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 ainda desempenham um papel central em cenários onde a automação é inviável, 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.

A literatura sobre priorização de casos de teste tem explorado diversas abordagens para ordenar testes de modo a maximizar a detecção precoce de falhas, reduzir custos ou otimizar métricas como cobertura e tempo de execução. Entretanto, a maioria dessas abordagens foca em testes automatizados, utiliza ordenações estáticas ou depende fortemente de histórico de falhas, deixando em aberto o desafio de como apoiar efetivamente o testador humano durante a execução manual de testes, especialmente em cenários sensíveis a estado.

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.

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 estruturas adequadas (por exemplo, grafos ou modelos de estados).
3. Projetar um sistema de recomendação interativo (human-in-the-loop) que sugira ordenações de execução de casos de teste de forma dinâmica, considerando o estado atual do sistema.
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, comparando-a com estratégias tradicionais de ordenação, quanto à redução de esforço, 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

A abordagem proposta adapta-se a diferentes níveis de experiência de testadores, aproximando-se do comportamento observado em testadores experientes ao longo do tempo.

3. Motivação e Exemplo Ilustrativo

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.

Na prática, testadores experientes tendem a explorar a continuidade do estado para reduzir esforço. 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 apresenta a proposta de um sistema de recomendação interativo para a ordenação adaptativa de casos de teste manuais, cujo objetivo é apoiar o testador humano na tomada de decisão durante a execução da suíte de testes, considerando dependências de estado e aprendendo progressivamente a partir do feedback do usuário.

4.1 Visão Geral da Arquitetura

O sistema proposto atua como um mecanismo de apoio à decisão, não executando diretamente os testes, mas sugerindo uma ordem de execução que visa minimizar esforço e redundância. A Figura X ilustra a arquitetura conceitual da solução.

(Figura X: Arquitetura do sistema de recomendação para ordenação de casos de teste manuais)

De forma geral, o sistema é composto pelos seguintes módulos:

1. Módulo de Entrada de Casos de Teste
2. Módulo de Modelagem de Estado
3. Módulo de Recomendação
4. Módulo de Interação e Feedback
5. Módulo de Aprendizado Adaptativo

4.2 Modelo de Entrada

O sistema recebe como entrada uma suíte de casos de teste manuais, onde cada caso é descrito como uma sequência ordenada de ações. Cada ação pode ser enriquecida com metadados simples, fornecidos explicitamente pelo testador ou inferidos automaticamente.

Formalmente, um caso de teste é definido como:

$$t_i = \langle a_1, a_2, ..., a_n \rangle$$

Cada ação possui atributos como:

- tipo da ação (ex.: criação, verificação, modificação),
- impacto no estado (não destrutiva ou destrutiva),
- pré-condições conhecidas.

Esse modelo permite capturar similaridades estruturais entre diferentes casos de teste e identificar ações compartilhadas entre cenários distintos.

4.3 Modelagem de Estado

Para lidar com dependências entre ações, o sistema mantém uma representação explícita do estado do sistema sob teste. Essa representação é modelada como um grafo de estados, no qual:

- Nós representam estados observáveis do sistema;
- Arestas representam ações de teste que provocam transições entre estados.

Ações não destrutivas, como verificações visuais, são modeladas como transições que preservam o estado, enquanto ações destrutivas ou transformadoras resultam em mudanças de estado. Essa distinção permite ao sistema identificar pontos ideais para execução de verificações, evitando que ações posteriores inviabilizem validações anteriores.

4.4 Mecanismo de Recomendação

Com base no estado atual do sistema e na estrutura do grafo de estados, o sistema gera uma ordenação recomendada de ações de teste. Essa ordenação não é fixa, sendo recalculada dinamicamente conforme o testador avança na execução da suíte.

O mecanismo de recomendação considera:

- ações ainda não executadas,
- estado atual do sistema,
- histórico de execuções anteriores,
- preferências aprendidas do testador.

O resultado é uma sequência sugerida que busca maximizar o reaproveitamento de estados intermediários e minimizar repetições desnecessárias de passos.

4.5 Interação Human-in-the-Loop e Feedback

Um aspecto central da proposta é a incorporação do testador humano no ciclo de decisão. O sistema não impõe a ordenação sugerida, permitindo que o testador aceite, ignore ou modifique a recomendação.

O feedback coletado pode ser:

- Implícito: ordem seguida, tempo gasto em cada ação, desvios da recomendação;
- Explícito: avaliações diretas sobre a utilidade da sugestão.

Esses sinais são utilizados para ajustar futuras recomendações, permitindo que o sistema aprenda progressivamente o estilo de execução do testador.

4.6 Aprendizado Adaptativo

O módulo de aprendizado utiliza o feedback acumulado para atualizar o modelo de recomendação ao longo do tempo. Inicialmente, o sistema pode operar com heurísticas simples baseadas na estrutura dos testes e no impacto das ações no estado. À medida que mais dados são coletados, o modelo passa a refletir preferências individuais ou padrões recorrentes observados durante a execução.

Essa abordagem permite que o sistema:

- se adapte a diferentes perfis de testadores,
- evolua com o uso contínuo,
- aproxime suas recomendações do comportamento de testadores experientes.