Ferramenta de Rastreabilidade de Teste Baseada em Features para Software Altamente Configurável

Patrick Fernandes da Conceição
Faculdade Biopark
Toledo, Brasil
patrickfernandesconceicao@gmail.com

Willian Douglas Ferrari Mendonça
Faculdade Biopark
Toledo, Brasil
williandouglasferrari@gmail.com

***Abstract*—This study describes Test2Feature, a web tool for traceability of tests in Highly Configurable Systems (HCS). The project aims to assist software engineers in mapping and organizing tests through features like advanced customization, execution in specific patterns, and test case cataloging. With an intuitive interface and robust capabilities, the tool optimizes workflow and enhances accuracy in test execution. The detailed analysis and reporting features provide insights into test coverage and software performance. Test2Feature integrates with existing tools and supports multiple development environments, promoting a strategic approach aligned with project goals.**

***Keywords*—Highly Configurable System; Test Management; Customization.**

***Resumo*—Este estudo descreve a Test2Feature, uma ferramenta web para rastreabilidade de testes em Sistemas Altamente Configuráveis (SAC). O projeto visa auxiliar engenheiros de software no mapeamento e organização de testes através de funcionalidades como personalização avançada, execução em padrões específicos e catalogação de casos de teste. Com interface intuitiva e recursos robustos, a ferramenta otimiza o fluxo de trabalho e melhora a precisão na execução de testes. Os recursos de análise e relatórios detalhados fornecem *insights* sobre cobertura dos testes e desempenho do software. A Test2Feature integra-se com ferramentas existentes e suporta múltiplos ambientes de desenvolvimento, promovendo uma abordagem estratégica alinhada aos objetivos do projeto.**

***Palavras-chave*—Sistema Altamente Configurável; Gerenciamento de Testes; Customização.**

I. Introdução

Este artigo apresenta a evolução da Test2Feature, uma ferramenta web, que será disponibilizada por meio de um site, para rastreabilidade de testes em Sistemas Altamente Configuráveis (SAC). Os SACs proporcionam soluções flexíveis e adaptáveis para enfrentar desafios complexos em diversas áreas. Esses sistemas são configurados de diferentes maneiras, permitindo a criação de produtos personalizados que atendem a necessidades específicas. No entanto, garantir a qualidade desses sistemas é um desafio significativo. O ideal seria testar todas as configurações possíveis, mas isso é inviável na prática, pois o número de combinações cresce exponencialmente conforme o número de *features* (tradução em inglês da palavra características) aumenta. Isso complica e encarece o processo de teste, já que diversas *features* precisam ser avaliadas, e muitos casos de teste acabam se sobrepondo. Por exemplo, em testes de regressão, é necessário adotar abordagens que levem em conta as mudanças no SAC ao longo do tempo, à medida que *features* são adicionadas, modificadas ou removidas [1].

Para enfrentar esses desafios, métodos e ferramentas que permitem rastrear os testes são essenciais. Tufail et al [2], desenvolveram uma ferramenta que possibilita a rastreabilidade de casos de teste em relação a um conjunto de requisitos, sem a necessidade de executar o programa ou depender da cobertura de teste, que pode ser custosa. Contudo, essas ferramentas não são projetadas para SACs. Um levantamento recente de ferramentas de teste para SACs mostrou que poucas abordam a rastreabilidade de casos de teste em relação às *features*. As ferramentas que existem geralmente se baseiam em *Feature Models* (FM). Isso pode ser problemático, pois muitos SACs não possuem um FM disponível ou atualizado. As ferramentas atuais que utilizam o código-fonte de SACs não conseguem gerar rastreabilidade diretamente para as *features*, apenas associam casos de teste a trechos de código.

Diante desses desafios, este artigo introduz a atualização da Test2Feature, uma ferramenta inovadora que, a partir do código-fonte anotado de um SAC e de um conjunto de casos de teste, gera relatórios que mapeiam os casos de teste para as *features* do sistema. A ferramenta é composta por três módulos principais, que produzem três resultados fundamentais: rastreabilidade das *features* para o código-fonte, rastreabilidade dos casos de teste para o código-fonte, e rastreabilidade das *features* para os testes.

O objetivo principal da atualização da ferramenta é aprimorar a apresentação dos resultados baseados nas *features* e disponibilizar a ferramenta através de uma interface web, permitindo que engenheiros de software possam acessar e utilizar a Test2Feature de forma prática e remota. Como objetivo

secundário, o projeto também contribui para o desenvolvimento acadêmico do aluno no processo de iniciação científica. Essa ferramenta, mapeia casos de teste para *features* a partir do código-fonte que foi costruído anteriormente. Com isso, engenheiros de *software* podem identificar de forma mais precisa as partes do código e as *features* associadas a cada caso de teste. Diferente das soluções existentes, o relatório gerado por essa ferramenta não só facilita tarefas como testes de regressão e gestão de *features*, mas também aprimora a evolução e manutenção dos SACs, elevando a qualidade e a clareza das apresentações.

II. Background

Nesta seção, serão abordados os desafios, a importância e a concepção do projeto.

A. Desafios na Garantia de Qualidade de Softwares Altamente Configuráveis

SAC são desenvolvidos para oferecer soluções adaptáveis e flexíveis, capazes de atender às demandas complexas do mundo real. Esses sistemas são projetados para serem configurados de várias formas, gerando diferentes *features* personalizadas para usuários e casos específicos. No entanto, a garantia de qualidade desses sistemas apresenta desafios consideráveis. Testar todas as possíveis configurações é ideal, mas frequentemente impraticável devido ao crescimento exponencial do número de combinações de *features* à medida que o software evolui. Como ilustrado na Figura 1, o formato CSV apresenta limitações significativas na visualização dos dados, dificultando a análise efetiva dos resultados dos testes e evidenciando a necessidade de uma solução mais robusta. Isso complica e encarece o processo de teste, especialmente quando se trata de testes de regressão, onde é necessário considerar as mudanças e evoluções do software ao longo do tempo.

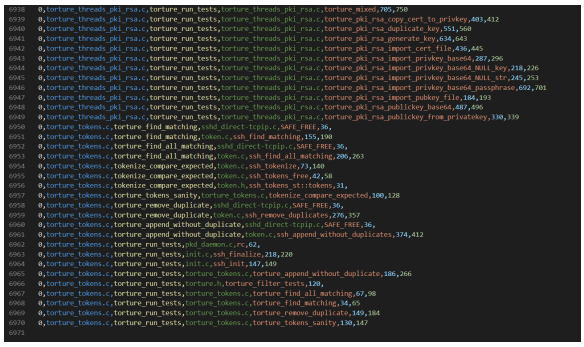


Fig. 1. MineTestLines - Tabela de Testes(Sem nenhum tratamento)

B. A Solução Proposta: Test2Feature

Para enfrentar esses desafios, o projeto propõe o desenvolvimento de uma ferramenta chamada Test2Feature. Esta ferramenta visa melhorar a rastreabilidade dos testes em SACs, vinculando casos de teste diretamente às *features* específicas do sistema. Diferente de ferramentas existentes, que muitas vezes se limitam a vincular testes às linhas de código ou dependem de *Feature Models* desatualizados, o Test2Feature oferece uma abordagem mais robusta. A ferramenta funciona a partir do código-fonte anotado de SACs e gera relatórios de rastreabilidade detalhados que mapeiam as *features* para as linhas de código e os casos de teste correspondentes. Essa inovação permite que desenvolvedores e testadores identifiquem rapidamente quais partes do código e quais *features* estão associadas a cada teste, facilitando processos como testes de regressão, gerenciamento de *features* e manutenção contínua da ferramenta.

C. Metodologia de Desenvolvimento da Ferramenta

O desenvolvimento da ferramenta Test2Feature utilizou uma arquitetura *web* com uma interface desenvolvida em *Flask* [3], *HTML* [4] e *CSS* [5], proporcionando uma experiência de usuário fluida e responsiva. A interface permite que os engenheiros de *software* insiram dados, configurem opções de rastreabilidade e visualizem os resultados em tempo real. Esse *front-end* comunica-se com o *back-end* para acionar funcionalidades de rastreamento e geração de relatórios.

O *back-end* foi implementado em *Python* [6], que executa a análise do código-fonte e organiza as informações de rastreabilidade em arquivos CSV. Esses arquivos documentam as relações entre funcionalidades e casos de teste e servem como base para os *dashboards* de análise da ferramenta. Os relatórios gerados simplificam a visualização de dados, facilitando a tomada de decisão e a manutenção contínua dos SACs.

D. Fluxo Operacional da Test2Feature

A Test2Feature é composta por três módulos, conforme ilustrado na Figura 2 . O *input* fornecido pelo engenheiro de software é realizado através de formulários presentes na ferramenta, onde são informados os caminhos para o repositório do sistema e para a pasta de testes, além do commit atual a ser analisado. Essas informações são enviadas para o módulo `MineFeaturesLines`, responsável por localizar as *features* nas linhas de código, e para o módulo `MineTestLines`, responsável por identificar as linhas de código que correspondem a cada caso de teste na pasta de testes. O módulo `MineTestLines` utiliza a ferramenta Doxygen para gerar arquivos XML que representam o grafo de dependência das funções no código C/C++. As informações desses dois módulos são então utilizadas pelo

módulo `MergeTestFeaturesLines` para gerar um arquivo CSV contendo a rastreabilidade das *features* aos casos de teste.

Esses arquivos CSV não apenas documentam a rastreabilidade, mas também servem como base para a criação dos *dashboards* da ferramenta, proporcionando visualizações analíticas para facilitar a tomada de decisão.

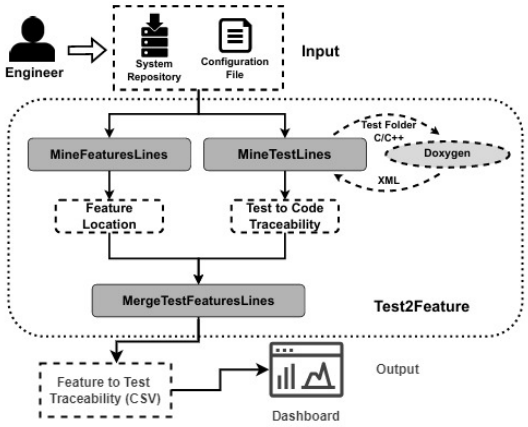


Fig. 2. Fluxo de Funcionamento da Ferramenta

III. Tecnologias Utilizadas

Segundo Pinto et al. [7], a escolha tecnológica é crucial para desenvolver ferramentas robustas e adaptáveis. Assim, a Test2Feature foi construída com tecnologias que asseguram flexibilidade, integração eficiente e desempenho otimizado.

A. JavaScript

JavaScript [8] foi utilizado no *front-end*, proporcionando flexibilidade e dinamismo para a criação de interfaces interativas e responsivas, essenciais para uma experiência de usuário fluida.

B. HTML e CSS

HTML [4] e *CSS* [5] estruturaram e estilizaram as páginas da aplicação, garantindo uma interface intuitiva e visualmente atraente com *design* responsivo.

C. Python e Flask

Python [6] e o *framework Flask* [3] foram utilizados no *backend*, pela simplicidade e escalabilidade, permitindo criar uma API robusta para comunicação entre sistemas.

D. Charts

Charts [9], uma biblioteca JavaScript para gráficos, facilitou a criação de visualizações dinâmicas, possibilitando uma análise intuitiva de dados e métricas de desempenho.

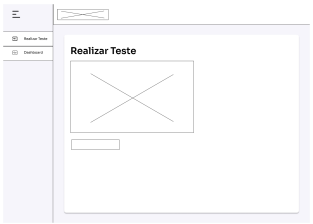
IV. Processo de Ideação e Prototipação

O desenvolvimento de uma ferramenta começa com uma ideia, mas transformar essa ideia em um produto funcional requer um processo estruturado de ideação e prototipagem.

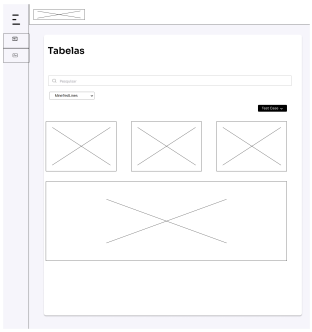
A. Ideação

Na fase inicial, a equipe se dedicou a sessões de *brainstorming*, discussões detalhadas e à análise de requisitos. O objetivo era compreender profundamente as necessidades de gerenciamento de testes da ferramenta, e identificar os principais desafios enfrentados pelos usuários. Várias ideias foram sugeridas, avaliadas e aprimoradas, resultando na criação de um esboço preliminar da ferramenta, priorizando a eficiência e a precisão no gerenciamento de testes ao longo do tempo e espaço.

Foram criados protótipos descartáveis de telas tanto para a etapa de ideação quanto para a prototipação. O primeiro exemplo apresentado foi a Figura 3a, que exibe a tela inicial da ferramenta utilizada para a realização dos testes e geração das tabelas baseadas nos endereços dados pelo usuário. Outro exemplo da fase de ideação é apresentado na Figura 3b, que ilustra as planilhas de testes e os gráficos gerados. Todos os esboços e protótipos foram criados utilizando o Figma [10], um *software de design* gráfico vetorial e prototipagem de projetos.



(a) Protótipo Descartável do Menu Inicial da Aplicação



(b) Protótipo Descartável dos gráficos de testes

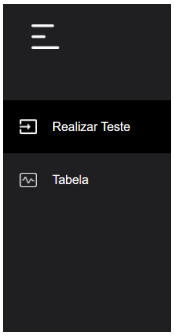
Fig. 3. Protótipos Descartáveis

B. Prototipação

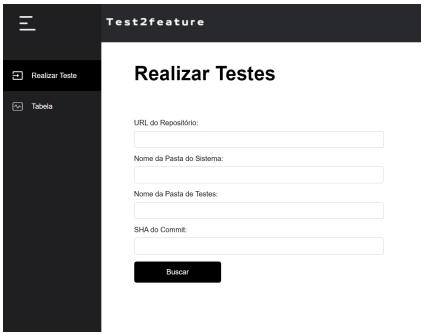
Nesta etapa, a equipe avançou para a prototipação, criando *wireframes* e *mockups* do aplicativo. Esses protótipos representaram a interface do usuário e a interação entre as diferentes *features*, servindo como um guia visual. Eles permitiram à equipe visualizar o fluxo do aplicativo e identificar possíveis áreas de melhoria. Além disso, foram fundamentais para coletar *feedback* inicial, tanto da equipe quanto de potenciais usuários,

garantindo que o *design* fosse intuitivo e adequado às necessidades do público-alvo.

Como complemento à fase de ideação, desenvolvemos os protótipos do aplicativo. A Figura 4a ilustra o menu da tela inicial do aplicativo, destacando suas principais funcionalidades, enquanto a Figura 4b Realiza o processo de construção de uma nova tabela de testes.

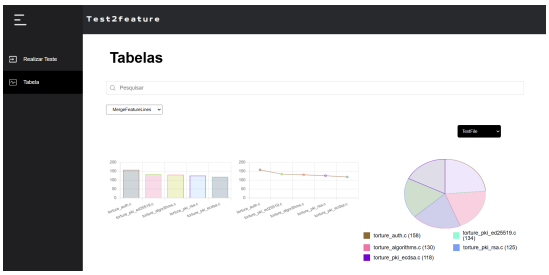


(a) Protótipo do Menu Inicial da Aplicação

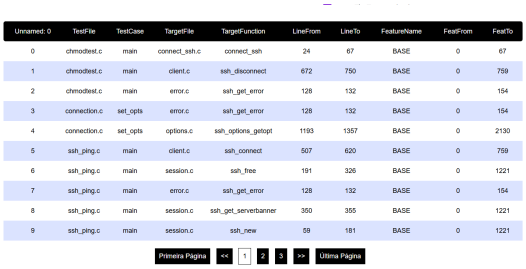


(b) Protótipo Realizar Testes

Fig. 4. Protótipos Válidados



(a) Protótipo dos gráficos de testes



(b) Protótipo para Visualizar as Tabelas

Fig. 5. Protótipos Válidados

V. Resultados Parciais

A Test2Feature tem demonstrado resultados promissores durante os testes iniciais, especialmente na melhoria da organização e rastreabilidade de casos de teste em SACs.

Concluímos o desenvolvimento parcial da interface do usuário, cujas principais funcionalidades podem ser observadas nas figuras a seguir. A Figura 4b mostra a tela inicial da ferramenta, enquanto as Figuras 5b e 5a demonstram a visualização das planilhas e gráficos.

Os testes iniciais retornaram *feedback* positivo, destacando a eficácia das planilhas e gráficos gerados no controle e acompanhamento dos testes. Estes recursos facilitaram tanto a visualização dos dados quanto a identificação de áreas que necessitavam de ajustes, contribuindo para uma gestão mais eficiente do processo de testes.

VI. Conclusão

A Test2Feature destaca-se como uma solução robusta para superar as limitações das ferramentas existentes, que frequentemente restringem-se a ambientes específicos como código Java ou não fornecem rastreabilidade adequada. A ferramenta possibilita que engenheiros de software identifiquem facilmente quais *features* estão associadas a cada caso de teste, contribuindo para uma gestão mais eficiente dos testes em SACs.

Como trabalho futuro, planejamos disponibilizar a ferramenta na web, o que exigirá ajustes para manipulação dinâmica das tabelas geradas em tempo de execução, superando a atual limitação de uso exclusivamente em ambiente local. Além disso, pretendemos realizar uma análise de resultados utilizando sistemas de nível industrial open-source, que passam por atualizações diárias.

Referências

[1] W. D. Mendonça, S. R. Vergilio, G. K. Michelon, A. Egyed, and W. K. Assunção, "Test2feature: Feature-based test traceability tool for highly configurable software," in *Proceedings of the 26th ACM International Systems and Software Product Line Conference-Volume B*, 2022, pp. 62–65.

[2] H. Tufail, M. F. Masood, B. Zeb, F. Azam, and M. W. Anwar, "A systematic review of requirement traceability techniques and tools," in *2017 2nd International Conference on System Reliability and Safety (ICSRS)*, 2017, pp. 450–454.

[3] Armin Ronacher, "Flask: Web Development, One Drop at a Time," 2024. [Online]. Available: https://flask.palletsprojects.com/

[4] World Wide Web Consortium (W3C), "HTML Living Standard," 2023. [Online]. Available: https://html.spec.whatwg.org/

[5] ——, "Cascading Style Sheets (CSS) Snapshot 2023," 2023. [Online]. Available: https://www.w3.org/TR/css-2023/

[6] Python Software Foundation, "Python Programming Language," 2024. [Online]. Available: https://www.python.org/

[7] F. Pinto, T. Oliveira, and M. Santos, "The use of agile methodologies in software development: A systematic review," *Journal of Systems and Software*, vol. 162, pp. 110–123, 2020.

[8] Mozilla, "Javascript - mdn web docs," 2024. [Online]. Available: https://developer.mozilla.org/pt-BR/docs/Web/JavaScript

[9] Chart.js Contributors, "Chart.js: Simple HTML5 Charts," 2024. [Online]. Available: https://www.chartjs.org/

[10] Figma, "Figma: The collaborative interface design tool," 2024. [Online]. Available: https://www.figma.com/