1. Introdução

A atividade de teste de software continua sendo um dos pilares fundamentais para garantir a qualidade de sistemas complexos. Em particular, Testes manuais continuam desempenhando um papel central na garantia da qualidade de software, especialmente em cenários onde a automação é inviável ou insuficiente (Myers et al., 2011)., custosa ou insuficiente para capturar aspectos subjetivos da experiência do usuário, como usabilidade, percepção visual e comportamento dinâmico de interfaces. No entanto, a execução manual de suítes extensas de testes impõe alto custo cognitivo e operacional aos testadores, especialmente quando os casos de teste apresentam dependências de estado e passos parcialmente sobrepostos e também quando se trata de testadores iniciantes ou menos experientes.

Na prática, testadores experientes frequentemente adotam atalhos informais durante a execução de uma suíte de testes. Casos de teste relacionados a uma mesma funcionalidade tendem a compartilhar etapas iniciais semelhantes, permitindo que determinadas ações sejam executadas uma única vez e reaproveitadas para validar múltiplos cenários subsequentes. Esse comportamento humano, embora eficiente, não é explicitamente modelado ou explorado por técnicas tradicionais de priorização de casos de teste, que normalmente assumem a execução independente de cada teste a partir de um estado inicial.

Além disso, nem todas as ações presentes em um caso de teste possuem o mesmo impacto sobre o estado do sistema. Algumas ações são não destrutivas, como verificações visuais ou checagens de elementos da interface, enquanto outras são destrutivas ou transformadoras, alterando o estado da aplicação de forma irreversível ou parcialmente reversível. Ignorar essas diferenças pode levar a ordenações com quebras de estado, com isso verificações importantes são inviabilizadas por ações executadas prematuramente, aumentando a necessidade de reinicializações do sistema e, consequentemente, o esforço e retrabalho do testador.

Técnicas clássicas de priorização de casos de teste visam maximizar a detecção precoce de falhas, sendo amplamente estudadas no contexto de testes automatizados e regressão de software (Rothermel et al., 2001; Elbaum et al., 2002; Yoo & Harman, 2012). A literatura sobre priorização de casos de teste concentra-se majoritariamente em testes automatizados, especialmente em cenários de regressão de software (Yoo & Harman, 2012; Khatibsyarbini et al., 2018). Nesse contexto, este trabalho propõe uma abordagem baseada em sistemas de recomendação interativos para a ordenação adaptativa de casos de teste manuais. A ideia central é tratar a ordenação não como uma decisão fixa, mas como um processo dinâmico, no qual o sistema sugere sequências de execução que minimizam o esforço humano, respeitam dependências de estado e aprendem continuamente a partir do comportamento e feedback do testador. Dessa forma, busca-se capturar o conhecimento tácito de testadores experientes e incorporá-lo progressivamente ao processo de recomendação.

O objetivo deste trabalho é propor e avaliar um sistema de recomendação human-in-the-loop capaz de ordenar casos de teste manuais de forma adaptativa, considerando o impacto das ações no estado do sistema e explorando feedback explícito e implícito do testador e aprender com esse feedback. Espera-se que a abordagem reduza redundâncias, evite execuções desnecessárias e contribua para uma execução mais eficiente e natural das suítes de teste, aproximando o processo automatizado da forma como testadores experientes atuam na prática.

Diferentemente de abordagens tradicionais de priorização de testes, que focam em testes automatizados e ordenações estáticas, este trabalho investiga a ordenação adaptativa de testes manuais sob a perspectiva de sistemas de recomendação human-in-the-loop.

2. Objetivos e Hipóteses de Pesquisa

2.1 Objetivo Geral

O objetivo geral deste trabalho é propor e avaliar um sistema de recomendação interativo para a ordenação adaptativa de casos de teste manuais, capaz de minimizar o esforço do testador humano, respeitando dependências de estado entre testes e incorporando conhecimento tácito por meio de feedback explícito e implícito durante a execução da suíte.

2.2 Objetivos Específicos

Para atingir o objetivo geral, este trabalho busca:

1. Modelar casos de teste manuais como ações sensíveis a estado, distinguindo ações não destrutivas (verificações) de ações destrutivas ou transformadoras (mudanças de estado).
2. Representar dependências e transições de estado entre casos de teste por meio de grafos de estados abstratos, mitigando a complexidade de sistemas extensos por meio de níveis de granularidade ajustáveis.
3. Projetar um sistema de recomendação interativo (human-in-the-loop) que sugira ordenações de execução de forma dinâmica, adaptando-se em tempo real caso ocorram falhas de teste que impeçam a progressão do estado planejado.
4. Capturar feedback do testador, tanto implícito (ordem seguida, tempo de execução, desvios da recomendação) quanto explícito (avaliações ou preferências).
5. Avaliar experimentalmente a abordagem proposta quanto à redução de esforço (medido por tempo e número de passos repetidos), redundância e necessidade de reinicializações do sistema.

2.3 Hipóteses de Pesquisa

Com base nos objetivos definidos, este trabalho investiga as seguintes hipóteses:

H1 — Redução de Esforço

A ordenação adaptativa de casos de teste manuais baseada em recomendação interativa reduz o esforço total do testador quando comparada a ordenações estáticas tradicionais.

H2 — Preservação de Estado

Considerar explicitamente o impacto das ações no estado do sistema reduz a necessidade de reinicializações e reexecuções de passos redundantes.

H3 — Aprendizado a partir do Testador

O uso de feedback humano (explícito e implícito) permite que o sistema aprenda preferências de execução e gere ordenações progressivamente mais eficientes.

H4 — Adequação a Diferentes Perfis

Para avaliar a eficácia da abordagem proposta e validar as hipóteses de pesquisa (H1 a H4), será realizado um experimento controlado comparando a execução manual tradicional com o sistema de recomendação interativo.

3. Motivação da pesquisa

Apesar dos avanços nas técnicas de priorização de casos de teste, grande parte das abordagens existentes assume que os testes são executados de forma independente, iniciando sempre a partir de um estado conhecido e controlado do sistema. Essa suposição, embora válida em contextos de testes automatizados, não reflete fielmente a realidade da execução manual de testes, na qual o estado do sistema evolui continuamente ao longo da sessão de testes e decisões humanas influenciam diretamente a ordem de execução.

Estudos empíricos mostram que testadores frequentemente tomam decisões adaptativas durante a execução manual de testes, explorando o estado do sistema de forma incremental (Itkonen et al., 2009).. Em vez de executar cada caso de teste de forma isolada, eles identificam passos comuns entre cenários e organizam a execução de modo a reaproveitar estados intermediários sempre que possível. Esse comportamento é particularmente relevante em sistemas interativos, como aplicações de comunicação, onde ações sucessivas podem alterar o estado da aplicação de forma incremental.

Para ilustrar esse cenário, considere os três casos de teste apresentados a seguir, relacionados à funcionalidade de chamadas de voz e vídeo em uma aplicação de comunicação:

TESTE-001

Criar um contato

Iniciar uma chamada de voz com o contato

Verificar a presença do ícone de HD

TESTE-002

Criar um contato

Iniciar uma chamada de voz com o contato

Realizar o upgrade para uma chamada de vídeo

Verificar se o vídeo é exibido corretamente

TESTE-003

Criar um contato

Iniciar uma chamada de voz com o contato

Realizar o upgrade para uma chamada de vídeo

Realizar o downgrade para chamada de voz

Verificar se a chamada retorna ao modo de voz

Se esses casos fossem executados de forma estritamente independente, o testador precisaria repetir as ações iniciais de criação de contato e início de chamada para cada teste, incorrendo em redundância significativa de esforço. No entanto, um testador experiente tipicamente executaria os testes da seguinte forma:

1. Criar um contato
2. Iniciar uma chamada de voz
3. Verificar o ícone de HD (validando o TESTE-001)
4. Realizar o upgrade para chamada de vídeo
5. Verificar a exibição do vídeo (validando o TESTE-002)
6. Realizar o downgrade para chamada de voz
7. Verificar o retorno ao modo de voz (validando o TESTE-003)

Essa sequência permite validar os três casos de teste com menos passos repetidos, explorando o estado corrente do sistema. Contudo, esse tipo de otimização depende fortemente da experiência do testador e raramente é formalizado ou apoiado por ferramentas.

Além disso, observa-se que nem todas as ações possuem o mesmo impacto sobre o estado do sistema. A verificação da presença do ícone de HD, por exemplo, é uma ação não destrutiva, pois não altera o estado da aplicação. Em contrapartida, a ação de upgrade de uma chamada de voz para vídeo altera o estado do sistema, potencialmente inviabilizando verificações que dependem do estado anterior. Portanto, a ordem de execução das ações é crítica: realizar o upgrade antes da verificação do ícone de HD impediria a validação correta do TESTE-001 sem uma reinicialização do sistema.

Esse exemplo evidencia duas limitações centrais das abordagens tradicionais de priorização de testes:

(i) a ausência de uma modelagem explícita de dependências de estado entre ações de teste; e

(ii) a falta de mecanismos para capturar e reutilizar o conhecimento tácito empregado por testadores humanos durante a execução manual.

Diante desse cenário, surge a necessidade de uma abordagem que vá além da simples ordenação estática de casos de teste e que atue como um apoio à decisão para o testador humano. Um sistema capaz de modelar ações sensíveis a estado, sugerir sequências de execução que maximizem o reaproveitamento de estados intermediários e aprender progressivamente com o comportamento do testador tem o potencial de reduzir esforço, evitar reinicializações desnecessárias e tornar a execução manual de testes mais eficiente e sistemática.

4. Proposta do Sistema

Esta seção descreve a proposta de solução (Intelligent Assistant for the Reordering of Test Execution Sequences), um sistema de recomendação interativo destinado à ordenação adaptativa de casos de teste manuais, com o objetivo de reduzir o esforço do testador humano e minimizar a execução redundante de passos.

O sistema foi concebido para atuar como apoio à decisão, não realizando a execução automática dos testes, mas sugerindo ordenações de execução que respeitam dependências de estado, ações destrutivas e padrões de uso observados durante a interação com o testador.

4.1 Descrição dos Módulos da Arquitetura

A arquitetura do sistema proposto é composta por módulos independentes e interconectados, organizados de forma a permitir a modelagem explícita de estado, a

geração dinâmica de recomendações e a incorporação de feedback humano no processo de ordenação. A Figura 1 apresenta uma visão geral da arquitetura do sistema.

De forma resumida, o sistema recebe como entrada uma suíte de casos de teste manuais, modela os possíveis estados do sistema sob teste, gera uma ordenação sugerida com base no estado atual e interage continuamente com o testador humano, incorporando feedback explícito e implícito para aprimorar recomendações futuras.

A arquitetura é composta pelos seguintes módulos principais: (i) Módulo de Entrada de Casos de Teste, (ii) Módulo de Modelagem de Estado, (iii) Módulo de Recomendação, (iv) Módulo de Interação e Feedback (Human-in-the-Loop) e (v) Módulo de Aprendizado Adaptativo.

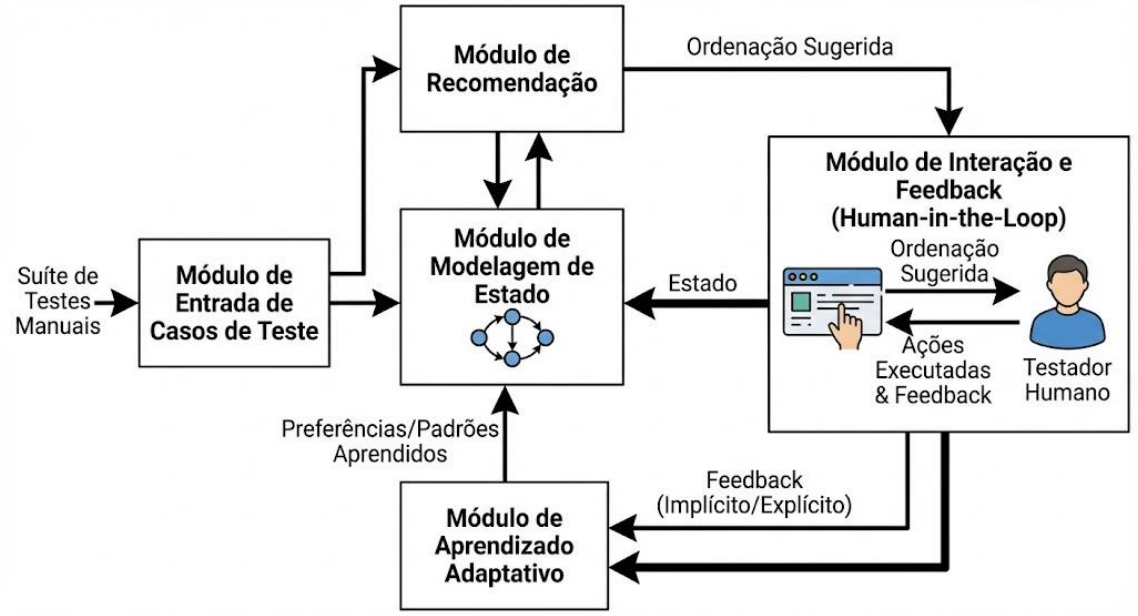

Figura 01

4.2 Modelo de Entrada

O Módulo de Entrada de Casos de Teste é responsável por receber e interpretar a suíte de testes manuais fornecida pelo usuário. Cada caso de teste é modelado como uma sequência ordenada de ações, podendo conter metadados associados, como identificadores, descrições e informações contextuais relevantes.

Nesse estágio, o sistema não assume nenhum formato específico de implementação, abstraindo os casos de teste como entidades conceituais que descrevem interações possíveis com o sistema sob teste. Essa abstração permite que a abordagem seja aplicada a diferentes domínios e ferramentas de teste manual.

4.3 Modelagem de Estado

A Modelagem de Estado constitui um dos componentes centrais do sistema. Nesse módulo, as ações descritas nos casos de teste são analisadas com o objetivo de identificar seus impactos sobre o estado do sistema sob teste.

As ações são classificadas em dois grupos principais:

(i) ações não destrutivas, que não alteram o estado do sistema, como verificações visuais ou validações de interface; e

(ii) ações destrutivas ou transformadoras, que provocam mudanças de estado, como transições de modo, criação ou remoção de entidades e alterações de configuração.

Essas relações são representadas por meio de uma estrutura baseada em grafos, na qual os nós correspondem a estados do sistema e as arestas representam transições causadas pela execução de ações. Essa representação permite capturar dependências entre testes e identificar sequências de execução que preservam o estado, reduzindo a necessidade de reinicializações.

4.4 Mecanismo de Recomendação

O Módulo de Recomendação é responsável por gerar uma ordenação sugerida dos casos de teste, considerando o estado atual do sistema, as transições possíveis e as ações ainda não executadas.

A ordenação não é estática, sendo recalculada dinamicamente à medida que o estado evolui durante a execução dos testes. O módulo prioriza sequências que maximizam o reaproveitamento de estado, executando primeiro ações não destrutivas sempre que possível e postergando ações que causam mudanças irreversíveis.

Esse mecanismo permite que o sistema se adapte ao progresso real da execução, oferecendo recomendações alinhadas com o contexto corrente da sessão de testes.

4.5 Interação Human-in-the-Loop e Feedback

O sistema adota uma abordagem human-in-the-loop, na qual o testador humano permanece no centro do processo de decisão. As ordenações sugeridas são apresentadas ao testador, que pode segui-las integralmente, executá-las parcialmente ou ignorá-las.

As ações efetivamente executadas pelo testador são monitoradas e utilizadas como fonte de feedback implícito, enquanto avaliações diretas ou ajustes manuais podem ser considerados como feedback explícito. Além disso, o estado atualizado do sistema sob teste é continuamente retornado ao módulo de modelagem de estado.

Essa interação contínua permite que o sistema aprenda com o comportamento real do testador, aproximando suas recomendações de estratégias utilizadas por testadores experientes.

4.6 Aprendizado Adaptativo

O módulo de aprendizado utiliza o feedback acumulado para atualizar o modelo de recomendação ao longo do tempo. À medida que mais dados são coletados, o modelo passa a refletir preferências individuais e padrões de eficiência observados. Além disso, o sistema deve demonstrar resiliência:

- Tratamento de Falhas: Caso o testador reporte uma falha em uma ação (feedback explícito), o sistema deve recalcular instantaneamente a recomendação para os testes restantes, priorizando caminhos que não dependam do estado corrompido ou sugerindo o reset necessário no momento mais oportuno.
- Evolução de Perfil: A abordagem permite que o sistema se adapte a diferentes níveis de experiência, fornecendo recomendações mais detalhadas para iniciantes e atalhos mais agressivos para especialistas.

4.7 Transparência das recomendações

Com o objetivo de aumentar a confiança do testador no sistema, as recomendações geradas são acompanhadas de explicações textuais que indicam os principais fatores que motivaram a ordenação sugerida. Essas explicações podem incluir a preservação de estado, a redução de redundância ou a priorização de ações de verificação.

A presença de mecanismos de explicabilidade contribui para a transparência do sistema e facilita sua adoção em ambientes de teste manuais.

5. Metodologia

Esta seção descreve a metodologia adotada para a geração da ordenação adaptativa de casos de teste manuais no sistema IARTES. A abordagem proposta combina heurísticas baseadas em estado com técnicas de aprendizado de máquina, organizadas em um pipeline híbrido que permite recomendações progressivamente mais eficientes à medida que o sistema interage com o testador humano.

A metodologia foi concebida para lidar com a natureza sequencial e dependente de estado dos testes manuais, bem como para incorporar conhecimento tácito por meio de feedback implícito e explícito.

5.1 Formulação do Problema

O problema abordado neste trabalho consiste em determinar uma ordenação de execução para uma suíte de casos de teste manuais que minimize o esforço do testador, considerando que:

- Os casos de teste compartilham passos e estados intermediários;
- Algumas ações alteram irreversivelmente o estado do sistema sob teste;
- O estado do sistema evolui ao longo da execução;
- O comportamento do testador pode divergir da recomendação sugerida.

Diferentemente de abordagens tradicionais de priorização, o problema não é tratado como uma simples ordenação estática, mas como um processo dinâmico de recomendação de sequências, sensível ao contexto e ao histórico de execução.

5.2 Representação de Features

Para permitir a aplicação de técnicas de aprendizado de máquina, cada caso de teste e cada possível ordenação são descritos por um conjunto de características (features) extraídas a partir da modelagem de estado e do histórico de execução.

As features utilizadas incluem, mas não se limitam a:

- Número de ações não destrutivas presentes no caso de teste;
- Impacto estimado da execução sobre o estado do sistema;
- Grau de reaproveitamento de estado em relação aos testes já executados;
- Frequência histórica de execução em contextos semelhantes;
- Aderência a preferências observadas do testador.

Essas características são utilizadas para capturar tanto aspectos estruturais dos casos de teste quanto padrões comportamentais emergentes durante o uso do sistema.

5.3 Abordagem de Aprendizado de Máquina

O aprendizado é formulado como um problema de regressão, no qual o modelo estima a qualidade relativa de um caso de teste (ou sequência de testes) em um determinado contexto de execução.

O modelo é treinado incrementalmente a partir de dados coletados durante sessões de teste, incluindo:

- Ordenações seguidas pelo testador;
- Tempo relativo de execução;
- Desvios em relação à recomendação sugerida;
- Feedback explícito, quando disponível.

Essa abordagem permite que o sistema se adapte progressivamente, refinando suas recomendações conforme observa novos padrões de uso.

5.4 Heurística Baseada em Estado

Antes da aplicação do modelo de aprendizado, o sistema emprega uma heurística baseada em estado para garantir a viabilidade da ordenação sugerida. Essa heurística atua como um mecanismo de filtragem inicial, assegurando que:

- Ações não destrutivas sejam priorizadas;
- Transições de estado incompatíveis sejam evitadas;
- Dependências explícitas entre casos de teste sejam respeitadas.

A heurística também permite a geração de uma ordenação válida mesmo em cenários nos quais o modelo de aprendizado ainda não possui dados suficientes, funcionando como um mecanismo de fallback.

5.5 Processo de Geração da Ordenação

A geração da ordenação segue um pipeline híbrido composto pelas seguintes etapas:

1. Identificação do estado atual do sistema sob teste.
2. Seleção dos casos de teste candidatos viáveis a partir da modelagem de estado.
3. Aplicação da heurística baseada em estado para gerar uma ordenação inicial.
4. Refinamento da ordenação por meio do modelo de aprendizado de máquina.
5. Ajustes locais na sequência, visando maximizar reaproveitamento de estado e minimizar ações destrutivas.

Esse processo é repetido dinamicamente à medida que o estado do sistema evolui durante a execução da suíte.

5.6 Reordenação Contextual e Garantia de Consistência

Durante a execução dos testes, podem ocorrer desvios em relação à ordenação sugerida, seja por decisão do testador ou por mudanças inesperadas no estado do sistema. Para lidar com esses cenários, o sistema realiza uma reordenação contextual baseada no estado atualizado.

Sempre que necessário, mecanismos de reparo lógico são aplicados para garantir a consistência da sequência resultante, utilizando técnicas como ordenação topológica sobre o grafo de estados, de forma a preservar dependências e evitar sequências inválidas.

5.7 Incorporação de Feedback Humano

O feedback humano é incorporado de duas formas:

- Feedback implícito, derivado das ações efetivamente executadas, da ordem seguida e do tempo relativo gasto em cada etapa;
- Feedback explícito, quando o testador fornece avaliações diretas sobre a utilidade da recomendação.

Esses sinais são utilizados tanto para atualizar o modelo de aprendizado quanto para ajustar os critérios heurísticos, permitindo um ciclo contínuo de adaptação.

5.8 Considerações sobre Reprodutibilidade

Com o objetivo de garantir a reprodutibilidade dos experimentos, o sistema mantém registros detalhados das sessões de teste, incluindo estados intermediários, ordenações sugeridas e decisões tomadas pelo testador. Esses registros permitem a reconstrução de cenários de execução e a avaliação controlada da abordagem proposta.

6. Discussão e Ameaças à Validade

Esta seção discute os principais resultados e características da abordagem proposta, bem como suas limitações e possíveis ameaças à validade do estudo. O objetivo é analisar criticamente o alcance da solução apresentada e delimitar o contexto no qual seus resultados podem ser interpretados.

6.1 Discussão dos Resultados e da Abordagem

A abordagem proposta neste trabalho endereça um problema recorrente em testes manuais: a execução redundante de passos e a perda de eficiência causada por ordenações estáticas que não consideram dependências de estado nem o conhecimento tácito do testador.

Ao tratar a ordenação de casos de teste como um processo dinâmico e sensível ao estado, o sistema aproxima-se do comportamento observado em testadores experientes, que naturalmente exploram atalhos e reaproveitam estados intermediários durante a execução de uma suíte de testes. A incorporação explícita do modelo de estado permite evitar sequências inválidas e reduzir a necessidade de reinicializações do sistema sob teste.

Além disso, a adoção de uma abordagem human-in-the-loop mostrou-se fundamental para manter o testador no centro do processo decisório. Diferentemente de soluções totalmente automatizadas, o sistema não impõe uma ordenação, mas atua como um mecanismo de apoio à decisão, aprendendo progressivamente a partir das escolhas

reais do usuário. Essa característica torna a solução particularmente adequada a contextos em que a automação completa é inviável ou indesejada.

A combinação de heurísticas baseadas em estado com técnicas de aprendizado de máquina também contribui para a robustez da abordagem. Enquanto as heurísticas garantem ordenações válidas mesmo na ausência de dados históricos, o componente de aprendizado permite a adaptação contínua do sistema a diferentes perfis de testadores e domínios de aplicação.

6.2 Limitações da Abordagem

Apesar de seus benefícios, a abordagem proposta apresenta algumas limitações que devem ser consideradas.

Primeiramente, a qualidade das recomendações depende da qualidade da modelagem de estado. Em cenários nos quais ações destrutivas ou dependências não são corretamente identificadas, o sistema pode gerar ordenações subótimas ou excessivamente conservadoras.

Em segundo lugar, o aprendizado do sistema está condicionado à disponibilidade de interações humanas. Em estágios iniciais de uso, quando há pouco histórico de execução, as recomendações tendem a se apoiar majoritariamente nas heurísticas, o que pode limitar os ganhos iniciais de eficiência.

Outra limitação refere-se à generalização entre domínios. Embora o modelo seja conceitualmente independente de domínio, a adaptação a novos contextos pode exigir ajustes na definição de ações, estados e critérios heurísticos.

6.3 Ameaças à Validade Interna

Uma possível ameaça à validade interna está relacionada à influência do próprio testador nos resultados observados. Testadores mais experientes podem naturalmente executar testes de forma mais eficiente, independentemente da recomendação fornecida pelo sistema, o que pode dificultar a atribuição causal dos ganhos observados à abordagem proposta.

Além disso, o uso de feedback implícito assume que desvios da recomendação refletem preferências ou estratégias conscientes do testador, o que pode não ser sempre verdadeiro, especialmente em cenários de interrupção ou execução parcial dos testes.

6.4 Ameaças à Validade Externa

No que diz respeito à validade externa, os resultados obtidos podem não se generalizar automaticamente para todos os tipos de sistemas sob teste ou processos de QA. Ambientes altamente dinâmicos ou com estados pouco observáveis podem limitar a efetividade da modelagem proposta.

Da mesma forma, equipes com processos de teste altamente padronizados ou rigidamente definidos podem se beneficiar menos da flexibilidade oferecida pelo sistema, quando comparadas a contextos exploratórios ou semi-estruturados.

6.5 Ameaças à Validade de Construção

Uma ameaça adicional refere-se à definição e mensuração do conceito de “esforço do testador”. Embora métricas como número de passos executados, reaproveitamento de estado e número de reinicializações sejam indicadores relevantes, elas podem não capturar completamente fatores subjetivos, como carga cognitiva ou fadiga.

Essas limitações indicam a necessidade de complementar avaliações quantitativas com análises qualitativas em estudos futuros.

6.6 Considerações Finais da Discussão

De modo geral, a abordagem proposta demonstra potencial para reduzir o esforço associado à execução de testes manuais, preservando o papel central do testador humano e explorando de forma sistemática dependências de estado frequentemente ignoradas por técnicas tradicionais de ordenação.

As limitações e ameaças identificadas não invalidam os resultados apresentados, mas delimitam seu escopo de aplicação e apontam direções claras para trabalhos futuros, discutidos na próxima seção.