Uma Avaliação da Adoção e Cobertura dos Critérios WCAG 2.1 em Projetos Open-Source no GitHub

Gabriel Estevão Sobrinho[1], **Luana Gonçalves Fleury**[1]

[1]Instituto de Ciências Exatas e Informática
Departamento de Engenharia de Software
Pontifícia Universidade Católica de Minas Gerais (PUC Minas)
CEP 30535-000 – Belo Horizonte – MG – Brazil

{gensobrinho, luana.fleury4}@gmail.com

Abstract. *This study is situated in the field of Software Engineering, focusing on digital accessibility in web applications. The central problem investigated is the lack of data regarding the practical adoption of automated accessibility evaluation tools, such as AXE and Lighthouse, in open-source projects hosted on GitHub. The main objective is to quantitatively analyze the usage frequency of these tools, their coverage of WCAG 2.1 success criteria, and the relationship between their adoption and the level of accessibility achieved in projects. To this end, JavaScript scripts were used to collect data from 19.488 public repositories and to evaluate a final sample of 237 projects, identifying the presence of tools in CI/CD pipelines and assessing the effectiveness of these solutions. The achieved results include the mapping of the current adoption scenario and the identification of the most widely used tools.*

Resumo. *Este trabalho insere-se na área de Engenharia de Software, com foco em acessibilidade digital em aplicações web. O problema central investigado é a carência de dados sobre a adoção prática de ferramentas automatizadas de avaliação de acessibilidade, como AXE e Lighthouse, em projetos open-source hospedados no GitHub. O objetivo principal é analisar quantitativamente a frequência de uso dessas ferramentas, a cobertura dos critérios de sucesso da WCAG 2.1 e a relação entre sua adoção e o nível de acessibilidade alcançado nos projetos. Para isso, foram utilizados scripts em JavaScript para coletar dados de 19.488 repositórios públicos e avaliar uma amostra final de 237 projetos, identificando a presença de ferramentas nos pipelines de CI/CD e avaliando a eficácia dessas soluções. Os resultados alcançados incluem o mapeamento do cenário atual de adoção e a identificação das ferramentas mais empregadas.*

Bacharelado em Engenharia de *Software* - PUC Minas
Trabalho de Conclusão de Curso (TCC)

Orientador de conteúdo (TCC I): Joana Souza - joanasouza@pucminas.br
Orientador de conteúdo (TCC I): João Pedro Batisteli - jp.batisteli@hotmail.com
Orientador de conteúdo (TCC I): Leonardo Vilela - leonardocardoso@pucminas.br
Orientador acadêmico (TCC I): Cleiton Tavares - cleitontavares@pucminas.br
Orientador do TCC II: Joana Souza - joanasouza@pucminas.br

Belo Horizonte, 23 de Setembro de 2025.

1. Introdução

A acessibilidade digital constitui um aspecto importante no desenvolvimento de software, especialmente em aplicações web que impactam diferentes usuários com necessidades variadas. As diretrizes de acessibilidade na web, como as *Web Content Accessibility Guidelines 2.1* (WCAG)[1], estabelecem critérios para a construção de interfaces digitais acessíveis a todos os usuários, independentemente de suas condições físicas, sensoriais ou cognitivas [Abu Doush et al. 2023]. A importância dessas diretrizes é evidenciada pelo comportamento dos usuários com deficiência, dos quais 71% abandonam sites quando encontram barreiras de acessibilidade, indicando a necessidade de implementar práticas de desenvolvimento que garantam a inclusão digital [Dias et al. 2022]. Nesse contexto, ferramentas automatizadas de avaliação de acessibilidade em aplicações web surgem como recursos para identificar e corrigir problemas de conformidade com os padrões estabelecidos, como AXE[2] e Lighthouse [3], permitindo a identificação de problemas de conformidade durante o ciclo de desenvolvimento.

No entanto, mesmo com o suporte dessas ferramentas e diretrizes, ainda existem desafios relacionados à adoção prática e à eficácia dos relatórios gerados. No contexto de projetos de *open-source* ("código aberto", em português), frequentemente disponíveis em plataformas como GitHub, observa-se que nem sempre essas ferramentas são integradas aos *pipelines* [4] de integração contínua (CI/CD)[5], o que pode comprometer a qualidade e acessibilidade do conteúdo produzido [Sane 2021].

A literatura destaca que a integração de testes automatizados de acessibilidade nesses *pipelines* é fundamental para identificar precocemente barreiras de acessibilidade, aumentar a qualidade do código e facilitar o cumprimento de requisitos legais e normativos [Velleman et al. 2017]. Conforme discutido por Sane (2021), essa prática contribui para a redução de custos associados à correção tardia de problemas, minimiza a dívida técnica relacionada à inclusão digital e promove entregas contínuas de software mais alinhadas às diretrizes internacionais. Além disso, a automação dos testes de acessibilidade no CI/CD incentiva uma cultura organizacional voltada à responsabilidade social e inovação, tornando a acessibilidade parte integrante do ciclo de vida do software, em vez de uma etapa isolada [Velleman et al. 2017]. Diante disso, **o problema investigado neste trabalho é a lacuna de dados quantitativos que caracterizem a adoção de ferramentas automatizadas de avaliação de acessibilidade para a verificação da conformidade com os critérios da WCAG 2.1 no contexto de integração contínua em projetos de código aberto disponíveis no GitHub.**

A investigação deste problema apresenta importância tanto no aspecto social quanto no técnico do desenvolvimento de software e pode ampliar o alcance de aplicações inclusivas e promover práticas de desenvolvimento *open-source*. Nesse sentido, a implementação de testes automatizados de acessibilidade em ambientes de integração

[1] `https://www.w3.org/WAI/standards-guidelines/wcag/`

[2] `https://www.deque.com/axe/`

[3] `https://developer.chrome.com/docs/lighthouse/performance/performance-scoring?hl=pt-br`

[4] *Pipelines*: fluxos automatizados de integração e entrega contínua que coordenam as etapas de compilação, verificação, testes e implantação de software.

[5] Sigla para *Continuous Integration/Continuous Delivery*, que significa Integração Contínua e Entrega Contínua.

contínua pode melhorar a qualidade do código, reduzir custos associados à dívida técnica e trazer maior competitividade empresarial [Sane 2021]. Adicionalmente, essa prática facilita o cumprimento de padrões internacionais e promove uma melhor experiência para usuários com deficiência [Ara and Sik-Lanyi 2025, Sane 2021]. Além disso, apesar das ferramentas existentes cobrirem apenas cerca de 44% dos critérios de acessibilidade estabelecidos pela WCAG 2.1 de forma totalmente automatizada, seu uso ainda supera a avaliação manual em termos de eficiência e aplicabilidade em projetos de grande escala [Abu Doush et al. 2023]. A análise da aplicabilidade dessas ferramentas é direcionada para plataformas como o GitHub, que se consolidou como o principal ecossistema para projetos *open-source*. Em 2025, a plataforma já abrigava mais de 180 milhões de desenvolvedores [6], o que o torna um ambiente representativo para o estudo das práticas de desenvolvimento de software em larga escala.

O objetivo geral deste trabalho é **avaliar a adoção de ferramentas automáticas para a verificação da conformidade com os critérios da WCAG 2.1 em *pipelines* de integração contínua de repositórios *open-source* hospedados no GitHub, utilizando ferramentas automatizadas**. Para alcançar este objetivo, foram definidos os seguintes objetivos específicos: (i) quantificar a proporção dos critérios de sucesso da WCAG 2.1 que podem ser verificados de forma totalmente automatizada pelas ferramentas, identificando o nível de cobertura automática que cada uma oferece; (ii) comparar a abrangência dos critérios de sucesso cobertos automaticamente por cada ferramenta de avaliação de acessibilidade com a frequência de utilização dessas ferramentas; e (iii) avaliar a correlação entre a adoção das ferramentas e a eficácia da acessibilidade nas aplicações dos projetos que as integram.

Como resultado, este trabalho mapeou o cenário atual de adoção de ferramentas automáticas de acessibilidade em projetos web *open-source* no GitHub. A investigação caracterizou como essas ferramentas são incorporadas e quais são mais utilizadas, avaliando sua eficácia na detecção de problemas de acessibilidade. A partir desses dados, foi estabelecida a correlação entre a adoção das ferramentas e o nível de conformidade alcançado pelos projetos. O objetivo final do estudo foi fornecer insumos para a proposição de recomendações práticas que possam orientar comunidades de desenvolvimento na implementação mais efetiva dessas ferramentas em seus projetos.

As demais seções deste artigo organizam-se da seguinte maneira: a Seção 2 apresenta a Fundamentação Teórica, abordando os principais conceitos utilizados neste estudo. Em seguida, a Seção 3 apresenta trabalhos relacionados relevantes para o problema de pesquisa aqui abordado. Ademais, a Seção 4 detalha a metodologia empregada neste estudo. A Seção 5 apresenta os resultados obtidos, oferecendo uma análise aprofundada das métricas coletadas diante dos objetivos da pesquisa. A Seção 6 discute as implicações e interpretações desses resultados. Posteriormente, a Seção 7 aborda as ameaças à validade do estudo. Por fim, a Seção 8 conclui o artigo, resumindo os achados e apontando direções para trabalhos futuros.

[6] https://github.blog/news-insights/octoverse/octoverse-a-new-developer-joins-github-every-second-as-ai-leads-typescript-to-1/

2. Fundamentação Teórica

Esta seção apresenta os principais conceitos, teorias e mecanismos que sustentam este estudo.

2.1. Acessibilidade Digital e Diretrizes WCAG

A acessibilidade digital é o conceito que garante a todos os usuários a capacidade de acessar, compreender e interagir com conteúdos na web[Macakoglu and Peker 2022]. O principal referencial internacional para garantir esse acesso são as WCAG, publicadas pelo W3C [Macakoglu and Peker 2022]. As WCAG organizam critérios técnicos em princípios como perceptível, operável, compreensível e robusto, subdivididos em três níveis de conformidade: A, AA, AAA. O nível A corresponde aos requisitos essenciais para garantir o acesso mínimo, enquanto o nível AAA contempla critérios mais avançados para pessoas com mais restrições no uso de aplicações. A adoção dessas diretrizes tornou-se padrão global e é frequentemente exigida por legislações internacionais [Sane 2021].

Apesar da existência desses padrões, muitos sites ainda apresentam barreiras significativas de acessibilidade, pois desenvolvedores nem sempre consideram adequadamente as necessidades de pessoas com deficiência durante o processo de criação e manutenção dos *websites* [Macakoglu and Peker 2022]. Além disso, embora as WCAG sejam amplamente utilizadas como referência técnica, sua implementação prática depende de processos contínuos de avaliação e ajuste, especialmente em ambientes dinâmicos como projetos *open-source* [Macakoglu and Peker 2022].

O crescimento da complexidade dos sistemas web e o aumento do volume de páginas demandaram soluções eficientes para avaliar a conformidade com padrões de acessibilidade [Dias et al. 2022]. Para lidar com esse desafio, o uso de ferramentas automatizadas de avaliação tornou-se uma prática central no desenvolvimento web. Essas soluções permitem analisar grandes quantidades de páginas de forma rápida e padronizada, facilitando a identificação de problemas objetivos de acessibilidade e apoiando equipes de desenvolvimento na busca por conformidade contínua [Abu Doush et al. 2023].

No desenvolvimento moderno de *software*, a aplicação dessas ferramentas de forma recorrente e automatizada é frequentemente realizada através de sua integração em *pipelines* de integração contínua, um conceito que será detalhado na próxima seção.

2.2. Integração de Ferramentas de Acessibilidade em Pipelines CI/CD

A Integração Contínua e a Entrega Contínua (CD, do inglês *"Continuous Delivery"*) são práticas técnicas fundamentais no desenvolvimento moderno de software. Por meio de *pipelines* automatizadas, essas abordagens permitem que todo código submetido ao repositório passe automaticamente por etapas de compilação, execução de testes e validações, promovendo agilidade, padronização e rastreabilidade no ciclo de vida do software [Sane 2021]. Em projetos *open-source*, onde há grande volume de contribuições e colaboração distribuída, o uso de CI/CD facilita a integração eficiente de mudanças e a manutenção da qualidade do projeto[Moser and Yli-Hukka Högback 2024].

No contexto da acessibilidade digital, a integração de ferramentas automáticas de avaliação nessas *pipelines* representa um avanço significativo. Ferramentas podem

ser configuradas para executar automaticamente a cada novo *commit*[7] ou *pull request*[8], analisando o código-fonte e identificando violações às diretrizes WCAG [Sane 2021]. Dessa forma, a automação dos testes de acessibilidade torna-se parte natural do fluxo de trabalho, permitindo que equipes identifiquem e tratem questões relacionadas à acessibilidade de maneira recorrente e alinhada às melhores práticas de engenharia de software[Kelasidou and Pettersson 2025].

3. Trabalhos Relacionados

A acessibilidade digital em aplicações web tem recebido atenção crescente, impulsionada pela necessidade de garantir inclusão e conformidade com diretrizes internacionais [Ara and Sik-Lanyi 2025, Macakoglu and Peker 2022]. A literatura aborda diversos métodos para avaliação e melhoria da acessibilidade, incluindo soluções automáticas e processos de integração contínua. No entanto, persistem desafios técnicos e organizacionais para a adoção efetiva dessas práticas, especialmente em projetos *open-source*. Esta seção apresenta trabalhos que tratam de temas correlatos.

A integração de ferramentas automatizadas de acessibilidade em *pipelines* de CI/CD é uma estratégia consolidada na literatura para aprimorar a acessibilidade do software [Sane 2021, Moser and Yli-Hukka Högback 2024, Kelasidou and Pettersson 2025]. Sane (2021) analisou especificamente o uso de AXE e Pa11y[9], destacando tanto os benefícios da automação quanto os desafios na análise do DOM. Complementarmente, estudos como os de Moser e Högback (2024) e Pettersson e Kelasidou (2025) investigaram, em ambientes controlados, a integração de um conjunto expandido de ferramentas, incluindo Asqatasun[10], HTML CodeSniffer[11], AChecker[12], WAVE[13] e Lighthouse. Embora esses estudos tenham demonstrado a importância e o potencial dessas ferramentas, eles também revelaram que a taxa de detecção automática de erros permanece modesta, reforçando a necessidade de análises mais aprofundadas.

Diante deste cenário, e considerando o potencial das soluções investigadas por esses três trabalhos para integração em ambientes de desenvolvimento contínuo, este estudo adotará o conjunto de ferramentas estabelecido por eles. O objetivo é ampliar o escopo da análise, passando de ambientes controlados para projetos *open-source* em larga escala, a fim de avaliar diferentes abordagens para a detecção automatizada de barreiras de acessibilidade no GitHub.

Para fundamentar a análise deste estudo, ele se apoia nos resultados de Abu Doush et al. (2023) [Abu Doush et al. 2023], que mapearam o potencial e os limites da automação na verificação dos critérios da WCAG 2.1. A pesquisa deles destaca os principais desafios enfrentados pelas ferramentas, como a incapacidade de validar a correção semântica e testar interações complexas, e quantifica a viabilidade da automação por

[7] *Commit*: Um conjunto de alterações salvas no histórico de um repositório.

[8] *Pull Request*: Uma solicitação para que as alterações feitas em uma ramificação (*branch*) sejam incorporadas à ramificação principal. É um mecanismo central para a revisão de código em equipe antes da integração.

[9] `https://pa11y.org/`

[10] `https://asqatasun.org/`

[11] `https://squizlabs.github.io/HTML_CodeSniffer/`

[12] `https://achecks.org/achecker/`

[13] `https://wave.webaim.org/?url=`

nível: 58% no nível A, 50% no nível AA e apenas 21% no nível AAA. Neste estudo, o mapeamento de Abu Doush et al. (2023) foi utilizado como referência teórica para contextualizar a cobertura das ferramentas analisadas. Dessa forma, o presente estudo complementa o trabalho dos autores ao aplicar seus achados em uma análise quantitativa em larga escala, investigando a adoção prática dessas ferramentas e sua cobertura real em todos os três níveis de conformidade em repositórios do ecossistema *open-source*.

No contexto da avaliação automatizada da acessibilidade web, Alsaeedi (2020) propôs um *framework* específico para comparar o desempenho de diferentes ferramentas de avaliação, introduzindo a métrica *Coverage Error Ratio* (CER). Essa métrica quantifica a capacidade de cada ferramenta de identificar erros únicos de acessibilidade em relação ao total de problemas detectados pelo conjunto de ferramentas analisadas. Por meio da aplicação do CER, foi possível obter uma análise objetiva sobre a abrangência e a efetividade das soluções avaliadas, permitindo identificar quais ferramentas apresentaram maior cobertura dos critérios de sucesso definidos pelas diretrizes WCAG. Para atender ao segundo objetivo específico, este trabalho adotará a métrica CER de Alsaeedi (2020) para realizar uma análise quantitativa da cobertura de cada ferramenta.

4. Materiais e Métodos

Nesta seção, são descritos os materiais e métodos que foram utilizados na condução deste estudo. A pesquisa desenvolvida neste trabalho é do tipo aplicada, de natureza quantitativa e exploratória, tendo como objetivo geral analisar a adoção prática de ferramentas automatizadas de avaliação de acessibilidade em projetos *open-source* hospedados no GitHub. Busca-se mapear padrões de implementação, cobertura dos critérios das diretrizes WCAG 2.1 e a eficácia dessas ferramentas automatizadas como meio para garantir a conformidade de acessibilidade nas aplicações. A abordagem quantitativa justifica-se pela necessidade de mensurar a frequência de uso das ferramentas, a extensão da cobertura automática dos critérios de sucesso e a correlação entre a adoção dessas soluções e a conformidade com padrões de acessibilidade. O caráter exploratório se deve à escassez de estudos específicos sobre o tema no ecossistema *open-source*.

A Figura 1 resume a metodologia adotada neste estudo, composta por cinco etapas principais: coleta de repositórios, filtragem manual de repositórios, execução das ferramentas nos repositórios coletados, cálculo de métricas e análise dos resultados.

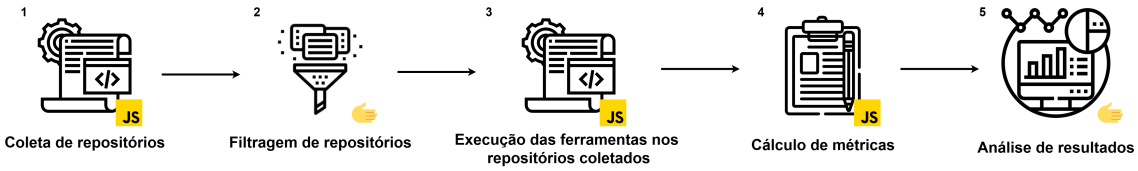

Figura 1. Fluxo metodológico em cinco etapas: coleta de repositórios, filtragem de repositórios, execução das ferramentas, cálculo de métricas e análise dos resultados.

O ambiente considerado para a realização da pesquisa compreende a plataforma GitHub, reconhecida como principal repositório global de projetos *open-source* e amplamente utilizada na literatura devido à facilidade de acesso via API (do inglês *Application*

Programming Interface) para coleta automatizada de dados. Neste estudo, foi empregado o uso de *scripts* desenvolvidos em JavaScript, que utilizam a API do GitHub com GraphQL[14] para extração dos dados necessários. Os repositórios foram coletados e organizados em ordem decrescente pelo número de estrelas, iniciando pelos mais populares e abrangendo projetos com menor visibilidade dentro do recorte definido, de modo a compor uma amostra diversificada e representativa. Para os fins deste estudo, uma aplicação web foi definida com o propósito principal de entrega de uma interface funcional ao usuário final através de um navegador, utilizando tecnologias da *Web Platform*. Ressalta-se que não foi feita nenhuma restrição quanto à linguagem ou tecnologia das aplicações web consideradas.

O mecanismo de funcionamento do *script* consiste em realizar consultas paginadas à API GraphQL do GitHub, ordenando os resultados pelo número de estrelas em ordem decrescente. Além disso, o *script* incorporou um critério de atividade mínima, retornando apenas repositórios que apresentaram pelo menos um *commit* nos doze meses anteriores à data da coleta, realizada entre agosto e setembro de 2025. Esse critério, amplamente utilizado em estudos de caracterização de projetos *open-source*, é empregado para definir um repositório ativo, permitindo excluir projetos abandonados ou inativos e garantindo que a análise se concentre em iniciativas com manutenção recente e relevância prática[Cánovas Izquierdo et al. 2017]. Para cada repositório retornado, foram analisados arquivos e configurações que possam indicar a presença de ferramentas automatizadas de avaliação de acessibilidade, incluindo elementos de *workflows* de integração contínua, arquivos de dependências e configurações específicas dessas ferramentas. A identificação ocorreu por meio da busca de termos associados às ferramentas, considerando menções diretas ou indiretas nos artefatos analisados.

Após a execução automática do processo de mineração, foi conduzida uma etapa de filtragem manual com o objetivo de eliminar falsos positivos. Essa filtragem consistiu na inspeção individual dos repositórios identificados para verificar se o uso das ferramentas de acessibilidade ocorria no contexto de aplicações *web* com *pipelines* de integração contínua (CI/CD) ativos. Foram removidos casos em que as ferramentas estavam presentes apenas como dependências transitivas, exemplos de configuração, tutoriais ou testes isolados, bem como repositórios cujo propósito principal não envolvia a entrega de uma aplicação *web* funcional. Essa etapa de filtragem foi crucial para garantir que a amostra final fosse composta exclusivamente por projetos que são, de fato, aplicações web funcionais, excluindo-se repositórios que, embora mencionem as ferramentas, atuam como bibliotecas, *frameworks*, *templates* ou outros artefatos que não representam o produto final de um software web. Somente os repositórios que atenderam a esses critérios permaneceram na amostra final utilizada para análise.

Em seguida, uma etapa adicional de seleção foi aplicada para compor a amostra final de teste. O *script* de extração de *homepages* foi utilizado para separar os repositórios em dois grupos: aqueles com uma *homepage* configurada no GitHub e aqueles sem. Para os fins da análise de eficácia automatizada, apenas os repositórios que possuíam uma *homepage* diretamente acessível foram considerados. A justificativa para esta decisão metodológica reside na necessidade de um ponto de entrada (URL) padronizado e confiável para a execução das ferramentas de avaliação (AXE, WAVE e Lighthouse). Repositórios

[14]GraphQL: linguagem de consulta para APIs desenvolvida pelo Meta.

sem uma *homepage* exigiriam uma configuração manual e execução local de seus projetos, um processo que introduziria uma alta variabilidade e não seria escalável para uma análise automatizada em larga escala, comprometendo a consistência e a comparabilidade dos resultados.

4.1. Procedimentos e Métricas

No que diz respeito aos métodos utilizados, a coleta de dados envolveu a identificação dos repositórios por meio da API do GitHub com GraphQL, seguida da inspeção automatizada dos arquivos de configuração de CI/CD presentes nos projetos. Com base nos trabalhos relacionados, que estabelecem um conjunto de ferramentas proeminentes na avaliação de acessibilidade, o objetivo foi detectar a integração das seguintes soluções: AXE, Pa11y, WAVE, Lighthouse, Asqatasun, HTML CodeSniffer e AChecker. A identificação ocorreu por meio da busca de termos associados a essas ferramentas em arquivos de dependência e em *workflows* de integração contínua. Em um subconjunto dos projetos selecionados, foram executadas essas ferramentas de forma controlada para avaliar a cobertura automática dos critérios de sucesso da WCAG 2.1. Os dados obtidos são submetidos a análises descritivas, como frequências, médias e medianas, além de análises inferenciais, incluindo correlação de Spearman, para investigar relações entre variáveis como frequência de uso das ferramentas, número de violações detectadas, níveis de conformidade atingidos e características dos projetos, como tamanho, maturidade e linguagem utilizada.

Para fundamentar a análise, este estudo se apoia nos resultados de Abu Doush et al. (2023) , que mapearam o potencial e os limites da automação na verificação dos critérios da WCAG 2.1. O principal resultado do trabalho deles, que indica que apenas 44% de todos os critérios podem ser totalmente automatizados com tecnologias padrão, é utilizado neste estudo como um *benchmark* teórico. Este *benchmark* servirá de base para contextualizar e avaliar a significância da cobertura real oferecida pelas ferramentas populares do ecossistema *open-source* em todos os três níveis de conformidade (A, AA e AAA).

Também foi empregada a métrica *Coverage Error Ratio* (CER), conforme proposta por [Alsaeedi 2020], para comparar objetivamente a abrangência das ferramentas analisadas.

$$CER(t_x, w) = \frac{de(t_x, w)}{\bigcup_{t \in T} de(t, w)} \tag{1}$$

A equação 1 define o CER, sendo utilizada para medir a capacidade de uma ferramenta automatizada de avaliação de acessibilidade de identificar erros em uma página web, em comparação ao total de erros distintos encontrados por todas as ferramentas analisadas. Na fórmula, o termo $(CER(t_x, w))$ representa o valor do *Coverage Error Ratio* para a ferramenta (t_x) aplicada à página (w). O numerador, $(de(t_x, w))$, corresponde ao número de erros de acessibilidade detectados especificamente pela ferramenta (t_x) naquela página. Já o denominador, $(\bigcup_{t \in T} de(t, w))$, indica o total de erros distintos obtidos pela união dos resultados de todas as ferramentas do conjunto (T) para a mesma página (w). Dessa forma, o CER quantifica a fração dos problemas de acessibilidade que uma ferramenta é capaz de identificar, considerando como referência o universo de erros reportados por todas as ferramentas avaliadas. Valores mais próximos de 1 indicam maior

abrangência da ferramenta, enquanto valores menores sugerem cobertura limitada frente ao conjunto analisado. A combinação desses métodos permite uma avaliação objetiva, escalável e reprodutível do problema investigado.

Além disso, foram adotadas as seguintes métricas de avaliação neste estudo: (i) quantidade de erros de acessibilidade detectados por uma ferramenta específica em um determinado projeto, métrica que representa o desempenho individual da ferramenta e corresponde ao numerador da fórmula do CER; (ii) conjunto de erros distintos detectados por todas as ferramentas analisadas em uma mesma página, o qual constitui o denominador da fórmula do CER e fornece a base de comparação para mensurar a abrangência relativa de cada ferramenta; (iii) frequência de adoção, determinada pela contagem de repositórios que apresentam ferramentas de acessibilidade configuradas em seus *pipelines* de CI/CD; e (iv) correlação entre a adoção das ferramentas e o nível de conformidade observado, analisada por meio do cálculo dos coeficientes de correlação de Spearman, conforme a adequação estatística dos dados.

4.2. Instrumentos

Para a realização deste estudo, adotam-se os seguintes instrumentos, com suas devidas justificativas:

1. Um *crawler* customizado em JavaScript, implementado sobre a API GraphQL do GitHub, para extração sistemática de metadados e arquivos de configuração de CI/CD. A escolha do GraphQL assegura consultas precisas e econômicas em termos de chamadas; o uso de JavaScript permite integração nativa com bibliotecas Node.js e facilita a manipulação assíncrona de grandes volumes de dados.
2. *Scripts* de preparação de dados foram utilizados para processar a lista de repositórios. Um *script* foi responsável por extrair as `homepages` via API do GitHub, separando os projetos com URLs testáveis. Outro conjunto de *scripts* teve a função de gerar os mapeamentos de equivalência, processando a documentação das ferramentas para criar arquivos que associam cada regra de teste a um critério WCAG.
3. Um orquestrador de testes automatizados foi o instrumento central para a geração dos dados de eficácia. Sua função foi iterar sobre a lista de repositórios que possuem uma *homepage* configurada. Para cada repositório, o orquestrador utilizou um *sitemapper* para identificar URLs adicionais dentro do domínio (limitado a um máximo de 10 URLs), garantindo uma cobertura mais ampla do que apenas a página inicial. Sobre este conjunto de páginas, ele executou sistematicamente as ferramentas de avaliação (AXE Core, Lighthouse CI e a API da WAVE), consolidando os resultados de violações em seus respectivos arquivos de saída no formato JSON.
4. Módulo em JavaScript para cálculo do CER. A presença desse módulo promove um fluxo de dados contínuo, reduzindo pontos de falha.
5. Bibliotecas estatísticas em JavaScript, como *simple-statistics*, para análises descritivas (média, mediana, desvio-padrão) e inferenciais (coeficientes de correlação). O uso de uma biblioteca leve e amplamente testada fornece confiança na precisão dos resultados estatísticos e simplifica a produção de relatórios programáticos.

A adoção exclusiva de JavaScript justifica-se pela ampla presença em ecossistemas Web, pela vasta disponibilidade de bibliotecas específicas para extração, processamento e

visualização de dados.

5. Resultados

Foram analisados 19.488 repositórios, dos quais 370 apresentaram pelo menos uma ferramenta de acessibilidade configurada em seus *pipelines* ou dependências, resultando em uma taxa de sucesso de 1,90%. Após a coleta inicial, foi realizada uma filtragem manual para remover falsos positivos, como bibliotecas, *frameworks*, *templates* e outros artefatos que, embora contenham referências a ferramentas de acessibilidade, não configuram uso efetivo em aplicações web com *pipelines* de integração contínua (CI/CD) ativos.

A filtragem reduziu o conjunto para 237 repositórios, correspondendo a 1,22% do total inicial. Esse resultado sugere que a presença de ferramentas automatizadas de acessibilidade ainda é baixa no ecossistema *open-source*, mesmo entre projetos populares; por exemplo, o repositório `excalidraw/excalidraw` [15], uma aplicação web com mais de 110.000 estrelas, não apresentou evidências da integração das principais ferramentas de acessibilidade. Todas as análises apresentadas a seguir consideram exclusivamente os resultados pós-filtragem.

Considerando os 237 artigos que compõe a base analisada, a ferramenta AXE foi identificada em 229 repositórios, correspondendo a 96,62% da amostra. O Lighthouse foi encontrado em 14 repositórios (5,91%) e o WAVE em 10 repositórios (4,22%). A ferramenta HTML CodeSniffer foi identificada em um repositório (0,42%) e as ferramentas Pa11y, Asqatasun e AChecker não foram identificadas em nenhum repositório. É importante notar que, embora o HTML CodeSniffer tenha sido identificado em um repositório, ele foi desconsiderado das análises comparativas de desempenho, uma vez que sua última atualização oficial ocorreu há mais de três anos. A utilização de uma ferramenta desatualizada poderia comprometer a relevância e a validade dos resultados em relação aos padrões atuais da WCAG. A análise das combinações de ferramentas mostrou que 220 repositórios (92,83%) utilizam apenas uma ferramenta, sendo o uso isolado do AXE o mais frequente. Apenas 17 repositórios (7,%) empregam duas ferramentas, com 12 casos de AXE + Lighthouse e 5 casos de AXE + WAVE. Não foram observadas combinações envolvendo três ou mais ferramentas, o que indica que a integração simultânea de múltiplas soluções de avaliação automatizada de acessibilidade é pouco adotada.

A distribuição dos repositórios filtrados por faixa de estrelas é apresentada na Tabela 1. Observa-se que 144 repositórios (60,76%) possuem até 49 estrelas, 23 repositórios (9,70%) estão na faixa de 50 a 99 estrelas, 42 repositórios (17,72%) entre 100 e 499 estrelas, 11 repositórios (4,64%) entre 500 e 999 estrelas e 17 repositórios (7,17%) possuem 1000 ou mais estrelas. A média de estrelas da amostra filtrada foi de 736,19, com valor mínimo de 2 e máximo de 75.448. O AXE foi identificado em todas as faixas de popularidade, enquanto o WAVE e o Lighthouse apresentaram maior frequência relativa nos projetos com 1000 ou mais estrelas.

A caracterização dos dados indica que a adoção de ferramentas automatizadas de acessibilidade em *pipelines* de CI/CD no ecossistema *open-source* é limitada, mesmo entre projetos com maior número de estrelas. A concentração no uso de uma única ferramenta, especialmente o AXE, implica que a cobertura dos critérios da WCAG 2.1 permanece restrita, uma vez que nenhuma ferramenta, isoladamente, é capaz de verificar todos

[15] `https://github.com/excalidraw/excalidraw`

Ferramenta	0–49	50–99	100–499	500–999	1000+
AXE	142	23	41	10	13
WAVE	3	2	0	0	5
Lighthouse	3	1	4	2	4
HTML CodeSniffer	1	0	0	0	0

Tabela 1. Distribuição das ferramentas por faixa de estrelas nos repositórios filtrado.

os critérios de forma totalmente automatizada. A ausência de ferramentas como Pa11y e Asqatasun, citadas na literatura, evidencia uma diferença entre as soluções discutidas em estudos acadêmicos e aquelas efetivamente incorporadas nos projetos analisados.

A partir da análise dos dados extraídos, foram obtidos os seguintes resultados em relação aos objetivos específicos propostos:

5.1. Cobertura automática dos critérios de sucesso da WCAG 2.1

Esta seção quantifica a proporção dos 78 critérios de sucesso (CS) da WCAG 2.1 que são cobertos pelas ferramentas de avaliação automatizada selecionadas: AXE, WAVE e Lighthouse. A análise visa determinar não apenas a cobertura total combinada, mas também a abrangência individual de cada ferramenta.

A equivalência entre as regras internas de cada ferramenta e os critérios de sucesso da WCAG foi estabelecida através da análise programática dos arquivos de mapeamento fornecidos no escopo deste estudo. Cada um desses arquivos fornece uma ligação direta entre o identificador de uma regra específica da ferramenta e um ou mais critérios WCAG, permitindo uma contagem precisa e objetiva da cobertura.

A análise da união dos critérios cobertos pelas três ferramentas revelou que, em conjunto, foi possível verificar automaticamente 58 dos 78 critérios de sucesso (74,4%) da WCAG 2.1. Este resultado é significativo quando contextualizado com o *benchmark* teórico estabelecido por Abu Doush et al. (2023), que indica que apenas 44% de todos os critérios WCAG podem ser totalmente automatizados com as tecnologias atuais. A discrepância sugere que a seleção de ferramentas modernas e populares, como as utilizadas neste estudo, oferece uma capacidade de verificação mais ampla do que a média teórica geral, destacando a importância da escolha da ferramenta correta.

A Tabela 2 detalha a cobertura individual de cada ferramenta, segmentada por nível de conformidade WCAG (A, AA e AAA). A ferramenta AXE se destaca com a maior cobertura, sendo capaz de verificar 51 critérios (65,4%) de forma isolada. A ferramenta WAVE segue com uma cobertura de 35 critérios (44,9%), enquanto o Lighthouse oferece a cobertura mais modesta, com 20 critérios (25,6%).

Feramenta	Nível A	Nível AA	Nível AAA	Total	Cobertura (%)
AXE	28	21	2	51	65,4%
WAVE	22	12	1	35	44,9%
Lighthouse	11	9	0	20	25,6%

Tabela 2. Nível de cobertura automática dos critérios WCAG 2.1 por ferramenta.

Os dados demonstram que, embora a combinação de ferramentas amplie significativamente a cobertura, nenhuma solução individual se aproxima de uma verificação completa. A liderança do AXE em termos de abrangência é clara, cobrindo a maior parte dos critérios de níveis A e AA. A quantificação da cobertura individual e conjunta, em vez de uma análise detalhada da sobreposição de critérios, é o foco deste objetivo. Isso permite estabelecer os limites e as capacidades do teste automatizado no contexto deste estudo.

5.2. Cobertura automatizada dos critérios e frequência de utilização das ferramentas

Esta seção compara a abrangência de cada ferramenta com sua frequência de uso no ecossistema *open-source*. A abrangência foi mensurada utilizando a métrica *Coverage Error Ratio*, que calcula a proporção de erros únicos que uma ferramenta detecta em relação ao total de erros distintos identificados por todas as ferramentas em conjunto. Um CER alto indica que a ferramenta é eficaz em encontrar problemas que outras podem não detectar.

A análise dos dados demonstra uma relação inversa entre a frequência de uso de uma ferramenta e sua capacidade de cobertura única. A ferramenta WAVE alcançou o maior CER médio (0,93), indicando que identifica 93% dos erros únicos no conjunto de dados. Contudo, sua frequência de uso é a mais baixa, sendo adotada por apenas 4,22% dos projetos.

Em contraste, a ferramenta AXE, com uma taxa de adoção de 96,62%, está posicionada com um CER médio de 0,23. O Lighthouse situa-se com baixa frequência de uso (5,91%) e o menor CER entre as três ferramentas (0,21).

Para quantificar a relação entre o desempenho das ferramentas, foi calculada a matriz de correlação de Spearman entre seus valores de CER. Os resultados mostraram uma correlação positiva (0,8192) entre AXE e Lighthouse, indicando que ambas tendem a detectar um conjunto similar de erros, o que sugere redundância. Por outro lado, a WAVE apresentou uma correlação negativa com AXE (-0,3597) e Lighthouse (-0,1580), o que demonstra um comportamento complementar, destacando-se na detecção de erros que as outras duas não cobrem.

Em conclusão, os dados indicam que a frequência de adoção de uma ferramenta não está diretamente correlacionada com sua capacidade de fornecer a cobertura mais ampla de erros únicos. A ferramenta com o maior CER (WAVE) é a menos utilizada, enquanto a ferramenta mais popular (AXE) possui uma cobertura única significativamente menor. Isso sugere que outros fatores, como facilidade de integração ou documentação, podem ser mais influentes na decisão de adoção.

Para aprofundar o entendimento sobre essa dinâmica, foi realizada uma análise da frequência dos 15 tipos de violação mais recorrentes em todo o conjunto de dados. Os resultados, apresentados na íntegra no Apêndice A, revelam que a maioria dos erros se concentra em um pequeno subconjunto de problemas. A Tabela 3, que resume os cinco critérios mais recorrentes, demonstra que apenas estes somam 2.283 ocorrências, representando 58% de todas as violações mais comuns identificadas no estudo.

Pos.	Critério WCAG	Total	AXE	Lighthouse	WAVE
1	4.1.2	566	160	155	251
2	1.4.3	516	182	143	191
3	2.4.4	453	114	98	241
4	1.1.1	387	67	49	271
5	1.3.1	361	23	27	311

Tabela 3. Os 5 Tipos de Erros de Acessibilidade Mais Frequentes.

A análise da Tabela 3 confirma o papel de cada ferramenta. O AXE demonstra alta eficácia na detecção desses erros de alto volume, o que ajuda a justificar sua popularidade e seu impacto positivo na redução do número geral de violações. Em contraste, a WAVE já demonstra seu comportamento complementar, sendo a principal ferramenta para detectar violações do critério `1.3.1 (Info and Relationships)`. A análise dos erros menos frequentes evidencia ainda mais o papel da WAVE como a única a identificar problemas em diversos outros critérios, o que explica seu alto valor de CER.

5.3. Adoção das ferramentas e a eficácia da acessibilidade

Foi realizada uma análise para avaliar se a adoção de uma ferramenta de acessibilidade está correlacionada com a eficácia da acessibilidade de um projeto, medida pela redução do número de violações. Para quantificar a relação e avaliar sua significância estatística, foi calculado o coeficiente de correlação de Spearman. A Tabela 4 detalha os coeficientes de correlação entre a adoção de cada ferramenta e o número total de violações.

Par de Variáveis	Correlação
Usa AXE vs. Total Violações Geral	-0.2006
Usa Lighthouse vs. Total Violações Geral	0.2071
Usa WAVE vs. Total Violações Geral	0.1344

Tabela 4. Correlação de Spearman entre a Adoção da Ferramenta e o Número Total de Violações.

Os resultados numéricos da Tabela 4 revelam dois padrões distintos. A correlação negativa para o AXE (-0.2006), embora fraca, suporta a hipótese de que seu uso está associado a uma melhoria na acessibilidade (menos erros). Por outro lado, as correlações positivas para Lighthouse (+0.2071) e WAVE (+0.1344) indicam um padrão contraintuitivo.

6. Discussão

A análise quantitativa da adoção e eficácia das ferramentas de acessibilidade no ecossistema *open-source* revela uma dinâmica complexa entre a popularidade de uma ferramenta e sua capacidade de detecção. O principal achado deste estudo é a relação inversa entre a frequência de uso de uma ferramenta e sua capacidade de cobertura única de erros, medida pelo CER. A ferramenta mais utilizada, AXE (96,62% de adoção), apresentou um dos menores valores de CER (0,23), enquanto a menos utilizada, WAVE (4,22% de adoção), obteve o maior CER (0,93).

A explicação para este fenômeno pode ser a combinação de dois fatores: a natureza dos erros detectados e a facilidade de implementação. A adoção do AXE está correlacionada a uma redução no número de violações (-0.2006), o que sugere que ele é eficaz em identificar os erros de acessibilidade mais comuns. Esta hipótese é confirmada pela análise de frequência das violações, que demonstra que um pequeno número de problemas, como falhas de contraste e links sem nome, é responsável pela maioria dos erros. A eficácia do AXE reside precisamente em sua capacidade de detectar esses erros de alto volume. Sua vasta popularidade, no entanto, pode ser melhor explicada pela sua facilidade de integração em processos de CI/CD. Conforme discutido por Sane (2021)[Sane 2021], a disponibilidade de *templates* e ações prontas para pipelines de automação é um fator decisivo para a adoção de ferramentas em ambientes de desenvolvimento ágil. A simplicidade de adicionar o AXE a um *workflow* com poucas linhas de configuração o torna a escolha pratica para equipes que buscam um retorno rápido com mínimo esforço.

Por outro lado, a correlação positiva observada para WAVE (+0.1344) e Lighthouse (+0.2071) não indica que essas ferramentas sejam prejudiciais. O alto CER da WAVE, que por sua vez é justificado por sua capacidade de ser a única ferramenta a detectar violações em critérios como `2.4.6 (Headings and Labels)` e `2.4.1 (Bypass Blocks)`, conforme detalhado no Apêndice A. Isso indica que ela é mais rigorosa em encontrar erros únicos que o AXE não detecta.

As implicações práticas destes achados são diretas. A dependência exclusiva da ferramenta mais popular, embora conveniente, pode levar a uma falsa sensação de segurança, deixando vulnerabilidades de acessibilidade não detectadas. A estratégia mais eficaz seria uma abordagem em duas camadas: utilizar uma ferramenta de alta popularidade, como o AXE, para *feedback* contínuo sobre erros comuns, e complementar essa prática com auditorias periódicas utilizando uma ferramenta de alta cobertura única, como a WAVE.

Essa abordagem em duas camadas é uma contribuição metodológica que sugere um roteiro para aprimorar as práticas de garantia de qualidade em acessibilidade. Ao tomar decisões mais informadas sobre ferramentas, as equipes podem otimizar seus processos, equilibrando velocidade com uma cobertura de testes robusta e, em última análise, capacitando-as a construir um ecossistema digital mais acessível de forma eficiente.

7. Ameaças à validade

A pesquisa realizada possui algumas limitações que podem afetar a validade dos resultados, as quais são apresentadas a seguir:

A análise se concentrou em repositórios *open-source* do GitHub, o que pode não refletir completamente a realidade de outros ecossistemas de desenvolvimento. Além disso, o estudo focou em um conjunto específico de três ferramentas (AXE, WAVE e Lighthouse), e a dinâmica de complementaridade poderia ser diferente se outras ferramentas fossem incluídas na análise.

A análise de correlação identifica associações, mas não pode provar uma relação de causa e efeito. A correlação positiva observada entre a adoção de WAVE e Lighthouse e o número de violações, por exemplo, é mais provavelmente um artefato de um viés de seleção, onde projetos mais complexos ou maduros são mais propensos a adotar essas ferramentas, do que um efeito causal das ferramentas em si.

As métricas utilizadas neste estudo são aproximações dos conceitos reais. A eficácia da acessibilidade foi quantificada pela contagem total de violações, um método que não diferencia um erro crítico de dezenas de erros menores. Da mesma forma, a adoção de uma ferramenta foi identificada pela presença de seus arquivos de configuração, o que não garante que os desenvolvedores de fato utilizem os relatórios gerados por ela.

8. Conclusão

A principal contribuição deste estudo é a demonstração empírica do paradoxo entre a popularidade de uma ferramenta e sua capacidade de cobertura única de erros. Foi constatado que a ferramenta mais utilizada pela comunidade, AXE, não é a que possui a maior abrangência diagnóstica, uma característica da ferramenta WAVE, que por sua vez é a menos adotada.

A interpretação destes achados sugere que a escolha de ferramentas no ecossistema *open-source* é impulsionada mais pela conveniência do que pela busca da cobertura de testes mais completa. A facilidade de integração em *pipelines* de CI/CD parece ser um fator decisivo, tornando ferramentas como o AXE a escolha mais prática para a detecção dos erros mais comuns. Este comportamento, no entanto, pode levar a uma falsa sensação de segurança, deixando vulnerabilidades de acessibilidade não detectadas.

Portanto, a conclusão central deste trabalho é que a estratégia mais eficaz para as equipes de desenvolvimento não é a adoção de uma única ferramenta, mas sim uma abordagem combinada que aproveite a conveniência das ferramentas populares para *feedback* contínuo e a capacidade de cobertura das ferramentas menos utilizadas para auditorias mais profundas. A divulgação destes resultados pode auxiliar as comunidades de desenvolvimento a tomar decisões mais informadas, promovendo práticas de teste mais robustas e contribuindo para a criação de um ecossistema digital mais acessível.

A partir das limitações e hipóteses levantadas na discussão, surgem diversas oportunidades para trabalhos futuros. Uma investigação qualitativa, por meio de entrevistas ou questionários com desenvolvedores, poderia validar as hipóteses sobre as motivações para a adoção de ferramentas. Tal estudo poderia confirmar se fatores como a facilidade de integração são mais decisivos do que a abrangência da cobertura de critérios. Adicionalmente, a generalização destes achados pode ser testada por meio da replicação desta metodologia em outros contextos de desenvolvimento, como projetos de código fechado.

Finalmente, a evolução mais significativa deste trabalho seria a realização de um mapeamento mais granular da cobertura de problemas dentro de cada critério WCAG. Enquanto a presente análise operou no nível do critério de sucesso (ex: 1.1.1), um trabalho futuro poderia decompor cada critério em seus contextos de aplicação específicos (Controles, Mídia, CAPTCHA, etc.). Isso permitiria uma avaliação mais precisa da capacidade de cada ferramenta, identificando exatamente quais tipos de problemas são cobertos e quais são negligenciados.

9. Pacote de Replicação

O pacote de replicação deste trabalho encontra-se disponível em: `https://github.com/ICEI-PUC-Minas-PPLES-TI/plf-es-2025-1-tcci-0393100-pes-gabriel-estevao-e-luana-fleury`

Referências

[Abu Doush et al. 2023] Abu Doush, I., Sultan, K., Al-Betar, M. A., Almeraj, Z., Alyasseri, Z. A. A., and Awadallah, M. A. (2023). Web accessibility automatic evaluation tools: to what extent can they be automated? *CCF Transactions on Pervasive Computing and Interaction*, 5:288–320.

[Alsaeedi 2020] Alsaeedi, A. (2020). Comparing web accessibility evaluation tools and evaluating the accessibility of webpages: Proposed frameworks. *Information*, 11:40.

[Ara and Sik-Lanyi 2025] Ara, J. and Sik-Lanyi, C. (2025). Automated evaluation of accessibility issues of webpage content: tool and evaluation. *Scientific Reports*, 15(1):9516.

[Cánovas Izquierdo et al. 2017] Cánovas Izquierdo, J. L., Cosentino, V., and Cabot, J. (2017). An empirical study on the maturity of the eclipse modeling ecosystem. In *2017 ACM/IEEE 20th International Conference on Model Driven Engineering Languages and Systems (MODELS)*, pages 292–302.

[Dias et al. 2022] Dias, J., Carvalho, D., Rocha, T., and Barroso, J. (2022). Automated evaluation tools for web and mobile accessibility: proposal of a new adaptive interface tool. *Procedia Computer Science*, 204:297–304.

[Kelasidou and Pettersson 2025] Kelasidou, S. and Pettersson, N. (2025). Assessing evaluation tools through accessibility standards in web development: With ci/cd pipelines.

[Macakoglu and Peker 2022] Macakoglu, S. and Peker, S. (2022). Web accessibility performance analysis using web content accessibility guidelines and automated tools: a systematic literature review. In *2022 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*, pages 1–8.

[Moser and Yli-Hukka Högback 2024] Moser, E. V. and Yli-Hukka Högback, J. (2024). Incorporating web accessibility into a ci/cd pipeline: A study of open source evaluation tools.

[Sane 2021] Sane, P. (2021). A brief survey of current software engineering practices in continuous integration and automated accessibility testing. In *2021 Sixth International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, pages 130–134.

[Velleman et al. 2017] Velleman, E., Nahuis, I., and Geest, T. (2017). Factors explaining adoption and implementation processes for web accessibility standards within egovernment systems and organizations. *Universal Access in the Information Society*, 16.

Apêndice

A. Frequência de Violações por Critério WCAG

A Tabela 5 apresenta o *ranking* das 15 violações de acessibilidade mais frequentes encontradas no estudo, detalhando o número total de ocorrências e a contribuição de cada uma das três ferramentas de avaliação para a detecção de cada tipo de erro.

Pos.	Critério WCAG	Total	AXE	Lighthouse	WAVE
1	4.1.2	566	160	155	251
2	1.4.3	516	182	143	191
3	2.4.4	453	114	98	241
4	1.1.1	387	67	49	271
5	1.3.1	361	23	27	311
6	2.4.6	311	0	0	311
7	2.4.1	311	0	0	311
8	3.1.2	291	0	0	291
9	2.5.3	233	0	42	191
10	2.1.1	193	14	0	179
11	3.3.2	129	0	0	129
12	3.3.1	92	0	0	92
13	4.1.3	92	0	0	92
14	1.4.1	51	27	24	0
15	3.1.1	51	19	12	20

Tabela 5. Ranking dos 15 Erros Mais Frequentes e Contribuição por Ferramenta.