outro teste, por exemplo, um teste de login deve passar em um aplicativo baseado na web antes outros testes podem ser executados. O número de ambientes para testar um SUT geralmente aumenta a repetibilidade do teste, por exemplo, ao testar aplicativos Android, é necessário repetir o mesmo teste em diferentes Modelos de telefones Android.

- *"Quando automatizar os testes? Testes em várias plataformas de SO e sites multilíngues "*: em uma postagem no blog da Borland Corporation [Fonte 41].

Como outro fator, a importância do teste foi mencionada em 15 fontes e decorre de dois fatores. Em primeiro lugar, se um determinado teste visa as funcionalidades importantes de um SUT que é altamente crítico para a satisfação do usuário, isso torna o teste mais importante e, portanto, incentivados a serem automatizados. A outra fonte para a importância do teste é a probabilidade do teste revelar defeitos.

- *"O número de bugs que o caso de teste deve (ou pode) encontrar é outro ponto a ser considerado [na decisão de automatizar]",* citado em artigo técnico do Centro Brasileiro de Estudos e Sistemas [Fonte 2]

A existência e estabilidade de oráculos de teste (discutido em 10 fontes), ou seja, o mecanismo de como os defeitos são detectados, também são importante. Resultados imprevisíveis que exigem julgamento humano como oráculos podem não valer a pena automatizar. Por outro lado, se o oráculo de teste é estável e previsível, então a automação de teste é recomendada, pois os humanos não são excelentes em vigilância.

- *"Resultados não determinísticos podem dificultar os testes automáticos independentemente da API ou UI":* em um documento técnico de um arquiteto de teste em Microsoft [Fonte 64]
- *"Automação tem tudo a ver com previsibilidade. Se você não puder expressar a entradas precisas e saídas esperadas, você não pode automatizar um teste"* [Fonte 38]

A estabilidade do teste é outro fator nesta categoria, pois se (a lógica de) um teste não é estável; automatizar não é uma boa ideia [Fonte 60]. Por último, mas não menos importante, outros candidatos à automação são os testes cujas entradas são previsíveis [Fonte 38].

Resumo: A introdução da automação de teste geralmente aumenta o custo para criar testes, porém o custo de reexecutar um teste diminui. Assim, entender o número de reexecuções de teste necessárias e o esforço de criação e manutenção de testes são importantes. Além disso, alguns tipos de teste, como testes de desempenho, são alvos melhores para automação de teste em comparação com outros, como experiência do usuário testes. Finalmente, as fontes dos oráculos de teste e sua estabilidade também são questões críticas a serem consideradas.

6.2.5. Fatores relacionados à ferramenta de teste

Qualidade e adaptação de automação de teste adequada e adequada ferramentas também são fatores importantes para quando e o que automatizar perguntas, que foram destacadas por 8 fontes (Fig. 16).

- *"A ferramenta de teste que está sendo utilizada para testes automatizados é capaz de interagir com todos os atributos necessários do recurso para teste ? propósitos? (Por exemplo, ele pode interagir com ele, assim como os usuários será capaz de? Podemos capturar todos os dados necessários do GUI e objetos filho?)":* como mencionado em um livro de dois consultores do setor [Fonte 22].

Em geral, parece que as ferramentas de teste de nível inferior e estrutura (por exemplo, em nível de unidade) são mais do mesmo tipo e mecanismo, uma vez que quase todas as ferramentas de teste são baseadas no família da estrutura xUnit. No entanto, no teste do sistema nível, as ferramentas são muito diversas e altamente dependentes do domínio da aplicação, por exemplo, para testes automatizados de aplicações, a ferramenta Selenium é bastante popular e enquanto para testar sistemas de telecomunicações, ferramentas baseadas no TTCN (Testing e Notação de Controle de Teste) são difundidos. Se uma ferramenta de automação de teste adequada para uma situação ou necessidade específica estiver faltando e não for possível desenvolver tal ferramenta internamente, então é aconselhável não para progredir com a automação de testes. Vários artigos e fontes reconheceram a necessidade de fazer uma seleção adequada de ferramentas e algumas fontes apresentar um processo como fazê-lo.

- *"Existem centenas de ferramentas de automação disponíveis no mercado. Um esforço cuidadoso deve ser feito para decidir qual ferramenta seria mais adequado para automatizar o teste do seu produto/ aplicação.":* em um artigo técnico da Infosys [Fonte 15]
- *"A ferramenta de automação deve ser escolhida com muito cuidado antes da processo de automação de teste começa. ... Funções complexas que podem não tem a confiabilidade necessária por causa de uma automação as dependências da ferramenta devem ser bem consideradas antes de serem criadas. Parece um conselho simples, mas é uma questão muito difícil de resolver: resultados errados relatados pela ferramenta.":* em um artigo técnico [Fonte 2]

Resumo: É preciso entender a ferramenta de teste e sua compatibilidade com o SUT e o modelo de negócios do fornecedor da ferramenta. A direção futura da ferramenta também é importante, pois ferramenta de teste para outra no meio de um projeto geralmente não é trivial. Ferramentas populares de código aberto costumam ser boas opções, pois têm um baixo custo (por exemplo, apenas treinamento, etc.), não correm o risco de um único fornecedor de ferramentas (como aumento de preços, parada repentina de desenvolvimento de ferramentas, etc.), e grandes bases de usuários que podem ser vistas como a garantia da sustentabilidade e sucesso futuro das ferramentas.

6.2.6. Fatores humanos e organizacionais

Fatores humanos e organizacionais também afetam a automação de testes decisão (Fig. 16). A automação de teste requer habilidades diferentes (e muitas vezes adicionais) do teste manual. Se a equipe de testadores não tiver habilidades de programação, a introdução de automação de teste para essa equipe exige treinamento ou corre um alto risco de falha (discutido em 8 fontes). Desta forma, quando faltam competências e não há recursos disponíveis para o treinamento necessário para adquirir as habilidades, então talvez seja melhor não automatizar. Os testadores de software também podem visualizar

Machine Translated by Google

testar a automação como uma ameaça (à segurança do trabalho) e resistir a ela [Fonte 68].

- *"…diferentes habilidades são necessárias para implementar um programa de teste automatizado eficaz daquelas exigidas para testes manuais":* em um livro de três engenheiros de teste trabalhando em uma automação de teste empresa [Fonte 21]
- *"Organizações de testadores/QA podem se sentir confortáveis e experientes com testes manuais e se sentem ameaçados pela automação":* [Fonte 68]

Além das habilidades dos testadores, vários outros fatores humanos e organizacionais também foram discutidos em 11 fontes, por exemplo, maturidade organizacional [Fonte 8], restrições de tempo e recursos [Fonte 50] e necessidade de gerenciamento de mudanças adequado [Fonte 73]. Uma citação é o seguinte:

- *"Uma organização madura o suficiente para lidar com melhorias de processo de forma estruturada é um pré-requisito para o estabelecimento de testes automatizados. Se não maduro o suficiente, é inevitável que a organização produzirá ainda mais caos ao realizar a introdução":* um relato de experiência industrial de dois engenheiros da ABB Corporation [Fonte 8]

A automação de testes normalmente requer um alto investimento inicial antes que os benefícios comecem a aparecer [86]. Assim, um cronograma apertado pode impedir a introdução da automação de teste.

- *"Se você estiver em um projeto acelerado onde o gerenciamento de projetos tem um cronograma de entrega muito apertado, você pode esquecer sobre automação, a menos que tenha sido alocado tempo adequado especificamente para isso":* em um livro de dois consultores do setor [Fonte 20]

Resumo: A introdução da automação de testes também é uma mudança, um problema de gestão que é propenso a falhas e tem um grande corpo de literatura de consultoria dando conselhos sobre isso. Para considerar com sucesso esse fator na tomada de decisão, é preciso entender a política organizacional, as competências atuais da empresa, como organizar treinamento e, talvez, o mais importante, ter soft habilidades para fazer com que as pessoas aceitem e até se empolguem com o teste automação.

6.2.7. Fatores transversais e outros

Também identificamos vários fatores que eram transversais, ou seja, eles são aplicáveis no contexto de mais de um grupo. Aqueles fatores foram: (1) fatores econômicos, (2) automatização de testes, e (3) processo de desenvolvimento. Além disso, um grupo de "outros" fatores que não pode ser enquadrado em nenhuma das categorias acima foi identificados, que serão discutidos a seguir.

Os fatores na categoria de fatores econômicos referem-se principalmente a as compensações de custo e esforço entre o manual e o automatizado teste. No total, conforme mostrado na Fig. 2, 43 fontes mencionaram fatores que se enquadram nesta categoria. Aqui, a ênfase está no esforço gastos e benefícios obtidos por meio de testes automatizados ou manuais, por exemplo, cálculos de retorno do investimento (ROI). Um exemplo simples de o gráfico de ROI de automação de teste é mostrado na Fig. 17. Embora um A perspectiva de ROI mostrada nesta figura pode parecer muito simplista para alguns leitores, muitas das ferramentas de apoio à decisão existentes neste (por exemplo, o da IBM [Fonte 77]) são baseados neste modelo simples de ROI (veja a planilha online http://goo.gl/zwY1sj por detalhes).

O senso comum sugere que a automação de teste é mais econômica em comparação com o teste manual se for necessário realizar vários testes iterações de testes [86]. Normalmente, o custo inicial da automação é mais do que testes manuais, mas custos de execução de testes automatizados são menores do que os testes manuais, especialmente se os testes forem repetidos muitas vezes. Curiosamente, alguns profissionais também apresentaram contra-argumentos aos cálculos de ROI.

- *"se você tiver dificuldade em convencer alguém de que precisa ser automatizado e se você sentir que precisa de uma equação para ver as vantagens de longo prazo da automação, não se preocupe em automatizá-la" [86]:* um artigo online de um consultor de teste de software [Fonte 52]
- *"Automatize todas as coisas que oferecem retornos imediatos":* um white paper por um consultor de teste de software [Fonte 57]

Automatização de testes, ou seja, quão fácil é testar o SUT em de forma automatizada, é outro fator deste grupo (mencionado em 18 fontes). A automatização não é apenas uma propriedade do SUT mas também é afetado pela(s) ferramenta(s) de automação de teste e nível de habilidade, por exemplo. É por isso que o consideramos um fator "transversal".

- *"Por exemplo, fazer com que a equipe de design da interface do usuário altere campos editáveis em campos suspensos não editáveis sempre que possível— como campos de data e hora — podem reduzir drasticamente o tamanho do conjunto de testes de validação de entrada do usuário em potencial e ajudar nos esforços de automação.":* no guia do International Software Testing Certificação de Gerente de Teste Avançado do Conselho de Qualificações (ISTQB) [Fonte 18]

A escolha do processo de desenvolvimento como outro fator de impacto a decisão foi abordada em sete fontes. Apenas dois artigos abordaram o processo Agile vs. Waterfall em particular:

- *"Você deve automatizar todos os projetos (ágeis)? … eu não acho faz sentido para um projeto que tem apenas 2-3 sprints":* um artigo do Diretor de Garantia de Qualidade da eSecurtel [Fonte 28]
- *"Os testadores XP têm que dirigir na pista rápida… você precisa de um projeto de teste automatizado e ferramentas de teste leves":* em um artigo de dois engenheiros de teste [Fonte 55]

O número de lançamentos, um aspecto importante do processo de desenvolvimento de software, foi um insumo considerado em três dos quatro ferramentas de apoio à decisão que oferecem cálculos de ROI que estavam em nossa piscina. Maior frequência de liberação aumentou os benefícios do teste automação. Orientação mais geral em relação ao desenvolvimento processo também foi dado:

- *"De uma perspectiva de processo, [precisamos] determinar como a automação de teste se encaixará no processo de desenvolvimento do sistema":* em um artigo on-line de um arquiteto de teste sênior [Fonte 54]

Percebemos que alguns dos fatores discutidos neste artigo têm relação implícita com a questão de saber se o desenvolvimento processo afeta as decisões de automação de software, por exemplo, reutilização de teste e a repetibilidade é geralmente aplicável em processos de desenvolvimento Agile em que iterações freqüentes de teste são conduzidas. No outro Por outro lado, no modelo Waterfall, o teste é uma fase única para o final do projeto, a necessidade de automação de teste geralmente é menor, porque no início não há código para o qual a automação de teste possa ser usada para verificar, a fase de teste geralmente está sob pressão de tempo e há menos ocorrência de integração e entrega contínuas e, portanto, menos justificativa/necessidade de automação de teste. No entanto, tais generalizações podem ser perigosas, pois a decisão em esse contexto é muito "contextualizado" e nos lembra a famosa citação da engenharia de software: *"Depende!".*

25 do total de 78 fontes discutiram outros fatores que poderiam não ser colocado em nenhuma das categorias acima e foram bastante esparsas para merecer categorias (clusters) próprias. Nós assim agrupou-os na categoria "Outros" na Fig. 16. Alguns desses fatores (recomendações) que devem ser considerados são:

- *"[Automatizar quando] O escopo da automação foi definido":* [Fonte 15]
- *"Complexidade do ambiente de teste":* [Fonte 27]

Machine Translated by Google

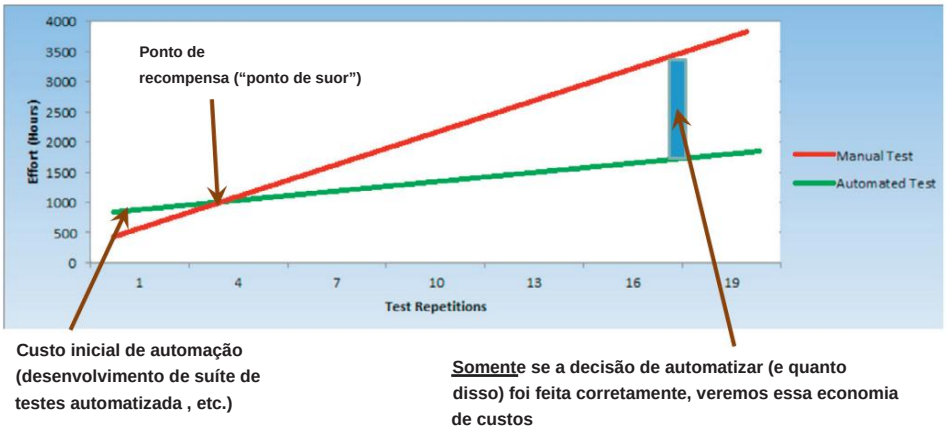


Fig. 17. Um gráfico de exemplo do ROI de automação de teste suportado por muitas ferramentas industriais.

- *“Os fatores-chave do programa mais comuns são: ... Objetivos de qualidade (velocidade de escape do defeito)” [Fonte* 54]
- *“[Necessidade de mais] cobertura, produtividade, precisão”:* [Fonte 68]

Resumo: Para usar com sucesso fatores transversais na decisão fazer, é preciso acompanhar os testes em relação ao custo (esforço) e benefícios que também permitiriam o cálculo do ROI da automação. Se o SUT ou casos de teste têm baixa automatização (como é fácil é testar de forma automatizada), o sucesso da automação estará perigo. O processo de desenvolvimento de software e o modelo de lançamento escolhidos no contexto de um projeto também são fatores importantes para considerar ao avaliar o uso da automação de teste.

6.3. RQ 3-ferramentas propostas para apoiar as perguntas quando/o que

Encontramos também quatro ferramentas online para apoiar o quando e quais perguntas na automação de teste de software. Todas as ferramentas onde na prática, calculadoras de ROI levando em consideração vários fatores que afetam os custos e benefícios de automação de teste de software semelhantes ao fatores apresentados na seção anterior. As ferramentas foram oferecidas por empresas com foco em TI geral, software e consultoria (IBM) [Fonte 75], soluções de automação de teste de software (Elbrus Ltd e Automated Testing Institute) [Fonte 76, 78] e desenvolvimento e teste de software (GlobalNow IT Services) [Fonte 77].

Em geral, as ferramentas ofereciam uma visão limitada dos fatores que afetam a automação de testes (Consulte a planilha http://goo.gl/zwY1sj para detalhes). Por exemplo, testes diferentes não foram considerados, mas em vez disso, todos os testes foram vistos como igualmente valiosos e igualmente automatizáveis. Também apenas uma ferramenta [fonte 75] considerou o treinamento de testadores. Três ferramentas não consideraram como fator a quantidade de regressão testando que era o fator mais popular, conforme discutido na seção anterior, veja a Fig. 16. No entanto, eles forneceram um número de fator de lançamentos, que é um dos fatores que explicam a necessidade de regressão teste. Surpreendentemente, descobrimos que nenhuma ferramenta listou o fator Maturidade do SUT como o 3º fator frequente na seção anterior. Desta forma, pensamos que a classificação da seção anterior e os detalhes da nossa planilha pode ser usado para construir um suporte de decisão com muito mais precisão e atualmente estamos no processo de fazê-lo.

6.4. Sistemas de software RQ 4 em teste ou projetos em estudo

6.4.1. RQ 4.1-Número de sistemas de software ou projetos em análise em cada fonte

Para cada fonte, estudamos quantos sistemas de software ou projetos em análise foram utilizados em cada fonte? Esperávamos que um determinado artigo ou artigo aplicasse a ideia proposta a pelo menos um sistema para mostrar sua eficácia. Descobrimos que apenas 28 de 78 (36%) fontes tinham pelo menos um sistema que eles havia investigado. Novamente, suspeitamos que o número é causado por um grande número de literaturas cinzentas, veja a Fig. 6. Descobrimos que em literatura cinzenta apenas 13 fontes de 52 (25%) enquanto formalmente trabalhos publicados, livros e artigos científicos, 15 das 26 fontes (62%) estudaram pelo menos um sistema. Além disso, descobrimos que, na maioria dos artigos, o número de sistemas ou projetos estudado foi um, veja a Fig. 18 (17/27) e no total havia apenas quatro artigos onde o número de sistemas estudados foram quatro ou mais alto.

6.4.2. RQ 4.2-domínios e tipos de sistemas de software ou projetos sob análise

Os domínios dos sistemas de software e projetos em análise variaram entre as fontes do pool, e não conseguimos encontrar um domínio particular que dominaria em nossa área. A vasta A variedade de domínios encontrados nas fontes foram, por exemplo, aplicativos da Web, aplicativos móveis, software de escritório, finanças, sistemas de controle (por exemplo, elevadores) e farmacêutico.

Com relação aos tipos de sistemas, ou seja, sistemas de brinquedo reais de código aberto, comerciais reais ou experimentais, descobrimos que sistemas comerciais reais sistemas foram estudados em 17 casos, enquanto sistemas experimentais ('brinquedos') foram utilizados em 11 fontes. Nenhuma fonte utilizou sistemas de código aberto reais, como kernel Linux e Mozilla Firefox. Surpreendentemente, seis das fontes que usam exemplos baseados em brinquedos vieram de literatura, enquanto cinco vieram da literatura formalmente publicada. Isso significa que a literatura formal contribuiu com 12 fontes do mundo real, enquanto a literatura cinzenta contribuiu com apenas cinco. Apesar do grande número de literatura cinzenta, eles forneceram surpreendentemente poucas evidências do mundo real, e mesmo essas evidências eram frequentemente relatados de forma menos rigorosa.

6.4.3. RQ 4.3-medidas para apoiar as perguntas quando/o que

Por fim, estávamos interessados em identificar o papel que realmente coletou evidências empíricas na forma de dados quantitativos medições em relação a perguntas sobre quando e o que automatizar. Obviamente, esta questão só é relevante com os 17 fontes que analisaram sistemas do mundo real. Descobrimos que lá foram 11 casos relatando benefícios quantitativos. O ROI foi relatado em quatro fontes e variou entre 40% e 3200% e o número de execuções de teste ou iteração necessária foi relatado em duas fontes que relataram 13 [Fonte 16] e 22 [Fonte 46] iterações de testes. No entanto, esses números não podem ser usados para qualquer tipo de generalizações como os fatores que afetam o quando e o que automatizar e a economia disso varia muito entre os casos, ilustrado por nossa taxonomia de fatores na Fig. 6.

Machine Translated by Google

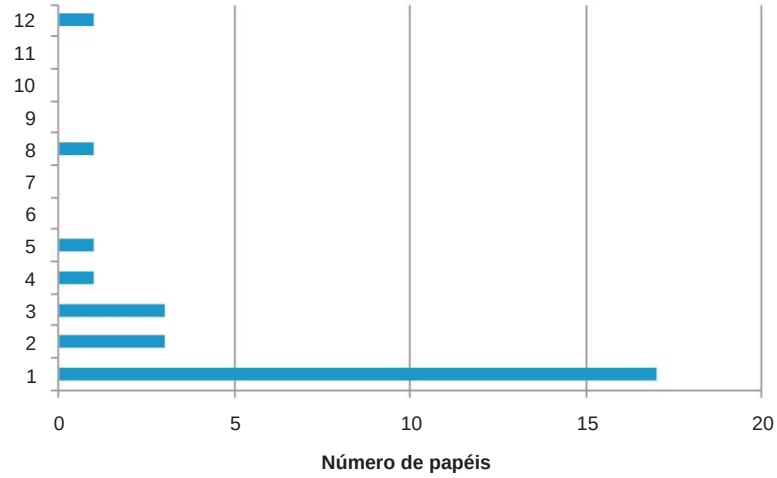


Fig. 18. Número de sistemas de software ou projetos em análise em cada fonte.

7. Discussões

A Seção 7.1 discute o resumo das descobertas e implicações do nosso MLR. A Seção 7.2 discute possíveis ameaças à validade de nosso estudo e as medidas que tomamos para minimizá-los ou mitigá-los.

7.1. Resumo das Constatações

Os profissionais de teste de software precisam de automação de teste apoio à decisão ou critérios. Achamos que isso é visível a partir do número de resultados do Google produzidos por nossas strings de pesquisa, consulte a Seção 4.1, e do fato de que dois terços da literatura desta revisão papel são literatura cinzenta publicada por profissionais, veja a Fig. 6. Além disso, de acordo com o melhor conhecimento dos autores, nenhuma sistemática revisões de literatura têm sido realizadas sobre as questões de quando e o que automatizar no teste de software. Esses motoristas motivaram para realizar um estudo de MLR nesta área. Nosso estudo MLR, respondeu quatro questões de revisão (RQ1 a RQ 4). Abaixo resumimos os resultados dos RQs e discutimos os resultados para a comunidade de pesquisa e profissionais.

7.1.1. RQ 1-mapeamento por facetas de contribuição e pesquisa

O RQ1 mapeou as fontes por contribuição e facetas de pesquisa. Nós constataram que a maioria dos artigos (58) de suporte de decisão mais fraco, pois fornecem apenas heurísticas ou diretrizes. Mais um suporte rigoroso à decisão é oferecido por 11 artigos cuja contribuição classificamos como método ou técnica. Sete artigos propõem uma ferramenta para decidir quando ou o que automatizar. Observe que alguns artigos que ofereciam técnicas também propunham uma ferramenta de apoio à sua técnicas. Quatro artigos apresentam um processo sobre como decidir sobre automação de testes e seis artigos forneceram modelos sobre o tema. Como um Em síntese, podemos afirmar que parece haver mais espaço para métodos, ferramentas, processos e modelos nesta área. Já existem abundância de heurísticas que representam a forma mais fraca de decisão Apoio, suporte.

Identificamos uma tendência semelhante ao mapear o sourcing em relação à contribuição da pesquisa. A maioria dos trabalhos foi baseada na opinião (34), que é a forma de evidência mais fraca, a experiência (34), que é a segunda mais fraca. Encontramos apenas seis estudos de validação e um estudo de pesquisa de avaliação que apresentam evidência empírica mais rigorosa do suporte à decisão. A comparação com outros SLRs mostrou que isso não é típico, veja a Fig. 13 . acho também que isso implica a importância prática do tema, mas a falta geral de interesse acadêmico em apresentar suporte sistemático à tomada de decisão para automação de teste de software.

7.1.2. RQ 2-fatores considerados para decidir quando/o que automatizar

Sintetizamos todos os fatores que afetam a decisão de automação de teste usando diretrizes de análise de dados qualitativos [83], conforme descrito na Seção 5.2. Identificamos 15 fatores que, além disso, classificamos em cinco grupos, ou seja, fatores relacionados ao SUT, fatores relacionados ao teste, fatores relacionados à ferramenta de teste, fatores humanos e organizacionais, e Transversais e outros fatores. Descobrimos que a necessidade de teste de regressão foi o fator mais mencionado com 44 fontes seguidas de perto por Fatores econômicos (43) e Maturidade do SUT (39).

As frequências listadas são, no entanto, apenas ilustrativas e devem não deve ser utilizado como indicação da importância de cada fator. Nós acho que cada caso particular em que a automação de teste ocorre deve considerar e classificar os fatores com base na importância nesse determinado contexto. Por exemplo, o nível de habilidade dos testadores foi mencionado em apenas oito fontes. Ainda assim, para algum nível de habilidade de contexto de testadores podem ser o obstáculo número um enquanto em alguns outros, por exemplo, aqueles que já fizeram muita automação de teste, precisariam prestar pouca atenção a esse fator. Assim, a classificação deve servir como uma lista de verificação para os profissionais que tomam decisões de automação de testes, em vez de uma lista normativa dando prioridades absolutas. Para acadêmicos, pode oferecer ideias para estudos futuros.

7.1.3. RQ 3-ferramentas propostas para apoiar as perguntas quando/o que

Encontramos quatro ferramentas online diferentes que estavam disponíveis abertamente ou preenchendo o cadastro gratuito. Estes fornecidos por empresas oferecendo automação de testes e outros tipos de serviços de TI, como IBM. Descobrimos que as ferramentas consideraram uma lista inadequada de tópicos para suporte à decisão de automação de teste quando comparado com nossa lista de fatores. Isso destaca a necessidade de construir um melhor suporte à decisão que está abertamente disponível para os praticantes.

7.1.4. Sistemas de software RQ 4 em teste ou projetos em estudo

Descobrimos que apenas 64% dos artigos não estudaram nenhum sistema ou projeto de software para justificar seus conselhos sobre quando automatizar e o que automatizar no teste de software. Foi ainda mais surpreendente que a completa falta de sistemas analisados foi mais frequente (75%) na literatura cinzenta decorrente principalmente de fontes industriais em comparação com a literatura formalmente publicada (38%) publicados por acadêmicos e profissionais da indústria. Isso sugere que os requisitos impostos à publicação formal realmente aumentam a quantidade de evidências empíricas em engenharia de software.
Por outro lado, muitas fontes de literatura cinzenta eram opinião ou baseado na experiência, mas não forneceu exemplos. Assim, as fontes conhecimento, ou seja, a fonte epistemológica, não foram formalmente identificado. Por exemplo, quando um autor estava afirmando “eu acho que X”

Machine Translated by Google

o autor poderia ter tido experiência ou mesmo dados de vários
projetos de automação de teste ou o X poderia ser baseado no que ele tinha
Acabei de ouvir recentemente de um colega como uma opinião grosseira.

Análises posteriores mostraram que o sistema de software ou projeto
em estudo vieram de vários domínios diferentes, por exemplo, web, celular, incorporado.
Com relação aos tipos de sistema, descobrimos que
17 artigos analisaram sistemas comerciais de código fechado enquanto
11 artigos analisaram pequenos brinquedos ou sistemas experimentais. Nós
descobriram que nenhum estudo havia analisado sistemas de código aberto do mundo
real. Isso foi bastante surpreendente, dada a popularidade de analisar
os sistemas de código aberto em comunidades de pesquisa com foco, por exemplo, em
Repositórios de Software de Mineração (MSR) ou Livre e Aberto
Fonte (FLOSS).

Por fim, estudamos a quantidade de sourcing fornecendo dados quantitativos
evidência de medição sobre os benefícios e desvantagens do software
automação de testes do sistema comercial real, assim, excluímos
os sistemas de brinquedos para esta parte. Descobrimos que apenas 11 fontes (14%)
forneceram evidências quantitativas sobre o tema. Assim, a falta de alta
existem estudos de qualidade do mundo real sobre este tópico.

7.2. Implicações para a prática e uma lista de verificação para apoiar tomando uma decisão

Em resumo, nosso estudo teve como objetivo beneficiar os praticantes por
apresentando uma única fonte que sintetiza e resume todas
as abordagens, diretrizes e opiniões baseadas na experiência wrt
os dois W's de teste de software automatizado (o que e quando), um
necessidade que muitos praticantes nos expressaram pessoalmente em
nossas interações indústria-academia, por exemplo, [13-16]. Nosso estudo também
uma contribuição para a comunidade de pesquisa, capturando o estado da arte e a
prática em suporte à decisão de automação de teste e
também apontando lacunas e necessidades de pesquisa nesta área ativa. Para
tornar nossos resultados mais úteis para os profissionais, desenvolvemos
e apresentar na Tabela 3 um rascunho da lista de verificação que os profissionais podem usar
ao avaliar o que e quando automatizar perguntas. A importância de cada item da lista
de verificação varia caso a caso. Portanto, é importante que os usuários avaliem a
importância de cada
reivindicar em seu próprio contexto. Muitas vezes, esse conselho é dado por
consultores e eles também aparecem em livros de texto dos profissionais. Nós
deve observar ao leitor que a lista de verificação representa nosso especialista
com base nos resultados de MLR apresentados neste artigo. Não é um
instrumento validado cientificamente, pois a utilidade da lista de verificação ainda não
foi validada empiricamente nem temos dados de linhas de base para mostrar as médias
da indústria. Ambos os passos são bons
temas para estudos futuros.

7.3. Identificando e abordando possíveis ameaças à validade

Baseado em diretrizes para realizar revisão sistemática da literatura
e estudos de mapeamento [63,64,66] e também com base em nossa experiência
anterior [6,43,70], identificamos sistematicamente e tratamos cuidadosamente as
ameaças potenciais à validade de nosso estudo, tomando medidas para
minimizá-los ou mitigá-los. Discutimos a seguir a validade potencial
ameaças no contexto dos quatro tipos de ameaças de validade com base em
uma lista de verificação padrão adotada de [87].

7.3.1. Validade interna

A abordagem sistemática que tem sido utilizada para a seleção da fonte é descrita
acima. Para garantir que esta revisão
é repetível, mecanismos de pesquisa, termos de pesquisa e inclusão/exclusão
critérios são cuidadosamente definidos e relatados. Problemas problemáticos em
processo de seleção são limitações de termos de pesquisa e motores de busca,
e viés na aplicação dos critérios de exclusão/inclusão.

A limitação de termos de pesquisa e mecanismos de pesquisa pode levar a um
conjunto completo de fontes primárias. Para mitigar o risco de encontrar
todos os estudos relevantes, a busca formal usando palavras-chave definidas foi
conduzido. Por isso, acreditamos que uma base adequada e inclusiva
foi coletado para este estudo e se houver fontes ausentes,
a taxa será insignificante.

A aplicação de critérios de inclusão/exclusão pode sofrer com o julgamento e a
experiência dos pesquisadores. O preconceito pessoal pode ser introduzido durante
este processo. Conforme apontado na Seção 4.1, ambos os pesquisadores buscaram
independentemente a literatura que já
consistiu na exclusão dos artigos que atenderam aos critérios de inclusão.
Isso poderia ter potencialmente excluído alguns que deveriam ter sido
incluído. No entanto, esse problema é parcialmente mitigado, pois ambos os autores
podem ser considerados especialistas em pesquisa de engenharia de software
e teste de software, por exemplo, ambos têm cargos permanentes de professor
e ambos tinham doutorado há mais de 9 e 5 anos, respectivamente, antes da realização
deste estudo. Se a busca tivesse sido feita
por novos estudantes de doutorado ou mesmo estudantes de mestrado, então o trabalho em pares seria
foram recomendados em todas as fases para substituir a falta de
perícia no assunto. Por exemplo, estudos sobre programação em pares [88] mostraram
que programadores juniores recebem mais
benefício do trabalho em dupla. Achamos que uma lógica semelhante funciona para a pesquisa como
Nós vamos. Além disso, para minimizar esse tipo de viés, a votação conjunta foi
aplicado após a inclusão inicial da fonte e somente as fontes que passam
a votação conjunta foram selecionados para este estudo.

7.3.2. Validade do construto

A validade de construto está preocupada com questões que
o objeto de estudo representa verdadeiramente a teoria por trás do estudo [87].
As ameaças relacionadas a esse tipo de validade neste estudo foram adequação
de RQs e esquema de categorização usado para a extração de dados.

As perguntas de revisão são projetadas para cobrir nosso objetivo. As perguntas são
respondidas de acordo com um esquema de categorização. Para projetar um
bom esquema de categorização, adaptamos classificações padrão de [65] e também
finalizamos o esquema usado por meio de
várias iterações.

Uma ameaça à validade de construto vem da falta de evidência empírica nos
estudos primários. A maior parte da literatura cinzenta era baseada em opiniões ou
experiências. Desde a literatura cinzenta
representa a voz dos praticantes, poderíamos supor que
eles estariam falando por experiência pessoal com sistemas industriais reais, embora
esses sistemas não tenham sido explicitamente mencionados/discutidos nas fontes.
Isso tornaria o empírico
bases mais sólidas. Por outro lado, pode ser que alguns profissionais possam
simplesmente estar repetindo as ideias/opiniões que tiveram
ouvido de outros praticantes. Assim, como fonte de conhecimento
não era tipicamente revelado na literatura cinzenta, estamos diante de uma
problema epistemológico. Não sabemos, como sabemos, o que
conhecer. Esta é uma séria limitação nos estudos primários, mas a fixação
obviamente não é possível.

No entanto, o facto de nos basearmos principalmente em opiniões e
evidência não significa que nosso resultado esteja incorreto. Se os mesmos indivíduos
que deram sua opinião ou documentos de experiência fossem
entrevistado ou pesquisado com um questionário do mesmo tópico,
seria altamente provável que o resultado fosse o mesmo. O único
coisa que seria diferente seria que o método de pesquisa
que costumava recolher a voz dos praticantes seria mais
rigoroso.

7.3.3. Validade da conclusão

A validade de conclusão de um estudo de revisão se preocupa em chegar a
conclusões apropriadas por meio de tratamento rigoroso e repetível. Para garantir a
confiabilidade de nossos tratamentos, toda a
conjunto de fontes primárias são analisados e os dados foram revisados,
extraído e sintetizado pelos dois autores.

Seguir a abordagem sistemática e o procedimento descrito garantiu a replicabilidade
deste estudo e assegurou que os resultados de estudo semelhante não terão grandes
desvios de nossa classificação
decisões.

Machine Translated by Google

Tabela 3

Uma lista de verificação para apoiar a tomada de decisão sobre automatizar o teste de software. O sinal "+" significa que a situação dada favorece a automação de testes enquanto o sinal "-" sugere não automatizar o teste. O peso da área é o número de fontes para cada fator em nosso estudo.

Categoria	Área (peso, ou seja, número de fontes)	Situação	+/ÿ
Fatores relacionados ao SUT	Maturidade do SUT (39)	O SUT ou os componentes visados sofrerão grandes modificações no futuro.	
		A interface através da qual os testes são realizados é improvável que mude.	+
	Outros aspectos do SUT (6)	SUT é um aplicativo com um longo ciclo de vida.	+
		SUT é um sistema genérico, ou seja, não feito sob medida ou fortemente sistema personalizado.	+
		O SUT está fortemente integrado a outros produtos, ou seja, não independente.	
		SUT é complexo.	
		O SUT é de missão crítica.	+
Fatores relacionados ao teste	Necessidade de teste de regressão (44)	Testes de regressão frequentes são benéficos ou essenciais.	+
	Tipo de teste (28)	Testes são testes de desempenho e carga.	+
		Os testes são testes de verificação de fumaça e construção.	+
		Testes são testes de unidade.	+
		Há um grande número de testes que são semelhantes a cada outro.	+
		Os testes requerem grandes quantidades de dados.	+
		Os seres humanos tendem a cometer erros ao realizar e avaliar esses testes, por exemplo, testes exigem vigilância em execução.	+
		Os computadores provavelmente cometerão erros ao executar e avaliar esses testes, por exemplo, a execução do teste não é determinista.	
	Testar reutilização/repetibilidade (17)	Os testes podem ser reutilizados como parte de outros testes.	+
		Testes precisam ser executados em vários hardwares e softwares ambientes e configurações.	+
		A vida útil dos testes é alta.	+
		O número de builds é alto.	+
	Importância do teste (15)	Os testes podem revelar defeitos, ou seja, áreas de alto risco.	+
		Os testes cobrem as características mais importantes, ou seja, alta áreas de importância.	+
	Testar oráculo (10)	Os resultados dos testes são determinísticos.	+
		Os resultados dos testes requerem julgamento humano.	
		A comparação automatizada será frágil, levando a muitos falso-positivo.	
	Teste de estabilidade (7)	Os testes são instáveis, por exemplo, devido ao tempo. Devemos realizar o testar repetidamente e se passar acima de um limite, considerar que o teste passa.	+
		Os testes são instáveis, por exemplo, devido ao tempo. Os resultados não podem ser confiado em tudo.	
Fatores relacionados à ferramenta de teste	Ferramenta de automação (teste) (8)	Experimentamos a ferramenta de automação de teste que planeja usar e os resultados são positivos.	+
		Está disponível uma ferramenta de teste adequada que se adapta ao nosso propósito.	+
		Decidimos qual ferramenta usar.	+
		Podemos arcar com os custos da ferramenta.	+
Fatores humanos e organizacionais	Nível de habilidades dos testadores (8)	Nossos engenheiros de teste têm habilidades adequadas para automação de teste.	+
		Podemos dar ao luxo de treinar nossos engenheiros de teste para teste automação.	+
		Temos experiência na abordagem de automação de testes e ferramenta que escolhemos.	+
	Outros zumbidos. e org. fatores (11)	No momento, estamos com um cronograma e ou orçamento apertados pressão.	
		Temos apoio organizacional e de gestão de topo para automação de testes.	+
		Há uma grande resistência à mudança contra o teste de software automação.	
		Temos a capacidade de influenciar ou controlar as mudanças SUT.	+
Fatores transversais e outros	Fatores econômicos (43)	Existem benefícios econômicos na automação de testes.	+
	Automatização de testes (18)	Os testes são fáceis e simples de automatizar.	+
		Os resultados dos testes são fáceis de analisar automaticamente.	+
		A automação de teste exigirá muito esforço de manutenção.	
	Processo de desenvolvimento (7)	Nosso processo de desenvolvimento de software requer testes automação para funcionar de forma eficiente, por exemplo ágil métodos.	+
		Fazemos vários lançamentos de nossos produtos.	+

Machine Translated by Google

7.3.4. Validade externa

A validade externa diz respeito a até que ponto os resultados de nosso estudo pode ser generalizado. Conforme descrito acima, a pesquisa definida termos na abordagem de seleção de fonte resultou em ter fontes todas escritas em língua inglesa; estudos escritos em outros línguas foram excluídas. A questão está em saber se o nosso selecionado obras podem representar todos os tipos de literatura na área de estudo (quando e o que automatizar). Para essas questões, argumentamos que a literatura relevante que selecionamos em nosso pool continha informações suficientes para representar o conhecimento relatado por outros pesquisadores e profissionais.

O Capítulo 8 de [87] descreve a validade externa como a capacidade de generalizar resultados para contextos industriais. Como pode ser visto a partir do dados coletados por meio de estudo, além de estudos acadêmicos, boa proporção de trabalhos industriais e colaborativos existe em nosso fontes. Isso significa que nosso processo inclusivo de seleção de artigos nos levou a ter uma base adequada para a conclusão de resultados úteis para a academia e aplicáveis na indústria. Observe também que nosso achados neste estudo estão principalmente dentro da área específica sob estudar (quando e o que automatizar os testes). Além deste campo, nós não temos intenção de generalizar nossos resultados. Portanto, poucos problemas com validade externa merecem atenção substancial.

8. Conclusões e trabalhos futuros

O teste automatizado de software e o desenvolvimento de código de teste são agora mainstream na indústria de software e tópicos de engenharia desafiadores por conta própria. Jeff Feldstein, que era um gerente de teste na Cisco Systems, mencionou que *"Nós projetamos um sistema de teste isso provavelmente é tão complicado quanto o próprio sistema".* Ainda, decisão sobre quando e o que automatizar no teste de software não foi investigado com uma revisão de literatura antes. Dada a importância de automação de testes e os grandes investimentos monetários que podem ser desperdiçado com decisões incorretas, achamos que esse estudo era necessário.

Este estudo traz quatro contribuições. Primeiro, descobrimos que o A maioria dos trabalhos anteriores são artigos de opinião ou experiência que fornecem heurísticas ou diretrizes, enquanto apenas uma pequena parte dos artigos apresentar um trabalho mais rigoroso de avaliação ou pesquisa de validação apresentar processos ou modelos. Isso entra em conflito com os estudos prévios de mapeamento de ping da engenharia de software. Além disso, descobrimos que apenas 22% dos artigos apoiam suas propostas com base empírica. estudo de um sistema ou projeto de software do mundo real.

Em segundo lugar, decidimos incluir a literatura cinzenta em nosso estudo como constatamos que o tema era de interesse prático e como os estudos acadêmicos sobre o tema eram raros. Até onde sabemos, este artigo é um dos primeiros MLRs (e SLRs) em engenharia de software que estudou a literatura cinzenta, além das formalmente publicadas literatura. De acordo com nossa experiência neste MLR, descobrimos que incluir literatura cinzenta em estudos de SLR é perspicaz e, portanto, o autores recomendam incluí-lo quando o tópico tem um número baixo de estudos acadêmicos, mas de alto interesse profissional. A literatura cinzenta pode ajudar a incluir as experiências e opiniões dos profissionais como parte de SLRs.

Terceiro, analisamos qualitativamente os fatores que afetam o software decisões de automação de teste. Encontramos 15 fatores que formaram cinco grupos: fatores relacionados ao SUT, fatores relacionados ao teste, fatores relacionados à ferramenta de teste Fatores Humanos e Organizacionais e Transversais e outros fatores. Descobrimos que a Necessidade de testes de regressão (44 fontes), Fatores econômicos (43) e Maturidade do SUT (39) são os fatores mais citados. Por fim, notamos que o fator frequências não são indicativos de importância dos fatores, pois as variáveis contextuais de cada caso devem determinar a importância de cada fator.

Quarto, usamos nossa lista de fatores da Fig. 16 para criar um lista de verificação mostrada na Tabela 3 que pode ser usada ao tomar decisões de automação de software. Até onde sabemos, esta lista de verificação é a mais abrangente e mais científica taxonomia e lista de verificação até agora.

Estamos planejando seguir as seguintes direções de trabalho futuro:

- Classificando as abordagens de suporte à decisão pela granularidade das decisões, por exemplo, no nível do caso de teste (quais casos de teste deve ser automatizado?), no nível do projeto (vamos usar teste automação?), ou nível de negócios.
- Realização de estudos de entrevistas e pesquisas para entender o estado da prática de suporte a decisões de automação de teste de software. Particularmente, estamos interessados em como os conceitos modernos de desenvolvimento, como desenvolvimento de software ágil, DevOps e A implantação contínua afeta a tomada de decisão. Mesmo se um deseja ir para a entrega contínua e totalmente automatizada testado há necessidade de ordem de priorização na automatização do antigo testes.
- Validando nossa lista de verificação inicial de automação de teste mostrada em A Tabela 3 fornece outro caminho para pesquisas futuras.

Reconhecimentos

Vahid Garousi foi parcialmente apoiado por vários bolsas concedidas pela Universidade Hacettepe e o Científico e Conselho de Pesquisa Tecnológica da Turquia (TÜBÿ ITAK) via concessão #115E805. Os autores gostariam de agradecer ao professor Burak Turhan e Dr. Pilar Rodriguez da Universidade de Oulu e os revisores anônimos por seus comentários sobre as versões anteriores deste manuscrito.

Conjunto de fontes

Fontes técnicas e científicas

[Fonte 1] Y. Amannejad, V. Garousi, R. Irving e Z. Sahaf, "A Abordagem baseada em pesquisa para automação de teste de software econômica Apoio à Decisão e um Estudo de Caso Industrial", no Proc. do International Workshop on Regression Testing, co-localizado com o Sexto IEEE International Conference on Software Testing, Verification, and Validação, 2014, pp. 302–311.

[Fonte 2] C. Gouveia, J. Oliveira e R. Quidute, "A way of Improving Test Automation Cost-Effectiveness", em Conferência do Associação para Teste de Software, 2006.

[Fonte 3] SA Jolly, V. Garousi e MM Eskandar, "Teste de unidade automatizado de um software de controle SCADA: um Estudo de caso baseado em Pesquisa de Ação", na Conferência Internacional IEEE sobre Teste, Verificação e Validação de Software (ICST), 2012, págs. 400–409.

[Fonte 4] SK Muthusundar, "Estudo comparativo de test au tomaton ROI", Indian Journal of Computer Science and Engineering, volume 2, 2011.

[Fonte 5] D. Hoffman, "Análise de Custo-Benefícios da Automação de Testes", na conferência de Análise e Revisão de Testes de Software no Oeste (STARWEST), 1999.

[Fonte 6] A. Laapas, "Análise de custo-benefício do uso de automação de teste no desenvolvimento de software embarcado", Escola de Engenharia e Gerenciamento Industrial, Lappeenranta University of Tecnologia, Finlândia, 2014.

[Fonte 7] R. Ramler e K. Wolfmaier, "Perspectivas econômicas em automação de teste: equilibrando testes automatizados e manuais com custo de oportunidade", em Proceedings of the 2006 international work shop on Automation of software test, 2006, pp. 85–91.

[Fonte 8] C. Persson e N. Yilmazturk, "Estabelecimento de Testes de Regressão Automatizados na ABB: Relatório de Experiência Industrial sobre 'Evitando as Armadilhas'", apresentado nos Anais do IEEE International Conference on Automated Software Engineering, 2004.

Machine Translated by Google

[Fonte 9] S. Berner, R. Weber e RK Keller, “Observações e lições aprendidas com testes automatizados”, em Proceedings of Conferência Internacional sobre Engenharia de Software, 2005, pp. 571–579.

[Fonte 10] U. Pranitha e BV Sastry, “Abordagem pragmática para automação de teste de software”, em anual International Software Testing Conferência (STC), 2013.

[Fonte 11] J. Kasurinen, O. Taipale e K. Smolander, “Automação de teste de software na prática: observações empíricas”, Adv. Suave. Eng., v. 2010, pp. 1-13, 2010.

[Fonte 12] B. Boehmer e B. Patterson, “Teste de software automação–Desenvolver uma infraestrutura projetada para o sucesso”, em Conferência STAREAST (Software Testing Analysis & Review East), 2001.

[Fonte 13] RW Rice, C. CSQA e L. Rice Consulting Solutions, “Sobrevivendo aos 10 principais desafios da automação de teste de software”, CrossTalk: The Journal of Defense Software Engineering, pp. 26-29, 2003.

[Fonte 14] S. Münch, Peter Brandstetter, K. Clevermann, O. Kieckhoefel e ER Schäfer, “O retorno sobre o investimento (ROI) de Automação de Testes”, Engenharia Farmacêutica, vol. 32, pp. 1-8, 2012.

[Fonte 15] V. Motwani, “The When & How of Test Automation,” na QAI India Ltd, 3ª Conferência Anual Internacional de Testes de Software, 2001.

[Fonte 16] E. Alégroth, R. Feldt e L. Ryrholm, “Visual GUI testando na prática: desafios, problemas e limitações”, Empirical Engenharia de Software, pp. 1–51, 2014/01/15 2014.

[Fonte 17] Z. Sahaf, V. Garousi, D. Pfahl, R. Irving e Y. Amannejad, “Quando automatizar o teste de software? Suporte à Decisão baseado em Dinâmica de Sistemas – Um Estudo de Caso Industrial”, em Proc. da Conferência Internacional sobre Processos de Software e Sistemas, 2014, págs. 149-158.

Livros e capítulos de livros

[Fonte 18] R. Black, Teste de Software Avançado - Vol. 2: Guia para a Certificação Avançada do ISTQB como Gerente de Teste Avançado, 2ª edição: Rocky Nook, 2014.

[Fonte 19] E. Dustin, J. Rashka e J. Paul, software automatizado testes: introdução, gerenciamento e desempenho: Addison–Wesley Profissional, 1999.

[Fonte 20] JA Whittaker, J. Arbon e J. Carollo, How Google Software de testes: Addison–Wesley Professional, 2012.

[Fonte 21] E. Dustin, T. Garrett e B. Gauf, Implementando testes de software automatizados: Como economizar tempo e reduzir custos enquanto aumentando a qualidade: Addison–Wesley Professional, 2009.

[Fonte 22] DJ Mosley e BA Posey, software suficiente Automação de Testes: Prentice Hall Professional, 2002.

[Fonte 23] A. Clausen, “Projeto 1: Falha!, Projeto 2: Sucesso!” em Experiências de Automação de Testes: Estudos de Caso de Teste de Software Automação, D. Graham e M. Fewster, Eds., ed: Addison–Wesley Profissional, 2012.

[Fonte 24] AF Benet, CE Lujua, HS Grau, MM Jáimez, FM Pérez e C. Bianco, “Software para Dispositivos Médicos e Nosso Need for Good Software Test Automation”, in Experiences of Test Automação: Estudos de Caso de Automação de Teste de Software, D. Graham e M. Fewster, Eds., ed: Addison–Wesley Professional, 2012.

[Fonte 25] M. Fewster e D. Graham, Software Test Automation: Effective Use of Test Execution Tools: Addison–Wesley, 1999.

[Fonte 26] S. Desikan e G. Ramesh, Teste de Software: Princípios e Práticas: Pearson Education India, 2006.

Artigos da Internet e white papers

[Fonte 27] M. Bartley, “Alcançar benefícios de negócios por meio de teste de software automatizado” , http://www.bcs.org/category/18128, 2012, Último acesso: maio de 2015.

[Fonte 28] S. Thompson, “Testadores ágeis, você deve automatizar?” http://professionalservices.matrixresources.com/blog/Agile-testers-should-you-automate, 2013, Último acesso: janeiro de 2016.

[Fonte 29] Exforsys Inc., Test ing Advantages, Disadvantages e “Automatizado Guide lines,” http://www.exforsys.com/tutorials/testing/automatizado-testing-advantages-disadvantages-and-guidelines.html, 2005, Último acesso: janeiro de 2016.

[Fonte 30] M. Clermont, “Automatizando testes vs. automação de teste,” http://googletesting.blogspot.com.tr/2007/10/automating-tests-vs-test-automation.html, 2007, Último acesso: Janeiro de 2016.

[Fonte 31] B. Galen, “Critérios de Seleção de Automação – Escolhendo os Candidatos “Certos”, http://www.compaid.com/caiinternet/ezine/Galen3.pdf, 2007, Último acesso: janeiro de 2016.

[Fonte 32] S. Ford, “Automation Testing versus Manual Testing Guidelines,” http://blogs.msdn.com/b/saraford/archive/2005/02/07/368833.aspx, 2005, Último acesso: janeiro de 2016.

[Fonte 33] Cognizant, “Evite a automação de teste descartável” https://www.sqe.com/ControlImages/sqe/Image/Webinars/sqe_cognizant_finalslides.ppt, 2008, Último acesso: janeiro de 2016.

[Fonte 34] BL Suer, “Escolhendo o que automatizar”, http://sqgne.org/presentations/2009-10/LeSuer-Jun-2010.pdf, 2010, Último acessado em: janeiro de 2016.

[Fonte 35] Galmont Consulting, “Determinando o que fazer Automatizar,” http://galmont.com/wp-content/uploads/2013/11/Determinando-o-que-automatizar-2013_11.13.pdf, 2013, Último acesso: janeiro de 2016.

[Fonte 36] L. Hayes, “A automação de testes salva Tempo e dinheiro?”, http://www.stickyminds.com/article/faz-teste-automação-economize-tempo-e-dinheiro, 2001, Último acesso: Janeiro de 2016.

[Fonte 37] I. Testy, “Para aqueles de vocês que sonham 100% sonho de automação ... por favor, acorde!,” http://blogs.msdn.com/b/imtesty/archive/2006/08/02/686010.aspx, 2006, Último acesso: jan. 2016.

[Fonte 38] L. Hayes, “Automação de teste funcional”, em Teste SAP R/3: Guia passo a passo do gerente, J. Fajardo e E. Dustin, Eds., ed: Wiley, 2007, Último acesso: janeiro de 2016.

[Fonte 39] B. McLeod, “Se você vai fazer um teste mais de uma vez , deve ser automatizado.” http://www.teknologika.com/blog/tenet-if-you-are-going-to-run-a-test-mais-de-uma-vez-it-should be-automated/, 2005, Último acesso: janeiro de 2016.

[Fonte 40] M. Larsen, “Aprenda quando automatizar e quando não Para: 99 Ways Workshop #5,” http://www.mkltesthead.com/2013/07/99-ways-workshop-5-learn-when-to.html, 2013, Último acesso: Janeiro de 2016.

[Fonte 41] A. O'Doherty e Borland Software Corporation, “Manual vs. Automated Testing – explorado,” http://blog.borland.com/manualvsautomatedtesting/192/, 2013, Último acesso: Janeiro de 2016.

[Fonte 42] Gerrard Consulting, “Teste de regressão – o que para automatizar e como”, http://gerrardconsulting.com/sites/default/files/PaulGerrardRegressionTestingWhatAndHow.pptx, 1997, Last acessado em: janeiro de 2016.

[Fonte 43] Caritor Inc., “ROI em Automação de Testes - A Abordagem simples, mas poderosa”, http://www.cfoworld.co.uk/white-paper/software/4055/roi-on-test-automation–a-simple-yetpower-approach/, 2007, Último acesso: janeiro de 2016.

[Fonte 44] R. Padmanaban, “Fontes de Retorno do Investimento (ROI) em Automação de Testes ,” http://www.qainfotech.com/blog/2012/07/sources-of-return-on-investment-roi-in-test-automation/, 2012, Último acesso: janeiro de 2016.

[Fonte 45] B. Padilla, “Automação de Testes – Conhecendo Quando automatizar”, http://oshyn.com/software-development/test-automation-when-automate, 2012, Último acesso: janeiro de 2016.

Machine Translated by Google

[Fonte 46] Atlassian, “Test Automation and Best Practices,” https://www.atlassian.com/test-automation, Último acesso: janeiro de 2016.

[Fonte 47] Infosys, “Test Automation for Effective Post Deployment testing”, http://www.infosys.com/[illegible] 2007, Último acesso: janeiro de 2016.

[Fonte 48] D. Graham, “Test Automation Patterns,” http: //www.dorothygraham.co.uk/patterns/desPatterns/index.html, Último acesso: janeiro de 2016.

[Fonte 49] SK Gogineni e S. Kuchi, “Processo de automação de teste”, http://stepauto.com/images/Santhi,%20Srihari IBM%20pune.pdf, 2007, Último acesso: janeiro de 2016.

[Fonte 50] http:// IM Testy, “ROI de automação de teste” blogs.msdn.com/b/imtesty/archive/2009/09/02/test-automation-roi-part-ii.aspx, 2009, Último acesso: janeiro de 2016.

[Fonte 51] A. Narayanan e Aspire Systems, “Test Automation ROI Calculator,” http://www.aspiresys.com/WhitePapers/whitepaper_Test_Automation_ROI_Calculator.pdf, 2010, Último acesso: janeiro de 2016.

[Fonte 52] S. Carroll, “Test Automation: The Whens, Hows, and Whys,” http://www.deriskit.com/news-article_test-automation-the-whens-hows-and-whys.html, 2014, Último acesso: janeiro de 2016.

[Fonte 53] DW Johnson, “Test automation: When, how and how ,” http://searchsoftwarequality.techtarget.com/tip/ Test-automation-When-how-and-how-much, 2011, Último acesso: Janeiro de 2016.

[Fonte 54] AppLabs, “Teste de Automação de Valor de Negócios”, http://www.informationweek.com/whitepaper/download/showPDF?articleID=6630000, 2008, Último acesso: janeiro de 2016.

[Fonte 55] L. Crispin, “Testing in the Fast Lane: Automating Acceptance Testing in an Extreme Programming Environment”, na Software Test Automation Conference, 2002, Último acesso: janeiro de 2016.

[Fonte 56] E. Adams, “The Business Argument for Investing in Test Automation,” http://www.ibm.com/developerworks/rational/library/content/RationalEdge/dec02/TestAutomation_ TheRationalEdge_Dec2002.pdf, 2002, Último acessado em: janeiro de 2016.

[Fonte 57] M. Kelly, “O ROI da Automação de Testes ”, http://www.sqetraining.com/sites/default/files/articles/ XDD8502filelistfilename1_0.pdf, 2007, Último acesso: maio de 2015.

[Fonte 58] K. Stobie, “Demasiada automação ou insuficiente? Quando automatizar”, http://www.uploads.pnsqc.org/2009/papers/Keith_Stobie_Too_Much_Automation_paper.pdf, 2009, Último acesso: janeiro de 2016.

[Fonte 59] S. Rowe, “Too Much Test Automation?”, http: //blogs.msdn.com/b/steverowe/archive/2005/01/28/362668.aspx, 2005, Último acesso: janeiro de 2016.

[Fonte 60] Vários membros da comunidade StackExchange, “Quais são as boas diretrizes para determinar quando automatizar um teste”, http://sqa.stackexchange.com/questions/822/what-are-good-guidelines-to- determine-when-to-automate-a-test, 2012, Último acesso: janeiro de 2016.

[Fonte 61] mVerify Corporation, “Qual é o meu ROI de teste?” http://robertvbinder.com/wp-content/uploads/sites/4/2011/06/ Whats-My-Testing-ROI1.pdf, 2011, Último acesso: janeiro de 2016.

[Fonte 62] EJ Correia, “Qual é o seu índice de automação?” http://www.testpoint.com.au/attachments/093_What's%20Your%20Automation%20Index.pdf, 2007, Último acesso: janeiro de 2016.

[Fonte 63] BQurious Software Inc., “When and How Much to Automate,” http://www.bqurious.com/ when-and-how-much-to-automate, 2014, Último acesso: janeiro de 2016.

[Fonte 64] O. Laursen, “Quando testes automatizados fazem sentido, e quando não fazem”, http://ole-laursen.blogspot.com.tr/2012/03/ when-automated-tests-make-sense -and.html, 2012, Último acesso: janeiro de 2016.

[Fonte 65] B. Marick, “Quando um teste deve ser automatizado?” , http://www.exampler.com/testing-com/writings.html, 1998, Último acesso: janeiro de 2016.

[Fonte 66] Funcionário da A. M. , “When To Automate,” http://blogs.msdn.com/b/micahel/archive/2005/12/14/whentoautomate.aspx, 2005, Último acesso: janeiro de 2016.

[Fonte 67] D. Weiss, “When to Automate Testing”, http: //davidweiss.blogspot.com.tr/2006/08/when-to-automate-testing. html, 2006, Último acesso: janeiro de 2016.

[Fonte 68] J. Fernandes e AD Fonzo, “When to Automatate Your Testing (and When Not To)”, http://www.oracle.com/technetwork/cn/articles/when-to-automate-testing- 1-130330.pdf, 2013, Último acesso: janeiro de 2016.

[Fonte 69] TestingWhiz, “When to do Manual Testing and when to Automated Testing,” http://blog.testing-whiz.com/2011/12/when-to-do-manual-testing-and-when-to .html, 2011, Último acesso: janeiro de 2016.

[Fonte 70] S. Rowe, “Quando testar manualmente e quando automatizar”, http://blogs.msdn.com/b/steverowe/archive/2008/02/ 26/when-to-test-manually-and-when-to-automate.aspx, 2008, Último acesso: janeiro de 2016.

[Fonte 71] Grupo de testes intensos, “Por que e quando realizar testes de automação”, http://intensetesting.wordpress.com/2013/11/07/why-and-when-to-perform-automation-testing/, 2013 , Último acesso: jan. 2016.

[Fonte 72] Software Testing Class, “Why How and When to Automate Testing,” http://www.softwaretestingclass.com/ why-how-and-when-to-automate-software-testing/, 2014, Último acesso: Janeiro de 2016.

Vídeos do YouTube

[Fonte 73] D. Graham, “Intelligent Mistakes in Test Automation,” https://www.youtube.com/watch?v=Dbim6ZkSHBg, 2013, Último acesso: janeiro de 2016.

[Fonte 74] Rice Consulting Services Inc., “What to Automate First,” https://www.youtube.com/watch?v=eo66ouKGyVk, 2014, Último acesso: janeiro de 2016.

Ferramentas

[Fonte 75] IBM Rational, “IBM Rational Quality Management ROI Calculator,” em http://www-01.ibm.com/software/rational/offers/testing/roi/tool/ROI_Rational.html, Último acesso: jan. 2016.

[Fonte 76] Elbrus Ltd., “Return on Investment Calculator for Test Automation”, em http://www.elbrus.com/services/test_automation_roi_calc/, Último acesso: janeiro de 2016.

[Fonte 77] GlobalNow Inc., “Calculadora de ROI da GlobalNowIT”, em http://globalnowit.com/calculator/, Último acesso: janeiro de 2016.

[Fonte 78] Automated Testing Institute, “Test Automation ROI Calculator”, em http://www.automatedtestinginstitute.com/home/index.php?Itemid=65, Último acesso: janeiro de 2016.

Referências

[1] HP Capgemini. Sogetti, Relatório de qualidade mundial 2014-2015. www.sogeti.com/solutions/testing/wqr/ , Último acesso: setembro de 2015.

[2] T. Britton, L. Jeng, G. Carver, P. Cheak e T. Katzenellenbogen, “Software de depuração reversível”, Universidade de Cambridge, Judge Business School, *Tehnical Report,* 2013.

[3] O. Taipale, J. Kasurinen, K. Karhu, K. Smolander, Trade-off entre testes de software automatizados e manuais, Int. J. Syst. Garantia Eng. Gerenciar. 2 (2011) 114-125.

Machine Translated by Google

[4] D. Huizinga, A. Kolawa, Automated Defect Prevention: Best Practices in Software Management, Wiley-IEEE Computer Society Press, 2007.
[5] J. Bach, óleo de cobra de automação de teste, em: Proceedings of International Conference e Exposição sobre Testes de Computador, 1997.
[6] DM Rafi, KRK Moses, K. Petersen, MV Mantyla, Benefícios e limitações de testes automatizados de software - revisão sistemática da literatura e praticante pesquisa, em: Workshop Internacional sobre Automação de Teste de Software, 2012, págs. 36–42.
[7] D. Hoffman, Análise de custo-benefício da automação de teste, Análise de teste de software sis and Review Conference (STARWEST), 1999.
[8] V. Garousi, J. Zhi, Uma pesquisa de práticas de teste de software no Canadá, J. Syst. Softw. 86 (2013) 1354-1376.
[9] V. Garousi, A. Cos¸ kunçay, AB Can, O. Demirörs, Uma pesquisa de práticas de engenharia de software na Turquia, J. Syst. Softw. 108 (2015) 148-177.
[10] K. Stobi, Demasiada automação ou insuficiente? Quando automatizar o teste, Pa Conferência de Qualidade de Software do Noroeste, 2009.
[11] DJ Mosley, BA Posey, Automação de Teste de Software Just Enough, Prentice Hall Profissional, 2002.
[12] C. Kaner, J. Bach, B. Pettichord, Lições Aprendidas em Teste de Software, John Wiley & Sons Inc, 2001.
[13] V. Garousi, Uma abordagem sistemática para automação de teste de software e como aumentar seu ROI, Invited Talk, TestIstanbul Industry Conference, 2013.
[14] V. Garousi, Tendências recentes em teste de software: oportunidades para colaborações entre indústria e academia, palestrante convidado, YouTube Corporation, 30 de junho de 2010.
[15] V. Garousi, K. Herkiloglu ÿ , Selecionando os tópicos certos para colaborações indústria-acadêmica em testes de software: um relato de experiência, IEEE International Conference on Software Testing, Verification, and Validatio, 2016.
[16] Z. Sahaf, V. Garousi, D. Pfahl, R. Irving, Y. Amannejad, Quando automatizar o teste de software? Suporte à decisão baseado em dinâmica de sistemas – um caso industrial estudo, em: Proc. da Conferência Internacional sobre Processos de Software e Sistemas, 2014, págs. 149-158.
[17] KC Archie, OR Fonorow, MC McGould, RE McLear, EC Read, EM Schaefer, et al., “Sistema de automação de teste”, ed: US Patent #US5021997, 1991.
[18] V. Garousi, R. Kotchorek, M. Smith, Custo-benefício do teste e densidade de defeitos: Um estudo de caso na plataforma Android, Adv. Computar. 89 (2013) 163-206.
[19] J. Feldstein, Como recrutar, motivar e energizar o teste superior, dezembro de 2014, Último acesso: http://www.youtube.com/watch?v=PyhtoQz7RHY.
[20] G. Meszaros, xUnit Test Patterns, Pearson Education, 2007 http://xunitpatterns.com.
[21] A. Page, K. Johnston, How We Test Software at Microsoft, Microsoft Press, 2008.
[22] JA Whittaker, J. Arbon, J. Carollo, How Google Tests Software, Addison-Wesley Profissional, 2012.
[23] P. Ammann, J. Offutt, Introdução ao Teste de Software, Universidade de Cambridge Imprensa, 2008.
[24] K. Stobie, “Demasiada automação ou insuficiente? Quando automatizar os testes”, 2009.
[25] S. Desikan, G. Ramesh, Teste de Software: Princípios e Práticas, Pearson Education India, 2006.
[26] E. Dustin, T. Garrett, B. Gauf, Implementação de testes automatizados de software: Como para economizar tempo e reduzir custos enquanto aumenta a qualidade, Addison-Wesley Profes sional, 2009.
[27] A. Ampatzoglou, A. Ampatzoglou, A. Chatzigeorgiou, P. Avgeriou, O aspecto financeiro da gestão da dívida técnica: Uma revisão sistemática da literatura, Informar. Softw. Tecnol. 64 (2015) 52–73 8//.
[28] RL Glass e T. DeMarco, *Software Creativity 2.0:* desenvolvedor.ÿ Books, 2006.
[29] RT Ogawa, B. Malen, Rumo ao rigor em revisões de literaturas multivocais: aplicando o método de estudo de caso exploratório, Revi. Edu. Res. 61 (1991) 265-286.
[30] WF Whyte, Pesquisa Ação Participativa, Publicações SAGE, 1990.
[31] KM Benzies, S. Premji, KA Hayden, K. Serrett, revisões do estado das evidências: vantagens e desafios da inclusão da literatura cinzenta, Worldv. Baseado em evidências Enfermagem 3 (2006) 55–61.
[32] Q. Mahood, D. Van Eerd, E. Irvin, Procurando literatura cinzenta para sistemática comentários: desafios e benefícios, Res. Sin. Métodos 5 (2014) 221–234.
[33] S. Hopewell, M. Clarke, S. Mallett, literatura cinzenta e revisões sistemáticas, em: HR Rothstein, AJ Sutton, M. Borenstein (Eds.), Publication Bias in Metanálise: Prevenção, Avaliação e Ajustes, John Wiley & Sons, 2006.
[34] H. S, M. S, C. M, E. M, Literatura cinzenta em meta-análises de ensaios randomizados de intervenções de saúde, Cochrane Datab. Sistema Rev. (2007).
[35] E. Tom, A. Aurum, R. Vidgen, Uma exploração da dívida técnica, J. Syst. Softw. 86 (2013) 1498-1516.
[36] I. Kulesovs, "teste de aplicativos iOS", vol. 3, pp. 138–150, 2015.
[37] M. Sulayman, E. Mendes, Uma revisão sistemática da literatura do processo de software aperfeiçoamento em pequenas e médias empresas web, em: D. Sl´ezak, T.-h. Kim, A. Kiumi, T. Jiang, J. Verner, S. Abrahão (Eds.), Advances in Software Engineering ing, 59, Springer Berlin Heidelberg, 2009, pp. 1–8.
[38] A. Yasin, MI Hasnain, Sobre a qualidade da literatura cinzenta e seu uso na síntese de informações durante revisões sistemáticas da literatura Tese de mestrado, Blekinge Instituto de Tecnologia, Suécia, 2012.
[39] V. Garousi, M. Felderer, MV Mäntylä, A necessidade de (mais) revisões de literatura multivocal em engenharia de software, Em revisão, Conferência Internacional sobre Avaliação e Avaliação em Engenharia de Software (EASE) *paper PDF:,* 2016 https://goo.gl/O0JPsF .
[40] V. Garousi, Online Paper Repository for Systematic Mapping of Secondary studies in software testing, http://goo.gl/Oxb0x8, Último acesso: setembro de 2015.
[41] W. Afzal, R. Torkar, R. Feldt, Um estudo de mapeamento sistemático sobre não-funcionais teste de software baseado em pesquisa, em: Conferência Internacional sobre Engenharia de Software e Engenharia do Conhecimento, 2008, pp. 488–493.
[42] PAdMS Neto, IdC Machado, JD McGregord, ESd Almeida, SRDL Meira, Um estudo de mapeamento sistemático de testes de linhas de produtos de software, Inform. Softw. Tecnol. 53 (2011) 407-423.
[43] I. Banerjee, B. Nguyen, V. Garousi, A. Memon, interface gráfica do usuário (GUI) testes: mapeamento sistemático e repositório, Informe. Softw. Tecnol. 55 (2013) 1679-1694.
[44] C. Çatal, D. Mishra, Priorização de casos de teste: um estudo de mapeamento sistemático, Softw. Qualidade J. 21 (2013) 445–478.
[45] VG Yusifoglu ÿ Y. Amannejad, A. Betin-Can, Engenharia de código de teste de software: a mapeamento sistemático, J. Informar. Softw. Tecnol. (2014).
[46] ACD Neto, R. Subramanyan, M. Vieira, GH Travassos, Uma pesquisa sobre abordagens de teste baseadas em modelo - uma revisão sistemática, em: Proceedings of the ACM Workshop Internacional sobre Avaliação Empírica de linguagens e tecnologias de engenharia de software, 2007.
[47] B. Haugset, GK Hanssen, Teste de Aceitação Automatizado - uma revisão da literatura e um estudo de caso industrial, em: Agile Conference, 2008, pp. 27–38.
[48] PK Singh, OP Sangwan, A. Sharma, Uma revisão sistemática sobre técnicas e ferramentas de teste de mutação baseadas em falhas para programas Aspect-J, em: IEEE International Conferência de Computação Avançada, 2013, pp. 1455–1461.
[49] S. Dogan , A. Betin-Can, V. Garousi, teste de aplicativos da Web: um litro sistemático revisor de atura, J. Syst. Softw. 91 (2014) 174-201.
[50] U. Kanewala, JM Bieman, Testando software científico: uma literatura sistemática revisar, informar. Softw. Tecnol. 56 (2014) 1219-1232 10//.
[51] D. Lee, M. Yannakakis, Princípios e métodos de teste de máquinas de estado finito - uma pesquisa, em: Proceedings of the IEEE, 84, 1996, pp. 1090-1123.
[52] S. Yoo, M. Harman, Minimização de testes de regressão, seleção e priorização: uma pesquisa, Softw. Teste. Verif. Confiável. 22 (2012) 67-120.
[53] M. Bozkurt, M. Harman, Y. Hassoun, Teste e verificação em serviços orientados arquitetura-a Survey, Softw. Teste. Verif. Confiável. 23 (2013) 261-313.
[54] M. Shirole, R. Kumar, geração de casos de teste baseado em modelo comportamental UML: um sur vey, ACM SIGSOFT Softw. Eng. Notas 38 (2013) 1–13.
[55] M. Harman, P. McMinn, M. Shahbaz, S. Yoo, Uma pesquisa abrangente de tendências em oráculos para teste de software, IEEE Trans. Softw. Eng. (2014).
[56] B. Kitchenham, P. Brereton, D. Budgen, O valor educacional dos estudos de mapeamento da literatura de engenharia de software, em: Engenharia de Software, 2010 ACM/IEEE 32ª Conferência Internacional em, 2010, pp. 589–598.
[57] RW Schlosser, “O papel das revisões sistemáticas na prática, pesquisa e desenvolvimento baseados em evidências”. Resumo Técnico FOCUS #15, http://ktdrr.org/ktlibrary/articles_pubs/ncddrwork/focus/focus15/Focus15.pdf, 2006.
[58] M. Bell, P. Cordingley C. Isham, R. Davis, “Relatório de praticante profissional uso de revisão de pesquisa: envolvimento do profissional em e/ou com pesquisa”, Coventry: CUREE, GTCE, LSIS e NTRP. Disponível em: http://www.curee-paccts.com/node/2303, 2010.
[59] H. Aveyard, fazendo uma revisão da literatura em saúde e assistência social: uma prática Guide: A Practical Guide Paperback, 2ª edição, Open University Press, 2010.
[60] V. Garousi, A. Mesbah, A. Betin-Can, S. Mirshokraie, Um mapeamento sistemático de teste de aplicativos da web, Informe. Softw. Tecnol. (2013).
[61] I. Banerjee, B. Nguyen, V. Garousi, A. Memon, interface gráfica do usuário (GUI) testes: mapeamento sistemático e repositório, Informe. Softw. Tecnol. (2013).
[62] N. Bencomo, S. Hallsteinsen, E. Santana de Almeida, Uma Visão da Dinâmica Paisagem da linha de produtos de software, computação IEEE. 45 (2012) 36–41.
[63] K. Petersen, R. Feldt, S. Mujtaba, M. Mattsson, estudos de mapeamento sistemático em engenharia de software, em: 12ª Conferência Internacional sobre Avaliação e Avaliação em Engenharia de Software, 2008, pp. 71–80.
[64] K. Petersen, S. Vakkalanka, L. Kuzniarz, Diretrizes para a condução sistemática estudos de mapeamento em engenharia de software: Uma atualização, Informe. Softw. Tecnol. 64 (2015) 1–18.
[65] K. Petersen, R. Feldt, S. Mujtaba, M. Mattsson, estudos de mapeamento sistemático em engenharia de software, apresentado na 12th International Conference on Evaluation and Assessment in Software Engineering (EASE), 2008.
[66] B. Kitchenham, S. Charters, Diretrizes para realizar revisões sistemáticas da literatura em engenharia de software”, em engenharia de software baseada em evidências, Evi nce-Based Softw. Eng. (2007).
[67] S. Ali, LC Briand, H. Hemmati, RK Panesar-Walawege, Uma revisão sistemática de a aplicação e investigação empírica da geração de casos de teste baseada em pesquisa, IEEE Trans. Softw. Eng. 36 (2010) 742-762.
[68] F. Elberzhager, J. Münch, VTN Nha, Um estudo de mapeamento sistemático sobre a combinação de técnicas de garantia de qualidade estáticas e dinâmicas, Inform. Softw. Tecnol. 54 (2012) 1–15.
[69] V. Garousi, Classificação e análise de tendências de livros UML (1997-2009), J. Softw. Sistema Modelo. (SoSyM) (2011).
[70] V. Garousi, Y. Amannejad, A. Betin-Can, Engenharia de código de teste de software: um mapeamento sistemático, Jf Inform. Softw. Tecnol. 58 (2015) 123-147.
[71] M. Ivarsson, T. Gorschek, Um método para avaliar rigor e relevância industrial de avaliações de tecnologia, Empirical Softw. Eng. 16 (2011) 365-395.

Machine Translated by Google

[72] M. Fewster, Erros comuns na automação de teste, http://www.agileconnection. com/sites/default/files/article/file/2012/XDD2901filelistfilename1_0.pdf, 2012.
Último acesso: setembro de 2015.
[73] K. Karhu, T. Repo, O. Taipale, K. Smolander, Observações empíricas sobre automação de teste de software, em: Verificação e validação de teste de software, 2009. ICST '09. Conferência Internacional sobre, 2009, pp. 201–209.
[74] B. Pettichord, Sete etapas para testar o sucesso da automação, (2001), Último acesso: setembro de 2015 https://www.prismnet.com/ÿwazmo/papers/seven_steps.
[75] C. Wohlin, Diretrizes para bola de neve em estudos de literatura sistemática e uma replicação em engenharia de software, em: apresentado nos Anais da 18ª Conferência Internacional sobre Avaliação e Avaliação em Engenharia de Software, Londres, Inglaterra, Reino Unido, 2014.

[76] V. Garousi, MV Mäntylä, repositório de papel on-line para o mlr sobre 'quando e o que automatizar no teste de software?' (2016), Último acesso: janeiro http://goo.gl/zwY1sj

[77] DS Cruzes e T. Dybå, "Synthesizing Evidence in Software Engineering Research," em *Proceedings of the ACM-IEEE International Symposium on Empirical Software Engineering and Measurement,* 2010
[78] DS Cruzes, T. Dybå, Passos recomendados para síntese temática em engenharia de software, em: Proc. Simpósio Internacional sobre Engenharia e Medição de Software Empírico, 2011, pp. 275–284.
[79] DS Cruzesa, T. Dybåb, Síntese de pesquisa em engenharia de software: um estudo terciário, Inform. Softw. Tecnol. 53 (2011) 440-455.
[80] GS Waliaa, JC Carverb, Uma revisão sistemática da literatura para identificar e classificar erros de requisitos de software, Informe. Softw. Tecnol. 51 (2009) 1087-1109.
[81] S. Ali, LC Briand, H. Hemmati, RK Panesar-Walawege, Uma revisão sistemática da aplicação e investigação empírica da geração de casos de teste baseada em pesquisa, IEEE Trans. Softw. Eng. 36 (2010) 742-762.
[82] H. Cooper, LV Hedges, JC Valentine, The Handbook of Research Synthesis and Meta-Analysis, 2ª ed, Russell Sage Foundation, 2009.
[83] MB Miles, AM Huberman, J. Saldana, Análise de Dados Qualitativos: Métodos A Sourcebook, Terceira Edição, SAGE Publications Inc, 2014.
[84] B. Daniel, V. Jagannath, D. Dig, D. Marinov, ReAssert: sugerindo reparos para testes de unidade quebrados, em: apresentado nos Anais da Conferência Internacional IEEE/ACM 2009 sobre Engenharia de Software Automatizada, 2009.
[85] Y. Amannejad, V. Garousi, R. Irving, Z. Sahaf, Uma abordagem baseada em pesquisa para suporte à decisão de automação de teste de software econômica e um estudo de caso industrial, em: Proc. of International Workshop on Regression Testing, co-localizado com a Sixth IEEE International Conference on Software Testing, Verification, and Validation, 2014, pp. 302–311.

[86] M. Kelly, O ROI da automação de testes, http://www.sqetraining.com/sites/default/files/articles/XDD8502filelistfilename1_0.pdf, 2007. Último acesso: maio de 2015.
[87] C. Wohlin, P. Runeson, M. Höst, MC Ohlsson, B. Regnell, A. Wesslén, Experimentation in Software Engineering: An Introduction, Kluwer Academic Publishers, 2000.

[88] E. Arisholm, H. Gallis, T. Dyba, DIK Sjoberg, Avaliando programação em par com relação à complexidade do sistema e experiência do programador, IEEE Trans. Softw. Eng. 33 (2007) 65-86.