Avaliação Entre-Pares na Disciplina de Programação OO

Vinicius Brandão Araújo[1], Matheus Gaudencio[1]

[1] Universidade Federal de Campina Grande (UFCG)
Caixa Postal 10.106 58429-900 -- Campina Grande, PB

vinicius.brandao.araujo@ccc.ufcg.edu.br

matheusgr@computacao.ufcg.edu.br

Abstract. *Peer review provides good feedback on assignments produced by students. In this work we evaluated how good a guided peer review was in an assignment on a Oriented Object Programming assignment. Most of the students produced a feedback very similar to what a teacher would provide and students were more punitive when evaluating assignments.*

Resumo. *A avaliação entre-pares é um mecanismo eficiente para prover feedback rápido e de qualidade aos estudantes. Neste estudo, aplicamos a avaliação entre-pares em um exercício de programação orientada a objetos. Descobrimos que, com a ajuda de um guia de correção, cerca de 66% dos alunos realizaram uma avaliação próxima a do professor e , na maioria dos casos onde houve divergência, aconteceu pela tendência dos alunos avaliarem de forma mais rigorosa que o professor da disciplina.*

1. Introdução

A resposta dada sobre uma avaliação é parte integrante do processo ensino e aprendizagem. Para minimizar o tempo entre a aplicação de um exercício avaliativo e o *feedback* dessa avaliação, exploramos a aplicação de avaliação entre pares no contexto de programação orientada a objetos.

Para agilizar o tempo de correção, existem alternativas de avaliação automática, como as propostas por Gaudencio et al. (2013). Estas são pertinentes às disciplinas de introdução à programação mas incompletas para a disciplina de programação orientada à objetos, onde aspectos de estruturação de código e design não são facilmente capturados automaticamente. A avaliação entre-pares foi utilizada e descrita por Sirotheau et al. (2011) por enriquecer o *feedback* dado ao aluno. Assim como Sirotheau et al., outras pesquisas de avaliação entre-pares, como aponta França et al. (2015), focam em aspectos gerais e introdutórios de programação.

No nosso trabalho, focamos a avaliação entre-pares de códigos criados em uma disciplina de programação orientada à objetos. A disciplina conta com exercícios avaliativos que exigem a avaliação de critérios como design, funcionalidade e qualidade dos testes produzidos. Na aula imediatamente após o exercício avaliativo, os alunos recebem acesso ao conjunto de exercícios dos colegas bem como a uma diretriz de correção indicando os critérios gerais de avaliação. A diretriz de correção determina os aspectos qualitativos nas quais os alunos devem comentar bem como apresenta instruções para a construção de uma nota somativa para o exercício realizado. Este guia

DOI: 10.5753/cbie.sbie.2018.1955

apresenta critérios subjetivos de avaliação assim cabe a cada aluno ser capaz de reconhecer um bom design, avaliar se o exercício implementa as funcionalidades exigidas e se os testes produzidos têm boa qualidade.

Como estudo inicial, dois dias após a realização de um exercício avaliativo, nós fizemos com que 39 alunos da mesma turma que realizou o exame, realizasse o processo de avaliação entre-pares utilizando a diretriz de correção. Como resultado, cerca de 66% dos alunos que participaram da avaliação atribuem uma nota com, no máximo, 1 de diferença da nota atribuída pelo professor.

2. Avaliação por Alunos em Programação OO

A disciplina prática de Programação 2 na Universidade Federal de Campina Grande têm como objetivos introduzir os alunos os conceitos de Orientação a Objetos e a programação de sistemas simples de informação. Durante cada exercício avaliativo da disciplina, o aluno recebe uma pequena especificação de um projeto (2 páginas) dividida em 3 ou 4 funcionalidades. Os alunos estão livres para criar o código que satisfaz essa especificação com o design que desejarem e devem aplicar conceitos pertinentes à disciplina. Estes exercícios também exploram a construção de testes para as algumas das funcionalidades apresentadas.

Para a realização desta pesquisa, contamos com a participação de 39 graduandos avaliando aleatoriamente um dos exercícios avaliativos realizados na aula anterior. Todos os alunos recebem um guia de correção indicando como avaliar os exercícios dos colegas como mostra a Tabela 1. Existem 3 macro-critérios de avaliação: design, funcionalidades e testes. Cada critério apresenta pequenos exemplos, mas são genéricos e não relacionados a especificação implementada. De posse dessa tabela de correção, o aluno deve descrever textualmente as infrações de design encontradas, as funcionalidades que estão incompletas (e em que aspectos) e quais testes estão ausentes ou apresentam problemas de implementação. Além dessa descrição, o aluno avaliador apresenta uma nota somativa por critério baseando-se na mesma tabela. Estes princípios são aplicados também na correção realizada pelo próprio professor que não teve conhecimento das avaliações realizadas pelos alunos.

Tabela 1. Critérios de avaliação, guia de correção e nota somativa

Critério	O aluno deve identificar...	Cálculo da nota do critério...
Design	- infrações de ocultação da informação - infrações de encapsulamento - infrações do GRASP	10 - 0.5 por infração
Funcionalidade	- funcionalidade funcionando (2.0) - funcionalidade parcialmente implementada (1.0) - funcionalidade não implementada (0.0)	soma dos valores por funcionalidade * (5 / numero de funcionalidades)
Testes	- testes ausentes	10 - 1 por teste ausente

Usamos o momento de avaliação entre-pares como uma oportunidade para que os alunos possam explorar e estudar diferentes soluções e implementações bem como exercitar a capacidade dos alunos de interpretar e avaliar códigos produzidos por outras pessoas.

3. Resultados Parciais

A Figura 1 apresenta o histograma da duração de correção para uma avaliação realizada por cada aluno. Considerando o tempo máximo gasto, de uma hora e quarenta minutos, observamos que o método de avaliação é aplicável no contexto da disciplina, dado o período de 2 horas por aula. Nenhum aluno indicou ter faltado tempo para realizar a avaliação.

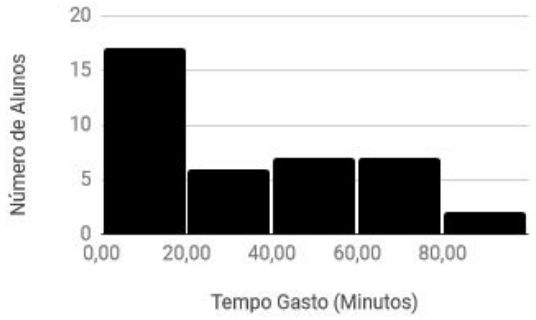

Figura 1. Histograma do tempo gasto da avaliação.

Já a Figura 2, representa a diferença da nota somativa que o professor deu para a avaliação em relação a nota do aluno. Um valor negativo representa que o aluno deu uma nota maior do que a nota do professor. Como resultados, temos que 26 (66%) dos alunos apresentaram até um ponto de diferença da nota do professor. Ainda, nove alunos pontuaram os exercícios de forma mais rigorosa que o professor e acima de um ponto de diferença. Por fim, três alunos pontuaram o exercício de forma menos rigorosa do que o professor.

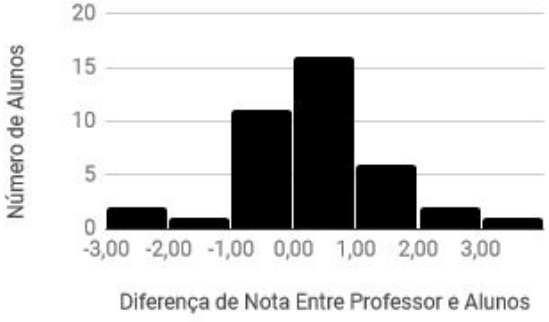

Figura 2. Histograma da diferença da nota do professor e do aluno (b).

Nós procuramos investigar essas 12 situações em que a diferença das notas entre professor e aluno foi maior do que um ponto. Identificamos que os alunos tendem, na maioria das vezes, a punir com mais rigor do que o professor em situações que, apesar de verdadeiras não justificam a necessidade de penalidade. Por exemplo, ao contrário do professor, um dos alunos avaliadores atribuiu nota zero à funcionalidade de um exercício pois o código não compilava por um erro de inicialização de um atributo de uma classe.

Avaliando cada critério em específico, identificamos que a maioria dos exercícios receberam pelos alunos e professor nota entre 8 e 10. Ainda, considerando os critérios avaliados pelos alunos:

- Design: é o aspecto que, apesar de subjetivo, foi mais direto e teve maior proximidade com a correção do professor. A maioria das notas apresentou uma diferença entre -1 e 1 pontos em relação a nota atribuída pelo professor.
- Funcionalidade: observamos que as justificativas dadas pelos alunos avaliadores foram extremamente coerentes e detectamos que nesses quesitos, os alunos atribuem uma penalidade maior do que a apresentada pelo professor. As maioria das notas divergem da nota professor em um valor entre -1,6 à 0,4.
- Testes: assim como o critério de funcionalidade, os alunos apresentaram justificativas coerentes para suas correções, apesar de serem mais rigorosos. A maioria das notas apresentou uma diferença de -1,25 a 2,0 pontos em relação a nota dada pelo professor.

4. Conclusões

A avaliação entre-pares é um mecanismo útil na medida que coloca o aluno no papel de revisar a atividade do colega e de ser capaz de gerar, rapidamente, algum tipo de *feedback*. Neste trabalho, ainda em andamento, descobrimos que a avaliação, dado um guia de correção, não diferiu muito da correção produzida pelo professor mesmo em critérios subjetivos como a avaliação de design na disciplina de programação OO.

Os comentários apresentados pelos alunos apresentam coerência e relevância durante o processo de correção. A nota somativa apresentada pelos alunos se aproxima daquela produzida pelo professor. Em trabalhos futuros pretendemos replicar o estudo em outras avaliações e: i) gerar os pares para a avaliação considerando características do código avaliado e do código produzido pelo avaliador, como, pedir que um aluno avalie o código que seja significativamente mais diferente que o seu; ii) avaliar o impacto da avaliação entre-pares no processo de aprendizagem observando se os alunos alteram características do seu processo de codificação após a realização de exercícios de avaliações entre-pares, e; iii) estudar o processo de auto-avaliação utilizando o guia de correção proposto.

Referências

França, R. S. de, Tesdesco, P. C. de A. R. (2015) "Caracterizando a pesquisa sobre autoavaliação na aprendizagem de programação para iniciantes". Anais do XXVI Simpósio Brasileiro de Informática na Educação, 2015.

Gaudencio, M., Wanderley, L. F., Lemos, F. W., De Araújo, E. C., Figueiredo, J. C. A., Guerrero, D. D. S. (2013) "Eu Sei o que Vocês Fizeram (Agora e) na Aula Passada: o TSTView no Acompanhamento de Exercícios de Programação". Anais do XXIV Simpósio Brasileiro de Informática na Educação, 2013.

Sirotheau, S., Brito, S. R., Silva, A. S., Eliasquevici, M. K., Favero, E. L., e Tavares, O. L. (2011) "Aprendizagem de iniciantes em algoritmos e programação: foco nas competências de autoavaliação. Anais do XXII Simpósio Brasileiro de Informática na Educação, 2011.