JOGO DAS SETE FALHAS: UM JOGO EDUCACIONAL PARA APOIO AO ENSINO DO TESTE CAIXA PRETA

Lúcio Lopes Diniz, Rudimar Luís Scaranto Dazzi

Grupo de Inteligência Aplicada – Centro de Ciências Tecnológicas da Terra e do Mar – Universidade do Vale do Itajaí (UNIVALI)
Rua Uruguai, 458 - 88302-202 – Itajaí– SC – Brasil

Lucio.diniz@cetil.com.br, rudimar@univali.br

Resumo: *As empresas estão investindo, cada vez mais na qualidade de produto, e uma das maneiras de garanti-la é através da atividade de teste de software. Apesar dessa importância, o teste de software recebe pouca atenção nos currículos de graduação, sendo poucas as horas dedicadas ao seu ensino. Além disso, existe outro fator, que é a dificuldade de ensinar teste de software, já que não é trivial, seja pela falta de tempo disponível ou pela ausência de atividades práticas. Existem várias iniciativas que estão sendo desenvolvidas, visando à inclusão da atividade de teste de software nos currículos de graduação, através da adoção de técnicas de ensino que ajudam/melhoram a aprendizagem dos alunos. A iniciativa proposta neste trabalho é a utilização de um jogo para simular os testes. Os jogos servem para estimular e motivar os alunos, já que possui características desafiadoras, além de prover a parte prática tão necessária ao ensino de teste de software. O jogo foi definido e construído baseado no jogo das 7 falhas (erros) proporcionando feedback e interação durante as atividades. Os resultados das avaliações aplicadas sugerem que o jogo desenvolvido pode ser uma eficiente técnica de ensino a ser utilizada no ensino de teste de caixa-preta.*

Palavras-chave: Jogo para teste de software, ensino de teste de software, jogos de simulação.

1. Introdução

Atualmente, exige-se cada vez mais que as empresas construam softwares com menos defeitos e mais qualidade. A qualidade passou de um diferencial entre as empresas para uma exigência do mercado já que é um dos fatores críticos para o sucesso do negócio, satisfação do cliente e aceitação. A falta de qualidade pode resultar em prejuízos financeiros, em perdas de imagens e de clientes, em insatisfação dos usuários e danos ao meio ambiente, podendo, inclusive resultar em mortes (LAPORTE; APRIL; BENCHERIF, 2007). Em 2002, o National Institute for Science & Technology (NIST) realizou um estudo em que foi estimado que falhas em *software* custam à economia dos Estados Unidos 59,5 bilhões de dólares por ano, o que leva à conclusão de que é necessário um maior investimento na qualidade de *software* (NIST, 2002).

Existem várias técnicas que são utilizadas com o objetivo de garantir a qualidade de software, este trabalho abordará a técnica de teste de software. O teste de software é uma importante técnica utilizada para garantir e melhorar a qualidade do software (MARY, 2000, CHEN, 2004) tendo-se tornado uma parte importante e valiosa dentro do ciclo de vida do desenvolvimento de software. Segundo Myers (2004) teste "é o processo de executar um programa com a intenção de encontrar erros".

Existem duas abordagens para o teste de software: uma utiliza a técnica de teste funcional; a outra, a técnica de teste estrutural. A técnica de teste funcional também é conhecida

como técnica de teste de caixa-preta, visto que não é necessário nenhum conhecimento da lógica interna do sistema para construir os casos de testes (PERRY, 2006). Nessa técnica, os casos de teste são construídos a partir de uma especificação do programa (CHEN; POON, 2004). Já a técnica estrutural recebe o nome de teste de caixa-branca, porque nela é necessário o conhecimento da lógica interna do sistema, para se desenvolverem os casos de testes (PERRY, 2006). Nesta última, os casos de teste são construídos a partir do código fonte do programa (CHEN; POON, 2004)

A técnica de teste de caixa-preta é a mais utilizada na indústria (CHEN; POON, 1998; KANER; PADMANABHAN, 2007). Em termos de eficácia na detecção de falhas, existem alguns estudos realizados, em que a quantidade de erros detectados se utilizando as técnicas de teste de caixa-preta é maior que o número de erros detectados se utilizando as técnicas de teste de caixa-branca, o que sugere que as técnicas de caixa-preta são mais eficazes (BASILI e SELBY, 1987; DAVIS, 1993; KAMSTIES; LOTT, 1995).

Tendo em vista a importância da atividade de teste de *software* para a garantia da qualidade, para a redução dos gastos com problemas em produção e para atender ao aumento da demanda de profissionais dessa área pelo mercado, é necessário investir cada vez mais no processo de ensino e aprendizagem dessa atividade, aumentando, assim, a qualificação desses profissionais.

Nesse sentido esse trabalho apresenta um jogo denominado “Jogo das 7 Falhas”, que busca auxiliar o ensino de testes de software (caixa preta) a alunos de cursos de graduação e profissionais que não tiveram formação nesse contexto.

2. Trabalhos Relacionados

Neste tópico serão apresentados alguns trabalhos relacionados, porém como nas pesquisas realizadas até meados de 2010 não foi localizado nenhum jogo (publicado) aplicado ao ensino de testes de caixa-preta, as pesquisas foram ampliadas para jogos na área de engenharia de software. Esta é a área que envolve os testes de software e o teste de caixa-preta.

O projeto SmartSim tem por objetivo primário o desenvolvimento de um framework para “jogos educacionais de negócio baseado em simulação”, com ênfase na gestão de pessoas, via a utilização de atores sintéticos para representar as personalidades humanas envolvidas no contexto da simulação integrando o conhecimento sobre pessoas e organizações.

O Virtual Team (CIn/UFPE, 2006) representa então, um jogo sério baseado em simulação de negócio no domínio de gerenciamento de projetos de software, visando desenvolver determinadas habilidades comportamentais e gerenciais em gerente de projetos de software novatos, via a utilização de atores sintéticos. As habilidades comportamentais e gerenciais a serem desenvolvidas no jogador são direcionadas as práticas de gerenciamento de pessoas em ambientes de software e se baseiam fortemente nos processos estabelecidos pelo PMBOK[1]. Os dois processos abordados dentro do gerenciamento de recursos humanos são: desenvolvimento da equipe de projeto e gerenciamento da equipe de projeto.

O Planager v1.6 (KIELING; ROSA, 2006) é um jogo de Gerência de Projetos que foi desenvolvido com base nos modelo conceitual do PMBOK. Objetivo geral é apoiar o ensino de conceitos de gerência de projetos de software de uma maneira mais didática do que as encontradas na literatura e nos cursos existentes para alunos que possuem pouca experiência sobre o assunto.

[1] O Project Management Body of Knowledge, também conhecido como PMBOK é um conjunto de práticas em gestão de projetos ou gerência de projetos publicado pelo Project Management Institute (PMI) e constitui a base do conhecimento em gerência de projetos do PMI.

O X-Med (Lino, 2007) é um jogo educacional que simula a realização de um programa de medição de software, englobando seu planejamento e execução voltada para a monitoração de projeto de software. Neste jogo o jogador tem a oportunidade de treinar e avaliar suas habilidades relacionadas à medição e análise de software, uma vez que o jogo permite que o jogador participe ativamente no desenvolvimento do programa de medição, tomando decisões relativas à medição e análise. O principal objetivo é a aplicação da medição de software no contexto de gerenciamento de projetos alinhados com os conceitos de maturidade do nível 2 do CMMI-DEV v1.2 (CMMI-DEV, 2006) e baseados no GQM – Goal/Question/Metric (BASILI e SELBY, 1987) e PSM – Practical Software and Systems Measurement (MCGARRY, 2003), bem como ajudar os participantes a adquirirem as habilidades necessárias para distinguir e descrever os elementos de um programa de medição.

O SIM Project Management Simulation (CALTRANS, 2008) é um jogo educacional voltado para a área de tomada de decisão onde o gerente de projetos tem a possibilidade de gerenciar a formação de um novo Planeta Terra. O jogo é dividido em lições que devem ser executadas durantes três dias. O principal objetivo é simular as tomadas de decisões baseadas nos conceitos de gerência de projetos lidando com as pressões de pessoas no decorrer da execução do projeto para a construção do planeta. O gerente de projetos deve aplicar durante toda a execução do projeto todos os estilos de comunicação para se comunicar com o time do projeto e com os interessados.

O IT MANAGER GAME (INTEL, 2006) é um simulador on-line onde o jogador pode gerenciar o departamento de TI de uma empresa virtual. O jogo foi lançado em 146 paises, em 11 idiomas diferentes. O IT Manager já é sucesso no mundo todo, com 1,2 milhões de pageviews. São 35 mil jogadores, na maioria profissionais e estudantes de TI, que jogam em média 8 horas cada um. E é na Inglaterra e Estados Unidos onde ficam o maior número de jogadores (SABOYA, 2007). O principal objetivo é simular o gerenciamento de um departamento de TI criando uma empresa mais eficiente e lucrativa possível por meio de liderança tecnológica e melhor ambiente de trabalho.

O SIMULTRAIN 7 (STS, 2005) é um simulador utilizado na gestão de projetos de programas de treinamento. O programa de simulação ocorre em dois períodos e é utilizada em uma sala de treinamento. A simulação é feita em um grupo de 3 a 4 pessoas. Está disponível em mais de 10 línguas, entre elas o inglês, francês, espanhol e alemão e já foi utilizado em mais de 30 países e tem contribuído para a formação de mais de 60.000 gerentes de projetos. O principal objetivo é proporcionar ao aluno a possibilidade de simular a gestão de projetos da mesma forma que pilotos usam as simulações de vôo para a sua formação. Os alunos confrontam as situações e problemas que surgem durante a execução do projeto e podem ver imediatamente as conseqüências das decisões tomadas. Além disso, eles podem acompanhar a evolução dos custos, o calendário, a qualidade e os fatores humanos.

PnP é um jogo de cartas direcionado ao ensino de engenharia de software, desenvolvido na Universidade da Califórnia e amplamente utilizado em diversas instituições. Seu objetivo é simular o processo de desenvolvimento de sistemas desde a concepção até a fase de entrega do software (BAKER, HOEK, 2005). Problems and Programmers é um jogo de cartas educacional para engenharia de software. Destina-se a simular o processo de desenvolvimento de software desde a concepção até sua conclusão. Os jogadores competem entre si e o objetivo é terminar os seus projetos, evitando os potenciais perigos da engenharia de software. Estes jogadores irão aprender rapidamente quais estratégias irão deixá-los ganhar e o jogo irão ajudá-los em tempo real.

O SimulES (Simulador de Uso da Engenharia de Software) (FIGUEIREDO et al. 2006), que incorpora as soluções propostas. É um jogo que é apresentado de forma comparativa ao PnP – Problems and Programmers. O objetivo de SimulES é que jogadores, idealmente alunos,

disputem para terminar um projeto de software e o vencedor será aquele que primeiro entregar ao cliente um produto com qualidade adequada.

O BELTS Challenge (SETEC, 2009) foi desenvolvido pela SETEC Consulting Group e é um jogo de estratégia e conhecimento. O jogador que quiser sagrar-se vencedor deverá aliar suas habilidades estratégicas a uma boa dose de conhecimento sobre Seis Sigma[2]. Aprofundar o conhecimento sobre Seis Sigma. Através do jogo os jogadores têm a oportunidade de reforçar os conhecimentos sobre o assunto.

O Jogo da ISO 14001 (SETEC, 2009) foi desenvolvido pela SETEC Consulting Group e é um jogo em que os participantes terão a oportunidade de conhecer mais sobre os requisitos e as características dessa norma e poderão vivenciar um processo fictício de certificação. Para tal, será promovida uma corrida. Vence o jogo aquele participante que chegar primeiro ao final do processo de certificação da ISO 14001. Através do jogo se pode entender o que são requisitos e conhecer alguns deles, aprender o que são situações conformes e não-conformes e tomar consciência de que são as pessoas e suas atitudes que fazem o sucesso da ISO 14001.

3. O Jogo das 7 Falhas

Um dos objetivos deste trabalho é desenvolver um jogo educacional de simulação baseado em computador que simule a execução de casos de testes de caixa-preta. Para tal é importante que os objetivos sejam capazes de motivar o jogador e que as regras definidas sejam claras para que o jogador possa tomar as decisões corretas para alcançar tais objetivos, já que um jogo nada mais é do que uma série de escolhas significativas (BRATHWAITE, SCHREIBER, 2009).

Embora o jogo seja voltado para pessoas iniciantes em teste de software, é necessário que os jogadores possuam alguns conceitos básicos de testes de software, saibam diferenciar erros, defeitos e falhas, conheçam testes de sistemas e a técnica de teste de caixa-preta. Também é pré-requisito que o jogador conheça o que é um caso de teste e as técnicas de seleção de casos de testes: classe de equivalência e análise de valor limite.

O jogo consiste em descobrir as falhas existentes nas funcionalidades de um software a ser testado em menos tempo possível. O jogo possui dois níveis de complexidade, baixa e média. Cada nível é composto por uma funcionalidade onde existem sete falhas a serem descobertas. O jogador só passará para o nível seguinte caso descubra as sete falhas existentes no nível dentro do tempo estimado (25 minutos para o nível um e 40 minutos para o nível dois). Caso o tempo se esgote antes de o jogador identificar as sete falhas em cada nível, o jogo se encerra, e o jogador é eliminado. Caso o jogador descubra as sete falhas do nível um e as sete falhas do nível dois dentro do tempo, o jogador é o vencedor, tendo alcançado o objetivo proposto pelo jogo.

Aqui serão apresentadas as escolhas possíveis e os desafios propostos pelo jogo das sete falhas. O jogo das sete falhas é um jogo “single player”, no qual o jogador assume o papel de testador em uma equipe de teste de software de uma empresa fictícia chamada “Diniz Quality Assurance”, com a finalidade de descobrir as sete falhas existentes em cada funcionalidade testada, correlacionando as falhas com uma classe de equivalência ou um valor limite.

Ao entrar no jogo, o jogador é apresentado à tela inicial (figura 1). Nesta tela existem dois botões: “Iniciante” e “Avançado. Ao selecionar a opção “Iniciante”, o jogador será levado à tela de login, cujo objetivo é identificar, através de um usuário, como cada jogador se saiu no jogo, a pontuação obtida, os erros identificados, se usou a opção de dica, entre outras informações. Na tela de login, são dadas as boas vindas, e também é solicitado ao jogador que leia as regras do jogo (disponíveis pelo botão “Regras”) e os requisitos das funcionalidades a

[2] Seis Sigma ou Six Sigma (em inglês) é um conjunto de práticas originalmente desenvolvidas pela Motorola para melhorar sistematicamente os processos ao eliminar defeitos.

serem testadas (disponíveis pelo botão “Requisitos”). Também é solicitado, após aleitura dessas informações, que o jogador faça o seu login para iniciar o jogo.

Figura 1: Tela Inicial do Jogo das Sete Falhas.

Ao pressionar o botão “Regras”, o jogador é apresentado à descrição da tarefa que deverá ser executada. Ela consiste do objetivo do jogo, de quanto tempo o jogador dispõe para alcançar o objetivo, das regras do jogo e de como a pontuação final é calculada. O objetivo do jogo é bem claro: encontrar as sete falhas dentro de cada nível o mais rápido possível. Ao pressionar o botão “Requisitos”, o jogador é apresentado aos requisitos das funcionalidades que deverão ser testados e validados. Por exemplo: características dos campos, regras de negócios, regras de botões e mensagens de erros e de sucesso, que deverão ser apresentadas em determinadas situações.

Após efetuar o login, o jogador recebe o título de Analista de Teste Junior, sendo exibida a tela de inicio do jogo (nível 1). Neste momento, também são sorteados as sete falhas que farão parte da jogada. A cada novo login do usuário, as sete falhas são sorteadas aleatoriamente dentre as 33 possíveis. O sorteio teve como objetivo proporcionar aos jogadores novos desafios e motivá-los a jogarem o jogo mais vezes, já que, a cada jogada, eles terão um novo desafio a completar. A tela inicial do jogo, conforme pode ser vista na Figura 2, é dividida em duas partes:

Figura 2: Tela inicial do jogo (nível 1).

- Lado Esquerdo - Nele se encontram os controles do jogo; o relógio que já estará em andamento (será iniciado com 25 minutos); um botão de Dicas, pelo qual o jogador terá direito a apenas uma dica sobre onde uma das falhas pode estar localizada. Também nesse lado, ficam a lista de falhas, que é preenchida à medida que as falhas são descobertas, a pontuação do jogo e os botões “Requisitos”, onde estão listados os requisitos da funcionalidade do nível um e “Regras”, contendo o objetivo e as regras do jogo.
- Lado Direito – Na sua parte superior se encontra a funcionalidade a ser testada, que, no caso do nível um, é o “Cadastro de Usuário”, com os campos “Nome”, “Usuário”,

"Senha" e "Confirmação", e os botões "Salvar" e "Limpar" (habilitados), além de "Excluir" e "Imprimir" (desabilitados). Na parte inferior, após uma falha ser simulada, são exibidas sete classes de equivalência e/ou valores-limite, para que o jogador possa selecionar uma delas. Ali, também, são apresentados o feedback e as mensagens, ao longo do jogo.

Ao iniciar o jogo, o jogador terá 25 minutos para identificar as sete falhas existentes na funcionalidade "Cadastro de Usuário", que estarão em desacordo com os requisitos da funcionalidade. Para identificar as falhas, o usuário irá executar os casos de testes elaborados anteriormente através da utilização das técnicas de classe de equivalência e análise de valor-limite. A cada caso de teste executado que não originar uma falha, será exibida, na parte inferior, a mensagem correspondente ao resultado esperado e um feedback informando a qual classe de equivalência ou valor-limite o caso de teste executado corresponde.

A Figura 3 mostra um exemplo onde foi executado o seguinte caso de teste: Passo 1 – Deixar o campo "Código" em branco; Passo 2 – Preencher o campo "Usuário" com um usuário válido; Passo 3 – Preencher o campo "Senha" com uma senha válida; Passo 4 – Preencher o campo "Confirmação" com o mesmo valor preenchido no campo "Senha"; Passo 5 – Pressionar o botão "Salvar". Este caso de teste tem como resultado esperado: Exibir a mensagem: "O campo Nome é obrigatório". Após a sua execução, são exibidos, na parte inferior do lado direito, o resultado esperado "O campo Nome é obrigatório." e o Feedback "Este caso de teste executado corresponde a Classe de equivalência inválida – Nome em branco.

Figura 3: Resultado esperado e Feedback exibidos após a execução.

O feedback após a execução de cada caso de teste tem como objetivo reforçar os conceitos de classe de equivalência e análise de valor-limite, prover a competência de correlacionar os casos de teste aos conceitos aprendidos, além de proporcionar ao jogador a oportunidade de incluir ou corrigir algum caso de teste na lista de casos de teste a serem executados.

Para cada caso de teste executado que originar uma falha, será exibida, na parte inferior, uma mensagem informando que uma falha foi descoberta e que é necessário selecionar a classe de equivalência ou o valor-limite que a originou. Também será exibida, abaixo da mensagem, uma lista com sete classes de equivalência e/ou valores-limite, para que o jogador selecione, entre elas, a que deu origem à falha simulada.

Considerando que um dos erros sorteados seja que o sistema esteja permitindo cadastrar usuário com caracteres especiais, quando o requisito informa que isso não seria permitido. A Fugura 4 corresponde a um exemplo onde foi executado o seguinte caso de teste para simular essa falha: Passo 1 – Preencher o campo Código com um valor válido; Passo 2 – Preencher o campo "Usuário" com um nome que possua pelo menos um caractere especial (!, @, #, %, *, ...); Passo 3 – Preencher o campo "Senha" com uma senha válida; Passo 4 – Preencher o campo "Confirmação" com o mesmo valor preenchido no campo "Senha"; Passo 5 – Pressionar o botão "Salvar". Este caso de teste tem como resultado esperado: Exibir a mensagem: "O campo Usuário não pode conter caracteres especiais". Como este foi um dos sete erros sorteados, a

mensagem não será exibida, sendo, então, simulada a falha. Após a execução desse caso de teste pelo jogador, será exibida, na parte inferior do lado direito a seguinte mensagem: “Parabéns, você simulou uma das sete falhas. Para ganhar os pontos, selecione a classe de equivalência ou valor-limite que originou esta falha”. Logo abaixo dessa mensagem, é exibida a lista com sete classes de equivalência/valores-limite que foram sorteados aleatoriamente, contendo sempre a classe de equivalência ou valor-limite correto entre os sete, para que o jogador possa selecioná-lo.

Figura 4: Tela exibida após a simulação de uma falha

O jogador após simular uma falha, deverá, então, selecionar a classe de equivalência ou valor-limite que originou a falha. Caso o jogador selecione a classe de equivalência ou o valor limite-correto, ele receberá dez pontos. A classe de equivalência ou o valor-limite selecionado irá para a parte esquerda da tela, no item falhas encontradas, e será exibido o feedback “Parabéns você selecionou a classe de equivalência ou valor limite que originou a falha e ganhou 10 pontos.” conforme exibido na Figura 5. Caso o jogador selecione a classe de equivalência ou o valor-limite errado, ele perderá dez pontos, e será exibido o feedback “Infelizmente você não selecionou a classe de equivalência ou valor limite que originou a falha e perdeu 10 pontos.” como mostra a Figura 6.

Figura 5: Tela de feedback exibida após a seleção da classe de equivalência ou valor limite correto.

Figura 6: Tela de feedback exibida após a seleção da classe de equivalência ou valor limite errado.

Caso o jogador venha simular novamente uma falha já simulada anteriormente, e que o jogador já tenha acertado a classe de equivalência ou valor limite que originou esta falha, será exibida um feedback informando que esta falha já foi encontrada e será informada a classe de equivalência ou valor limite que originou esta falha.

No nível um, existe ainda a possibilidade de o jogador obter uma dica sobre a localização de uma das falhas, bastando, para isso, pressionar o botão “Dicas”. Porém ao pressionar esse

botão, o jogador perde 4 pontos. Essa penalidade foi colocada para diferenciar o jogador que descobriu as sete falhas sem necessitar de ajuda, do aluno que precisou de uma dica para descobrir as sete falhas. Um exemplo de dica seria “Verifique todas as classes de equivalência do campo Nome”.

Caso o jogador não encontre as sete falhas do nível um dentro dos 25 minutos, será exibido um feedback, informando que o tempo expirou, sendo o jogador eliminado. Caso o jogador encontre as sete falhas dentro dos 25 minutos, ele será promovido à analista de teste pleno, e passará para o segundo nível do jogo. Além disso, o jogador ganhará um ponto a cada intervalo de 10 segundos que sobrar do tempo disponível. Por exemplo: se o jogador descobrir as sete falhas em 24 minutos, ou seja, 60 segundos menos que o tempo disponível, ele ganhará mais seis pontos. Essa recompensa foi colocada para diferenciar os jogadores que descobriram as sete falhas em menos tempo (lembrando que o objetivo do jogo é descobrir as sete falhas em cada nível o mais rápido possível). A Figura 14 mostra um exemplo em que as sete falhas do nível um foram descobertas em 15 minutos e 20 segundos. O jogador ganhou, então, 28 pontos extras, sendo promovido a analista de teste pleno e passando para o nível dois

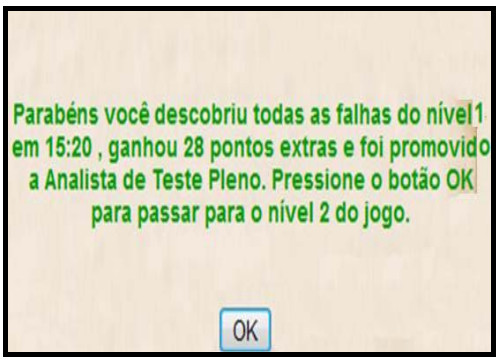

Figura 7: Tela de Feedback exibida após a descoberta das sete falhas do nível um.

Ao pressionar Ok para o feedback de finalização do nível um exibido na figura 7, o jogador vai para o nível dois. A mecânica do nível dois é a mesma do nível um. A diferença realmente está no grau de complexidade das falhas; por isso, não é apresentado em mais detalhes. Da mesma forma que no nível um, caso o jogador não encontre as sete falhas do nível dois em 40 minutos, será eliminado. Caso descubra as sete falhas do nível dois dentro do intervalo de 40 minutos, será promovido a analista de teste sênior e sairá como vencedor, ganhando também um ponto extra a cada intervalo de 10 segundos que sobrar dos 40 minutos.

3. Conclusões

O jogo das 7 falhas foi testado por um grupo de 5 (cinco) profissionais de uma empresa especializada em qualidade de software. Os testes realizados buscaram principalmente identificar falhas operacionais no jogo e obter uma analise dos testadores sobre a dinâmica do jogo. O principal objetivo dos testes foi identificar eventuais erros que pudessem prejudicar os experimentos com alunos para avaliar a efetividade do jogo quanto aos aspectos pedagógicos e de ensino. Foram encontrados apenas 2 erros, corrigidos antes da aplicação com alunos de graduação.

Após a correção dos erros, foram efetuados experimentos com 3 diferentes turmas do curso de graduação em Ciência da Computação, na disciplina de Engenharia de Software. Os experimentos foram aplicados com 11, 7 e 21 alunos respectivamente. No primeiro e terceiro experimento os alunos foram divididos aleatoriamente em grupo experimental e de controle. Os experimentos utilizaram as estratégias de pesquisa quantitativa e qualitativa. A estratégia quantitativa foi utilizada para avaliar a efetividade do ensino proporcionado pelo jogo. A estratégia qualitativa foi utilizada para avaliar, sob a ótica dos participantes, se o jogo torna o

processo de aprendizagem mais atrativo. Os resultados obtidos após a aplicação do teste estatístico de Mann-Whitney podem ser considerados positivos, pois em nenhum caso o resultado do grupo experimental foi inferior ao do grupo de controle, sendo inclusive superior no terceiro experimento. Em relação à avaliação qualitativa 87% consideraram o jogo agradável e 78% gostaram de jogá-lo.

A comparação entre os dados obtidos nos três experimentos leva à conclusão de que o jogo deve ser utilizado em alunos que já tenham um conhecimento prévio dos requisitos a serem testados e que já tenham derivados os casos de teste, para que possam entender a dinâmica do jogo e jogar adequadamente. Isso nos leva a conclusão de que o jogo é melhor aplicado como apoio a disciplina do que como auto ensino.

A principal contribuição deste trabalho foi a concepção e a implementação do Jogo das 7 Falhas, onde os experimentos realizados indicaram que o jogo desenvolvido é uma atividade efetiva de ensino, capaz de motivar os alunos, servindo, dessa forma, como apoio ao instrutor no ensino do teste de caixa-preta.

Referencias

BAKER, A., Navarro, E. AND HOEK A. "An Experimental Card Game for Teaching Software Engineering Processes". In: Journal of Systems and Software, v. 75, 1-2, pp. 3-16. 2005.

BASILI, Victor, SELBY, Richard Comparing the Effectiveness of Software Testing Strategies IEEE Transactions on Software Engineering , 1987, Pages: 1278 – 1296.

BRATHWAITE, Brenda; SCHREIBER, Ian. Challenges for Game Designers. Boston, Charles River Media, 2009

CALTRANS: [Online] Disponível em URL http://www2.dot.ca.gov/hq/cpsd/PM_sim/ [Acessado em 05 de Junho 2008]

CHEN, T. Y.; POON, P. L. Experience with teaching black-box testing in a computer science/software engineering curriculum. IEEE Transactions on Education, v. 47, n. 01, p. 42-50, 2004.

CHEN, T. Y.; POON, P. L. Teaching Black Box Testing Proceedings of the 1998 International Conference on Software Engineering: Education & Practice, 1998 Page: 324.

CIN/UFPE, 2006. [Online Versão Português]. Disponível em URL: http://www.cin.ufpe.br/~smartsim/portugues.html [Acessado em 25 de julho de 2008].

CMMI-DEV. CMMI Product Team. CMMI for Development, version 1.2, Software Engineering Institute, 2006. Disponível em URL: http://www.sei.cmu.edu/pub/documents/06.reports/pdf/06tr008.pdf [Acessado em 04 de junho de 2009].

DAVIS, ALAN Software Requirements: Objects, Functions and States, Revision Prentice Hall, Upper Saddle River, New Jersey, 1993.

FIGUEIREDO, E., Lobato, C., Dias, K., Leite, J. e Lucena, C. "SimulES: Um Jogo para o Ensino de Engenharia de Software". Relat. Técnico 34/06, Depto de Informática, PUC-Rio. 2006.

INTEL: 2006. [Online Versão Português] Disponível em URL: http://itmg2.intel.com/por/launch/default.aspx [Acessado em 26 de junho 2008]

KAMSTIES, Erik, LOTT, Christopher An Empirical Evaluation of Three Defect-Detection Techniques. Proceedings of the 5th European Software Engineering Conference, 1995, Pages: 362 – 383.

KANER, Cem, PADMANABHAN, Sowmya Practice and Transfer of Learning in the Teaching of Software Testing. Proceedings of the 20th Conference on Software Engineering Education & Training – 2007 pages 157 – 166.

KIELING, E.; ROSA, R. Planager – Um Jogo para Apoio ao Ensino de Conceitos de Gerência de Projetos de Software. Monografia de Trabalho de Conclusão de Curso de Graduação, Ciência da Computação, FACIN, PUCRS, Porto Alegre, 2006.

LAPORTE, C. Y.; APRIL, A.; BENCHERIF, K. 2007. Teaching Software Quality Assurance in an Undergraduate Software Engineering Program. Software Quality Professional, 4-10.

LINO, J. I. Proposta de um Jogo Educacional para Medição e Análise de Software. Trabalho de Conclusão de Curso, Sistemas de Informação, Universidade Federal de Santa Catarina, Florianópolis, 2007.

MARY, Jean, Harrold. Testing: A Roadmap. International Conference on Software Engineering. Proceedings of the Conference on The Future of Software Engineering 2000, Limerick, Ireland June 04 - 11, 2000.

MCGARRY, J.; et. Al. Practical Software and Systems Measurement: A Foundation for Objective Project Management, version 4.0c. Severna Park, MD: Practical Software and Systems Measurement, Março 2003. Disponível em URL: http://www.psmsc.com. [Acessado em 10 de julho de 2009].

MYERS, Glenford, J. The Art of Software Testing. Second Edition, Wiley, 2004.

NIST. National Institute for Science & Technology - Planning Report 02-3: The Economic Impact of Inadequate Infrastructure for Software Testing. Prepared by RTI for the National Institute of Standards & Technology, USA, May 2002. Disponível em: http://www.nist.gov/director/prog-ofc/report02-3.pdf.

PERRY, W. E. Effective Methods for Software Testing Third Edition, Wiley, 2006.

SABOYA, A., 2007. [Online] Disponível em: http://aluisiosaboya.com/2007/06/11/it-manager-game-20-da-intel/ [Acessado em 27 de junho 2008]

SETEC: 2009. [Online]. Disponível em URL: <http://www.setecnet.com.br/index.php> [Acessado em 02/09/2009].

STS: 2005. [Online] Disponível em URL: http://www.sts.ch/smt7/demo/demo_e.htm [Acessado em 26 de junho 2008]