6 Conclusões e trabalhos futuros

O principal objetivo deste trabalho foi pesquisar na literatura maneiras mais eficazes para facilitar e incentivar a documentação de requisitos testáveis e também minimizar o esforço e as dificuldades para automatizar testes funcionais em aplicações web. Ambos problemas foram observados pelo autor, juntamente com outros integrantes de uma equipe de software, que há três anos adotou a metodologia de desenvolvimento de software ágil ante a metodologia tradicional.

Através das pesquisas e estudos realizados, encontramos no caso de uso um instrumento capaz de apoiar e facilitar a equipe na elicitação e geração dos testes. Propôs-se utilizar uma abordagem de casos de uso direcionados por comportamento para documentar requisitos de software e, a partir dessas informações, automaticamente gerar e executar scripts de teste para verificar se o comportamento funcional de aplicações web está em conformidade com o especificado. Foi escolhida essa abordagem limitada dos casos de uso para facilitar na construção de uma ferramenta.

Para isso foi especificada, projetada e desenvolvida uma ferramenta para viabilizar a documentação de requisitos funcionais na forma de casos de uso, e que pudesse ser utilizada também por pessoas com pouco ou nenhum treinamento em computação, bem como utilizar dados dessa documentação para gerar e executar automaticamente scripts de testes e verificar se o comportamento da aplicações web estão em conformidade que o que foi descrito. A ferramenta desenvolvida disponibiliza um formulário que permite realizar o cadastro de casos de uso e depois utiliza as informações cadastradas, em especial dos fluxos de eventos (principal e alternativos), para gerar e executar scripts de teste utilizando a integração com a ferramenta *Selenium* (Selenium, 2011), responsável pela interação com os elementos da interface web.

A avaliação do desempenho do procedimento e da ferramenta deu-se através da aplicação destes na geração de testes funcionais para dois sistemas reais e através de comparação com técnicas manuais e técnicas tradicionais de geração de testes automatizados aplicadas ao mesmo sistema.

De acordo com os resultados obtidos, os testes gerados automaticamente com o auxílio da ferramenta apresentaram as seguintes vantagens:

- Pouco esforço para redigir todos os fluxos de eventos identificados e assegurar que foram gerados casos de teste suficientes para verificar o comportamento da aplicação web;
- Redução do tempo gasto para gerar testes em comparação com a geração manual;
- Redução no tempo necessário para gerar os testes em comparação com a abordagem *"capture and replay"* utilizando-se *Selenium IDE*;
- Minimizou o problema da falta de documentação de requisitos;
- Melhor detalhamento dos requisitos utilizando casos de uso.
- Documentação que gera testes ao invés de histórias.

Como desvantagens, podemos citar a dificuldade em identificar os objetos do formulário web na hora de digitar os passos, o que prolonga o tempo de escrita caso de uso. Além disso, testes com fluxos longos, com muitos passos, podem gerar casos de uso muito extensos e cansativos para ler.

Assim, acreditamos, perante os resultados observados, que a abordagem apresentada é uma boa alternativa para documentar requisitos, como também para gerar testes funcionais em aplicações web se comparada às técnicas mais utilizadas para geração e execução de testes funcionais, consumindo um tempo menor que as demais técnicas testadas. A abordagem apresentada possibilitou aos usuários uma maneira mais fácil para criar e alterar os testes, uma vez que a ferramenta encapsula os conhecimentos em programação, exigidos na técnica de codificação manual, como também os conhecimentos técnicos para refatorar e inserir oráculos, exigidos na técnica de "*capture and replay*".

Este trabalho permitiu também fazer uma análise detalhada da ferramenta *Selenium* (Selenium, 2011) e dos recursos oferecidos por ela para o desenvolvimento de scripts para automação de testes. Mesmo com todas as deficiências, o *Selenium* mostrou-se ser uma plataforma com bastante potencial para o desenvolvimento de scripts de teste, com suporte a vários recursos, e a possibilidade de extensão de diversas funcionalidades bastante interessantes.

6.1.Trabalhos futuros

Para dar continuidade a este trabalho, foram identificadas as seguintes melhorias na ferramenta proposta:

- Implementar uma funcionalidade para identificar os elementos do formulário *HTML* para o caso de aplicações web já implementadas. A identificação de objetos de interface gráfica continua sendo uma tarefa muito difícil que diminui consideravelmente a viabilidade e rentabilidade da automação de testes.
- Implementar o reaproveitamento de fluxos de eventos de outros casos de uso para diminuir a quantidade de passos em cada fluxo e assim diminuir o tempo de cadastro. Como exemplo, podemos citar o fluxo de eventos "Autenticar com sucesso" do caso de uso "Autenticar na central de relacionamentos" que poderia ser utilizado em todos os outros fluxos da aplicação desse aplicação.
- Implementar a utilização de um banco de dados específico de testes para as aplicações homologadas, onde a ferramenta possa preparar a massa de dados de teste de cada aplicação web, popular o banco com essas informações e no fim da execução do teste realizar a limpeza dos dados. Evitando a possibilidade de erros provindos de informações erradas.
- Melhorar o leiaute do formulário de cadastro de caso de uso e eliminar alguns itens desnecessários para deixar os documentos mais enxutos e interessantes;
- Enfim, realizar novos estudos de caso com diferentes aplicações e uma massa maior de dados para verificar se as melhorias propostas melhoram o processo.