AVALIAÇÃO DE QUALIDADE DE UM AMBIENTE DE APOIO AO ENSINO DE PROGRAMAÇÃO

Draylson Micael de Souza – ICMC-USP – draylson@icmc.usp.br
Marisa Helena da Silva Batista – ICMC-USP – marisa@grad.icmc.usp.br
Ellen Francine Barbosa – ICMC-USP – francine@icmc.usp.br

Resumo. Várias iniciativas têm sido propostas a fim de automatizar o processo de avaliação de trabalhos práticos de programação. O ambiente PROGTEST insere-se neste contexto como uma ferramenta de apoio à avaliação de trabalhos práticos de programação e teste de software. Experiências têm mostrado que o uso da PROGTEST traz benefícios para o ensino e aprendizado. No entanto, foi possível observar que melhorias técnicas na qualidade do ambiente proveriam um maior incentivo ao seu uso e adoção. Assim, este trabalho teve por objetivo realizar uma avaliação de qualidade da PROGTEST. Os resultados mostraram que, embora alguns aspectos ainda necessitam de melhorias, o ambiente PROGTEST possui uma boa qualidade. Sugestões de melhoria para a evolução da PROGTEST também puderam ser identificadas.

Palavras-chaves: avaliação de qualidade, avaliação automática de trabalhos de programação, ensino programação, teste de software

Abstract. Several initiatives propose to automate the assessment of practical programming assignments. The PROGTEST environment fits into this context as a tool to support the assessment of programming and software testing assignments. Experiments have shown that the use of PROGTEST brings benefits to the teaching and learning. However, we observed that improvements in the quality of the environment would provide a greater incentive to its use and adoption. Thus, this study aimed to conduct a quality assessment of PROGTEST. The results showed that although some aspects still need to be improved, the PROGTEST has a good quality. We could also identify a number of suggestions for improvement to the evolution of PROGTEST.

Keywords: quality evaluation, automated assessment of programming assignments, learning of programming, software testing

1. Introdução

A avaliação de trabalhos práticos em disciplinas de programação é uma tarefa difícil e, em geral, exige muito esforço e tempo do professor. O problema agrava-se principalmente quando o número de alunos envolvidos é grande.

Procurando minimizar tais esforços, iniciativas têm sido propostas a fim de automatizar o processo de submissão e avaliação de trabalhos práticos de programação (Rubio-Sánchez et al., 2014; Fernandez Aleman, 2011; English e Rosenthal, 2009; Douce et al., 2005). Dentre tais iniciativas, destacam-se ambientes que utilizam técnicas e critérios de teste para avaliar os programas e os conjuntos de teste desenvolvidos pelos alunos (Edwards e Perez-Quinones, 2008; Spacco et al., 2006; Souza et al., 2011a).

A ideia de avaliar os conjuntos de teste dos alunos de forma complementar à avaliação dos programas é uma das iniciativas que têm sido investigadas a fim de

introduzir conceitos de teste em disciplinas introdutórias de computação. Experiências têm sugerido que a introdução da atividade de teste pode ajudar o desenvolvimento das habilidades de compreensão e análise nos alunos, já que para sua condução é necessário que os alunos conheçam o comportamento dos seus programas (Edwards, 2004).

O ambiente PROGTEST insere-se neste contexto como uma ferramenta de apoio à submissão e avaliação de trabalhos práticos de programação, com base em teste de software (Barbosa et al., 2008; Souza et al., 2011). Por meio de ferramentas de teste integradas, a PROGTEST avalia os trabalhos submetidos pelos alunos, fornecendo a eles um *feedback* sobre a correção de seus programas e a adequação dos seus testes.

Experiências têm sido realizadas a fim de investigar a adoção da PROGTEST em disciplinas de graduação e pós-graduação. Em geral, o seu uso tem sido efetivo no ensino e aprendizado tanto de programação como de teste de software. No entanto, foi observado que melhorias na qualidade do ambiente proporcionariam um maior incentivo ao seu uso e adoção (Souza et al., 2013, 2014).

Este trabalho tem por objetivo complementar os estudos anteriores, realizando uma avaliação de qualidade do ambiente PROGTEST. A ideia é identificar possíveis deficiências e fornecer sugestões de melhorias a serem realizadas no ambiente. Para isto, foi utilizado como base o processo de avaliação descrito na norma NBR ISO/IEC 14598 (ABNT, 2001). De modo geral, os resultados obtidos indicam que o ambiente PROGTEST possui uma boa qualidade.

O restante deste artigo está organizado da seguinte forma. Na Seção 2 é apresentado o ambiente PROGTEST e as suas principais funcionalidades. Na Seção 3 são descritos a metodologia e os artefatos considerados na avaliação. Na Seção 4 são apresentados e discutidos os resultados da avaliação e as melhorias propostas para a PROGTEST. Por fim, na Seção 5 são apresentadas as conclusões e trabalhos futuros.

2. O Ambiente PROGTEST

A PROGTEST é um ambiente web para submissão e avaliação automática de trabalhos de programação baseada em atividades de teste (Barbosa et al., 2008; Souza et al., 2011). A PROGTEST avalia tanto a qualidade do código como a qualidade dos testes desenvolvidos pelos alunos. Para isso, ferramentas de teste são integradas ao ambiente.

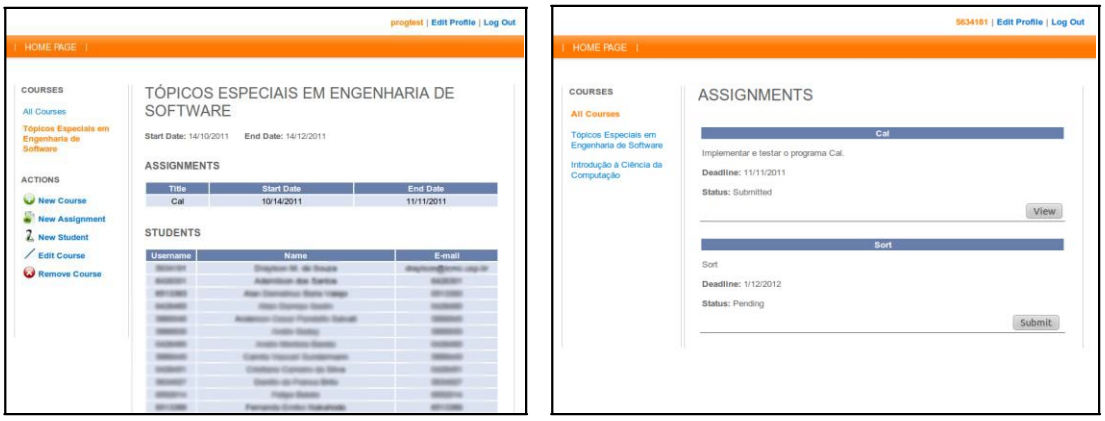
A PROGTEST foi desenvolvida em Java. Atualmente, ela apoia a execução automática de casos de teste, bem como a compilação e o teste estrutural e baseado em erros de programas escritos em linguagem C e Java.

As principais funcionalidades da PROGTEST podem ser acessadas por meio de duas visões – professor ou aluno. Em relação à visão do professor (Figura 1(a)), a PROGTEST permite ao usuário criar novos cursos, matricular alunos nos cursos e definir trabalhos de programação.

Para avaliar os trabalhos de programação dos alunos, o ambiente requer um "trabalho oráculo", o qual consiste em: (1) um programa de referência fornecido pelo professor, que implementa a solução correta para o trabalho proposto; e (2) um conjunto de teste para o programa implementado, sendo que este deve ser 100%-adequado aos critérios de teste considerados na avaliação.

Considerando a visão do aluno (Figura 1(b)), este tem acesso a todos os trabalhos associados aos cursos em que ele se encontra matriculado. Para submeter uma solução para um dado trabalho, o aluno deve enviar seu programa e o conjunto de teste que utilizou para testá-lo. A PROGTEST compila o programa do aluno e, por meio de

ferramentas de teste integradas à ela, calcula a cobertura para as seguintes combinações: (i) programa do aluno com o conjunto de teste do aluno; (ii) programa do professor com o conjunto de teste do aluno; e (iii) programa do aluno com o conjunto de teste do professor. Com base no resultado dessas execuções, a PROGTEST é capaz de sugerir uma nota para a solução submetida.



(a) PROGTEST: Visão do Professor (b) PROGTEST: Visão do Aluno

Figura 1. O Ambiente PROGTEST

Logo após ter submetido seu trabalho, o aluno pode visualizar o relatório de avaliação, além de outros relatórios fornecidos pelas ferramentas integradas. Se o trabalho submetido não estiver correto, o aluno poderá submeter novas versões do seu trabalho até atingir nota máxima.

A PROGTEST tem sido utilizada em cenários reais de ensino, considerando tanto o ensino de programação (Souza et al., 2013) como o ensino de teste de software (Souza et al., 2014). Em síntese, experiências tem sido realizadas envolvendo atividades com a PROGTEST em disciplinas oferecidas no ICMC/USP. Em tais atividades, os alunos eram encorajados a implementarem e testarem programas comumente utilizados em disciplinas de programação (ordenação, cálculo de fatorial, estruturas de dados, etc.).

Os resultados obtidos nessas experiências indicaram uma boa efetividade do aprendizado proporcionada pelo ambiente. Basicamente, a PROGTEST encoraja os alunos a escreverem casos de teste para os seus trabalhos, melhorando a qualidade dos seus conjuntos de teste. Além disso, ao executarem os testes, na maioria das vezes também são revelados erros nos programas desenvolvidos. Procurando solucionar os erros identificados, os alunos também aumentam a qualidade dos seus programas.

Apesar dos bons resultados obtidos na aplicação prática da PROGTEST, também foi possível notar que alguns problemas de qualidade no ambiente influenciavam negativamente os alunos na condução das atividades. Assim, a fim de identificar tais deficiências e propor melhorias à PROGTEST, uma avaliação de qualidade foi conduzida. Tal avaliação é descrita a seguir.

3. Avaliação de Qualidade da PROGTEST

Como discutido anteriormente, o objetivo deste trabalho foi realizar uma avaliação de qualidade do ambiente PROGTEST. A metodologia adotada foi estabelecida com base na norma NBR ISO/IEC 14598 (ABNT, 2001), que define um processo para a avaliação de um pacote de software.

A Figura 2 ilustra as etapas do processo de avaliação definidos pela norma, a saber: (1) estabelecimento dos requisitos de avaliação, em que são estabelecidos o propósito da avaliação, os tipos de produtos sendo avaliados e o modelo de qualidade a ser adotado; (2) especificação da avaliação, em que são definidas as métricas a serem adotadas, os seus níveis de pontuação e os critérios que serão utilizados para julgar a qualidade do produto; (3) projeto da avaliação, que consiste no desenvolvimento de um plano de avaliação; e (4) execução da avaliação, que consiste em obter as medidas segundo as métricas adotadas, comparar os resultados com os critérios de julgamento definidos e, consequentemente, obter um julgamento sobre a qualidade do produto.

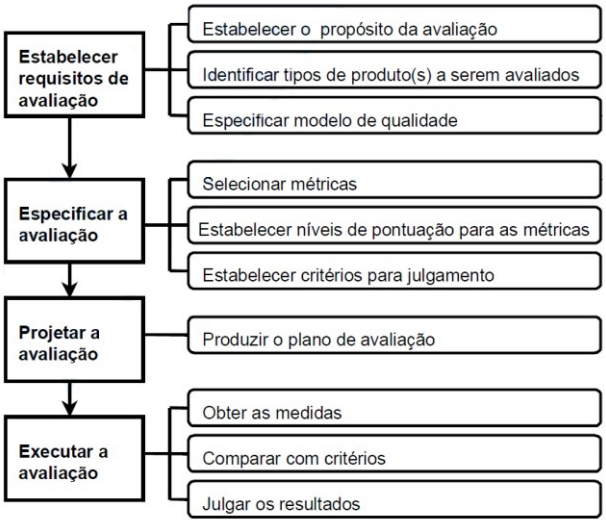


Figura 2. Processo de Avaliação de Qualidade

3.1. Estabelecimento dos Requisitos de Avaliação

O propósito da avaliação foi identificar possíveis deficiências no ambiente PROGTEST que poderiam desencorajar o seu uso e adoção. Assim, inicialmente, um modelo de qualidade foi desenvolvido considerando as características e subcaracterísticas de qualidade definidas na norma NBR ISO/IEC 9126-1 (ABNT, 2003).

A Figura 3 ilustra o modelo de qualidade proposto. Foram incluídas no modelo as características e subcaracterísticas identificadas como essenciais em ferramentas de apoio à avaliação de trabalhos práticos de programação. A ideia é que tal modelo possa ser utilizado na avaliação de qualquer ferramenta desse tipo e não somente para atender o propósito específico dessa avaliação.

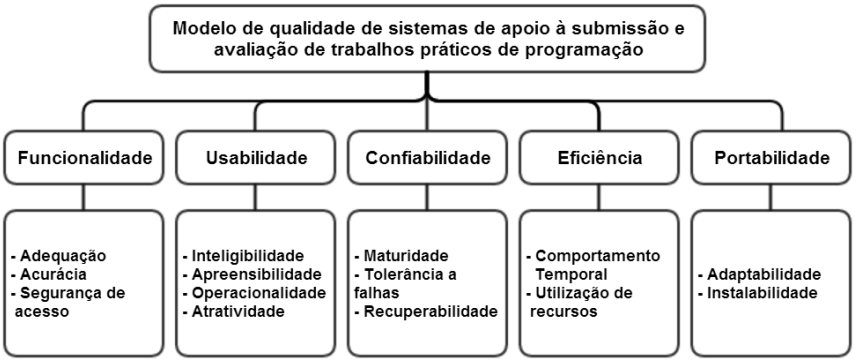


Figura 3. Modelo de Qualidade

3.2. Especificação da Avaliação

A partir do modelo de qualidade estabelecido foram definidas métricas de avaliação relativas a cada subcaracterística de qualidade presente no modelo. A Tabela 1 apresenta a quantidade de métricas definidas para cada subcaracterística, bem como exemplos de métricas associadas a elas.

Tabela 1. Quantidade de Métricas

Característica	Qtd.	Exemplo
Funcionalidade	16	O sistema fornece um *feedback* imediato ao aluno?
Usabilidade	19	O sistema apresenta mensagens de erros claras?
Confiabilidade	10	O sistema permite que o usuário realize o backup dos dados?
Eficiência	2	O sistema responde aos comandos do usuário em um tempo razoável?
Portabilidade	8	O sistema possui um assistente de instalação?

As métricas foram definidas no formato de questões. Assim, foram definidas cinco possíveis respostas a serem atribuídas a cada questão (Likert, 1932). Para cada resposta, um nível de pontuação foi associado. A Tabela 2 mostra as possíveis respostas para cada questão e os níveis de pontuação associados a cada resposta.

Tabela 2. Níveis de Pontuação

Pontuação	Significado
1.00	Totalmente de acordo
0.75	Parcialmente de acordo
0.50	Neutro
0.25	Parcialmente em desacordo
0.00	Totalmente em desacordo

Para cada característica de qualidade, uma medida de qualidade foi calculada. O cálculo consistiu na média aritmética das pontuações das métricas associadas à característica em questão. De forma análoga, a medida final de qualidade do software foi calculada pela média aritmética das medidas obtidas para cada característica.

Por fim, a Tabela 3 apresenta os critérios de julgamento adotados para avaliar a qualidade do software. Em função da medida final obtida, o software pode ser avaliado como “PÉSSIMO”, “RUIM”, “REGULAR”, “BOM” e “EXCELENTE”.

Tabela 3. Critérios de Julgamento

Escala	Julgamentos	
0.9 - 1.0	ÓTIMO	SATISFATÓRIO
0.8 - 0.9		
0.7 - 0.8	BOM	
0.6 - 0.7		
0.5 - 0.6	REGULAR	
0.4 - 0.5		INSATISFATÓRIO
0.3 - 0.4	RUIM	
0.2 - 0.3		
0.1 - 0.2	PÉSSIMO	
0.0 - 0.1		

Conforme observado anteriormente, as métricas, níveis de pontuação e critérios de julgamento definidos para a avaliação da PROGTEST também podem ser utilizadas na avaliação de outras ferramentas de apoio à avaliação de trabalhos de programação.

3.3. Projeto e Execução da Avaliação

Como descrito na Seção 2, a PROGTEST possui as visões de professor e de aluno. Assim, para realizar a avaliação de qualidade do ambiente foram identificadas as tarefas disponíveis aos usuários em cada uma dessas visões. Durante a execução, cada tarefa foi executada pelo menos uma vez a fim de estabelecer o nível de pontuação adequado à cada métrica. As tarefas identificadas são apresentadas na Tabela 4.

Tabela 4. Tarefas do Sistema

Tarefa	Descrição	Usuário
Cadastrar	Permite realizar um cadastro no sistema.	Todos
Autenticar	Permite se autenticar no sistema.	Todos
Editar Perfil	Permite alterar as informações cadastrais.	Todos
Visualizar Cursos	Permite visualizar uma lista de cursos criados.	Professor
Criar Curso	Permite criar um curso no sistema.	Professor
Visualizar Curso	Permite visualizar as informações de um curso.	Professor
Editar Curso	Permite editar as informações de um curso.	Professor
Remover Curso	Permite remover um curso do sistema.	Professor
Cadastrar Aluno	Permite cadastrar um aluno no sistema.	Professor
Adicionar Aluno	Permite associar um aluno à um curso.	Professor
Cria Trabalho	Permite criar um trabalho para um curso.	Professor
Visualizar Trabalho	Permite visualizar as informações e resultados de um trabalho.	Professor
Editar Trabalho	Permite editar as configurações de um trabalho.	Professor
Remover Trabalho	Permite remover um trabalho de um curso.	Professor
Visualizar Trabalhos	Permite visualizar uma lista de trabalhos disponíveis.	Aluno
Submeter Trabalho	Permite submeter uma solução para um trabalho	Aluno
Visualizar Avaliação	Permite visualizar os resultados de uma avaliação.	Aluno

Ressalta-se que o avaliador não possuía nenhum contato prévio com o ambiente PROGTEST. Por outro lado, possuía experiência tanto em Engenharia de Software como em Interação Humano-Computador. Dessa forma, o avaliador pôde ao mesmo tempo atuar como: (1) especialista, identificando não conformidades com relação às características de qualidade; e (2) usuário, percebendo as principais dificuldades, expectativas e satisfação de um aluno ou professor ao interagir com o ambiente.

Nas tarefas que envolveram a submissão e avaliação de programas (“Criar Trabalho” e “Submeter Trabalho”), foi utilizado o programa `Contratação`, que verifica se uma pessoa está apta a trabalhar de acordo com a sua idade. A Figura 4 ilustra a versão oráculo do programa `Contratação`.

```
1. int contrata(int idade) {
2.     if((idade >= 0 && idade < 16) || (idade >= 55 && idade < 100))
3.         return 0; //Não pode trabalhar
4.     else if(idade >= 16 && idade < 18)
5.         return 1; //Trabalhar meio período
6.     else if(idade >= 18 && idade < 55)
7.         return 2; //Trabalha período integral
8.     return -1; //Idade inválida
9. }
```

Figura 4. Oráculo: Programa `Contratação`

Após a execução das tarefas, os resultados foram coletados e analisados. Uma síntese dos principais resultados obtidos e apresentada a seguir.

4. Resultado da Avaliação

A Figura 5(a) apresenta as medidas obtidas para cada característica de qualidade considerada. Como é possível observar, a característica que teve a melhor pontuação foi

a funcionalidade (0,79), enquanto a usabilidade foi a que obteve o resultado mais baixo (0,50). A medida de qualidade final obtida para o ambiente foi de 0,63. Comparando com critérios de julgamento definidos, o ambiente PROGTEST foi classificado, de modo geral, como "BOM" e "SATISFATÓRIO".

Observando cada característica individualmente, três subcaracterísticas de funcionalidade (Figura 5(b)) foram consideradas: (1) adequação, em que foram observadas a presença/ausência de funcionalidades; (2) acurácia, em que foi observado se as funcionalidades executam de forma correta; e (3) segurança, em que foi observado a possibilidade de acesso indevido a dados e funcionalidades. As subcaracterísticas adequação, acurácia e segurança obtiveram como medida de qualidade, respectivamente, os valores 0,60, 0,83 e 0,83. O ambiente PROGTEST foi classificado como "BOM" em todas as subcaracterísticas de funcionalidade, uma vez que todas elas atingiram uma medida maior que 0,6.

Com relação à usabilidade (Figura 5(c)) quatro subcaracterísticas foram consideradas: (1) inteligibilidade, em que foi observado se as funcionalidades da PROGTEST são disponibilizadas de forma clara e intuitiva; (2) apreensibilidade; em que foi observada a facilidade em realizar as tarefas permitidas; (3) operabilidade, em que foi observada a facilidade em manter o controle da PROGTEST; e (4) atratividade, em que foi observado elementos que poderiam atrair potenciais usuários para o ambiente. Assim, o ambiente PROGTEST foi classificado como: (1) "BOM" em atratividade (0,69); (2) "REGULAR" em inteligibilidade (0,55) e apreensibilidade (0,42); e (3) "RUIM" em operabilidade (0,36).

Para confiabilidade (Figura 5(d)), três subcaracterísticas foram consideradas: (1) maturidade, em que foi observada a presença de falhas no ambiente PROGTEST; (2) tolerância a falhas, em que foi observada a capacidade da PROGTEST em continuar operando na ocorrência de uma falha; e (3) recuperabilidade, em que foi observada a capacidade do ambiente em se recuperar e continuar os processamentos em andamento caso ocorra uma queda do sistema. O ambiente PROGTEST foi classificado como "BOM" em maturidade (0,75) e tolerância a falhas (0,75). No entanto, foi considerado "RUIM" em recuperabilidade (0,25).

Com relação à eficiência (Figura 5(e)), duas subcaracterísticas foram consideradas: (1) comportamento temporal, em que foi observado se a PROGTEST apresenta um tempo de resposta razoável durante a realização das tarefas; e (2) utilização de recursos, em que foi observado se a PROGTEST previne a utilização demasiada dos recursos do sistema. O ambiente foi classificado como "BOM" tanto em comportamento temporal (0,75) como em utilização de recursos (0,75).

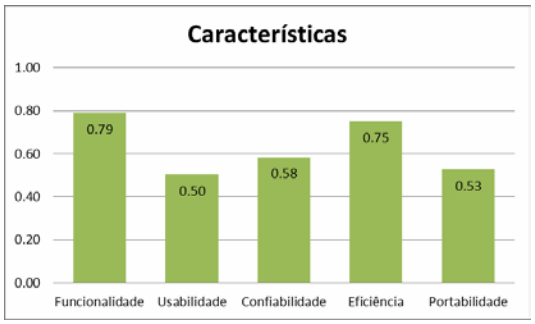
Por fim, duas subcaracterísticas de portabilidade (Figura 5(f)) foram analisadas: (1) adaptabilidade, em que foi observada a utilização da PROGTEST em diferentes navegadores e sistemas; e (2) instalabilidade, em que foram observados os mecanismos disponíveis para auxiliar a instalação da PROGTEST. O ambiente foi classificado como "BOM" em adaptabilidade (0,75) e "RUIM" em instalabilidade (0,30).

5. Sugestões de Melhoria

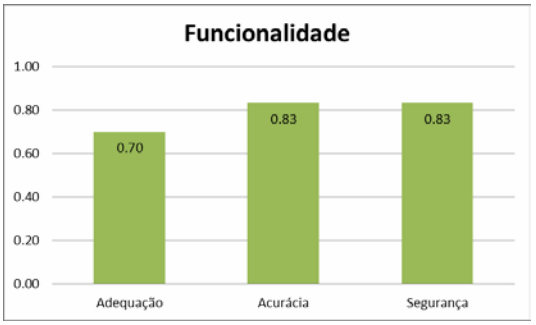
A partir da avaliação do ambiente PROGTEST, algumas sugestões de melhorias puderam ser identificadas, conforme apresentado a seguir:

- **Melhorias no gerenciamento das submissões:** A PROGTEST poderia disponibilizar mais opções de gerenciamento de submissões, como o bloqueio de

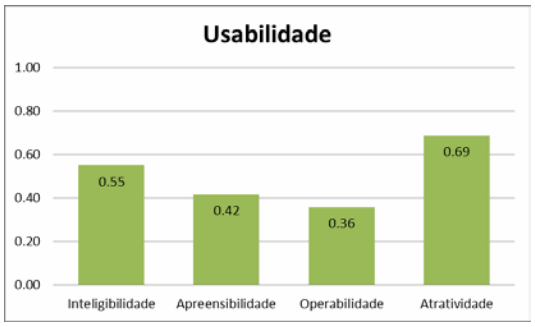
submissões com atraso ou o desconto automático de pontos na nota dos trabalhos com atraso. Atualmente, tal controle é realizado manualmente pelo professor.



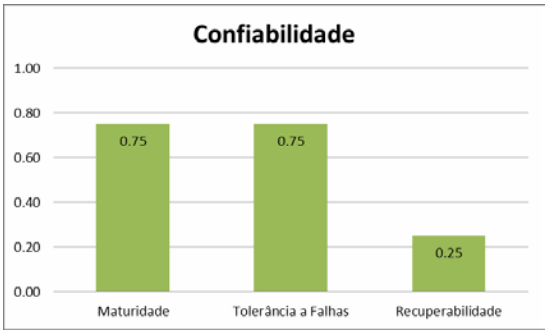
(a) Visão geral das características



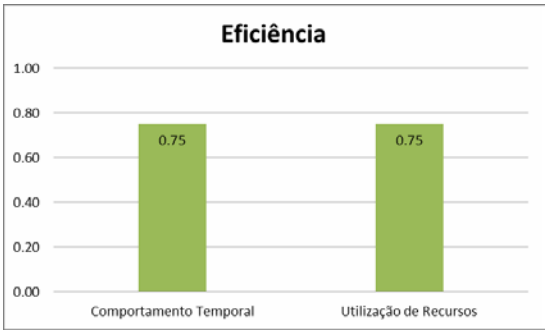
(b) Subcaracterísticas de Funcionalidade



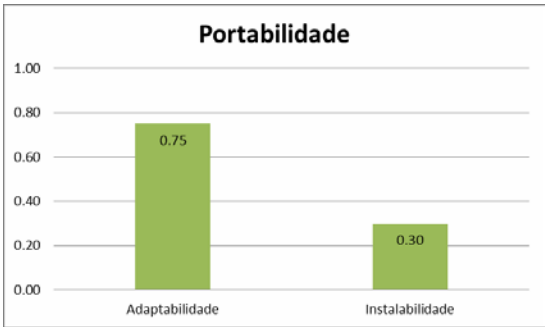
(c) Subcaracterísticas de Usabilidade



(d) Subcaracterísticas de Confiabilidade



(e) Subcaracterísticas de Eficiência



(f) Subcaracterísticas de Portabilidade

Figura 5. Resultados da Avaliação: Subcaracterísticas

- **Melhorias na visualização de notas:** A PROGTEST poderia disponibilizar um quadro geral de notas, em que tanto os professores como os alunos pudessem visualizar em uma mesma página as notas de todos os trabalhos de um curso. Adicionalmente, o cálculo da média de notas dos trabalhos de um curso também poderia ser considerado.
- **Implementação de mecanismos para recuperação de acesso:** A PROGTEST poderia disponibilizar mecanismos que ajudem os usuários a recuperarem as suas informações de acesso (nome de usuário e senha) caso eles venham a esquecê-las.
- **Implementação de uma visão de administrador**: A PROGTEST poderia disponibilizar uma visão de administrador para gerenciar aspectos administrativos do sistema. A visão de administrador poderia incluir funcionalidades como assistente de instalação, gerenciamento de ferramentas de teste, gerenciamento de usuários, dentre outras.

- **Apoio multilíngue:** A PROGTEST poderia disponibilizar apoio aos diferentes idiomas. Atualmente, a sua interface somente está disponível em inglês. Tal apoio poderia incentivar o uso do ambiente por usuários que dominam outros idiomas.
- **Melhorias no *feedback* ao usuário:** A PROGTEST poderia exibir uma mensagem ao usuário confirmando que uma operação foi realizada com sucesso. Ainda, campos obrigatórios em formulários poderiam ser melhor sinalizados.
- **Implementação de mecanismos de busca:** A PROGTEST poderia disponibilizar mecanismos de busca para cursos, trabalhos e alunos. Quando o número de cursos, trabalhos e alunos forem grandes, tais mecanismos facilitariam sua localização.
- **Melhor ordenação das listas:** A PROGTEST poderia disponibilizar mecanismos para ordenar as listas e tabelas de cursos, trabalhos e alunos, também facilitando a localização deles.
- **Ocultação de cursos e trabalhos antigos:** Ainda visando facilitar a localização de cursos e trabalhos, a PROGTEST poderia fornecer a opção de ocultar cursos e trabalhos já encerrados, ressaltando aqueles que se encontram em andamento.
- **Implementação de mecanismos de ajuda:** A PROGTEST poderia disponibilizar melhores mecanismos de ajuda, contribuindo para que os usuários tenham um melhor entendimento de como realizar as tarefas disponíveis, dos resultados da avaliação e dos significados das opções e ferramentas disponíveis.
- **Verificação prévia dos arquivos submetidos:** A PROGTEST poderia realizar uma verificação prévia dos arquivos submetidos (por exemplo, analisando se eles possuem tamanho e formato adequado). Tal verificação poderia prevenir erros durante a execução dos trabalhos e a utilização demasiada de recursos do sistema.

Tais melhorias estão sendo consideradas em uma nova versão da PROGTEST.

6. Conclusões e Trabalhos Futuros

Neste trabalho foi apresentada e discutida a avaliação de qualidade do ambiente PROGTEST. O objetivo da avaliação foi complementar os estudos anteriores que indicaram uma boa efetividade do aprendizado proporcionado pela utilização do ambiente em cenários reais de ensino e aprendizado. Em síntese, a avaliação mostrou que a PROGTEST possui uma boa qualidade. Apesar disso, deficiências principalmente relacionadas à Usabilidade, Portabilidade e Confiabilidade puderam ser identificadas.

A partir dos resultados obtidos, sugestões de melhorias foram propostas para o ambiente PROGTEST, incluindo tanto a melhoria de funcionalidades existentes como a implementação de funcionalidades adicionais para auxiliar na condução de algumas tarefas associadas à avaliação automática de programas. Além disso, ressalta-se que o modelo de qualidade proposto também pode ser utilizado para avaliar outras ferramentas de apoio à submissão e avaliação de trabalhos de programação. Assim, tanto o modelo de qualidade como o processo de avaliação podem ser utilizados como referência para a avaliação de ferramentas similares.

Como trabalhos futuros, pretende-se evoluir o ambiente PROGTEST com base nas sugestões de melhorias propostas. A avaliação e comparação da PROGTEST com outras ferramentas similares também deverão ser investigadas em curto prazo. Por fim, avaliações envolvendo um número maior de usuários e outros tipos de programas também deverão ser conduzidas.

Agradecimentos
Os autores agradecem o apoio financeiro das agências de fomento brasileiras: FAPESP, CAPES e CNPq.

Referências

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **NBR ISO/IEC 14598-1 Tecnologia de Informação – Avaliação de produto de Software – Parte 1: Visão Geral.** 2001.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **NBR ISO/IEC 9126-1 Engenharia de Software – Qualidade de produto de Software – Parte 1: Modelo de Qualidade.** 2003.

BARBOSA, E. F.; Silva, M. A. G.; CORTE, C. K. D.; Maldonado, J. C.. Integrated teaching of programming foundations and software testing. In**: Proceedings of the 38th Annual Frontiers in Education Conference**, p.5-10, 2008.

DOUCE, C.; LIVINGSTONE, D.; ORWELL, J.. Automatic test-based assessment of programming: A review. **J. Educ. Resour. Comput.**, 5(3):1-13, 2005.

EDWARDS, S. H.. Using software testing to move students from trial-and-error to reflection-in-action. In**: 35th SIGCSE Technical Symposium on Computer Science Education**, p.26-30, 2004.

EDWARDS, S. H.; PEREZ-QUINONES, M. A.. Web-CAT: automatically grading programming assignments. In**: Proceedings of the 13th annual conference on Innovation and technology in computer science education**, p.328-328, 2008.

ENGLISH, J.; ROSENTHAL, T.. Evaluating students' programs using automated assessment: a case study. **SIGCSE Bull.**, 41(3):371–371, 2009.

FERNANDEZ-ALEMAN; J. L.. Automated assessment in a programming tools course. **IEEE Transactions on Education**, 54(4):576-581, 2011.

LIKERT, R.. A technique for the measurement of attitudes. **Archives of Psychology**, p.5-55, 1932.

RUBIO-SÁNCHEZ, M.; KINNUNEN, P.; PAREJA-FLORES, C.; VELÁZQUEZ-ITURBIDE, A.. Student perception and usage of an automated programming assessment tool. **Comput. Hum. Behav.**, 31(0): 453-460, 2014.

SOUZA, D. M.; MALDONADO, J. C.; BARBOSA, E. F.. Progtest: An environment for the submission and evaluation of programming assignments based on testing activities. In**: 24th Conference on Software Engineering Education and Training**, p.1-10, 2011.

SOUZA, D. M.; COSTA, S. L.; FREITAS FILHO, N. D.; BARBOSA, E. F.. Um estudo experimental do ambiente PROGTEST no ensino de programação. In: **XVI Ibero-American Conference on Software Engineering – 10th Experimental Software Engineering Latin American Workshop.**, p.1–10, 2013.

SOUZA, D. M.; SOUZA, S. R. S.; OLIVEIRA, B. H.; MALDONADO, J. C.; BARBOSA, E. F.. An automatic test-based assessment system: a contribution for the teaching of software testing. **In 44th Annual Frontiers in Education.**, 2014.

SPACCO, J.; PUGH, W.; AYEWAH, N.; HOVEMEYER, D.. The marmoset project: an automated snapshot, submission, and testing system. In: **Companion to the 21st ACM SIGPLAN symposium on Object-oriented programming systems, languages, and applications**, p.669-670, 2006.