Um Ambiente Virtual com Feedback Personalizado para Apoio a Disciplinas de Programação

Fábio P. Alves, Patrícia Jaques

PIPCA – Universidade do Vale do Rio dos Sinos (UNISINOS)

fabiopachecoalves85@gmail.com, pjaques@unisinos.br

Abstract. *For most students, programming courses are challenging and often frustrating. They demand logical knowledge, which can take time to be acquired by the student. For teacher, the high workload can hamper the students monitoring, who always expect to have immediate feedback to their questions. Thus, as a way of supporting programming courses, both for students and teachers, this paper describes a virtual environment composed of resources, such as custom feedback, facilities for interaction between student and teacher and to help teachers to monitor students. This system has undergone a quantitative user experience evaluation with 24 students of a programming course, and had satisfactory results, indicating the system value.*

Resumo. *Disciplinas de programação são desafiadoras e muitas vezes frustrantes para os estudantes universitários. Elas exigem um raciocínio lógico que pode demorar a ser adquirido. Para o professor, a alta carga de trabalho pode prejudicar no acompanhamento dos alunos, que esperam contar com retornos imediatas para suas dúvidas. Como forma de apoiar as disciplinas de programação, tanto para alunos como para professores, este artigo descreve um ambiente virtual composto de recursos como feedback personalizado, facilidades para a interação e auxílio ao professor para acompanhamento dos seus alunos, funcionalidades não encontradas de forma integrada nos trabalhos relacionados. Este sistema passou por uma avaliação quantitativa de experiência de uso por 24 alunos de uma disciplina de programação, e obteve resultados satisfatórios, indicando o valor agregado pelo sistema.*

1. Introdução

Para apoiar o ensino de programação e buscar minimizar dificuldades para atender os alunos, professores buscam recursos e ferramentas computacionais que possam lhes auxiliar nessas tarefas. Uma das ferramentas empregadas são os Juízes *Online*. As plataformas de Juízes *Online* utilizadas em competições de programação são capazes de executar os códigos submetidos a elas e informar se o programa funcionou corretamente ou não. Porém, não possuem o recurso de fornecer um retorno apontando o tipo de erro ocorrido, por que ele aconteceu e nem onde este erro ocorreu. Este recurso é inexistente devido à natureza da sua utilização.

Alguns trabalhos como JOnline [Santos e Ribeiro 2011], MOJO [Chaves *et al* 2013] e BOSS [Joy, Griffiths e Boyatt 2005] estão fazendo uso destes recursos para

apoiar as disciplinas de programação, utilizando a automatização fornecida pelos Juízes *Online* para apresentar mensagens para o aluno de forma instantânea.

Visto estes pontos, este artigo descreve *feeper*, um ambiente virtual para apoiar o ensino/aprendizagem de programação no ensino superior. O trabalho desenvolvido fez uso de um mecanismo para melhorar a comunicação entre aluno e professor, permitindo sinalizar partes do código fonte escrito. Além disso, o *feedback* fornecido pelo Juiz *Online* foi personalizado para apresentar mensagens que indiquem ao aluno onde aconteceu o erro, por que ele aconteceu e como corrigi-lo e para permitir a execução de diferentes tipos de teste no programa do aluno.

Como diferencial, além da correção automática, o trabalho proposto emite dicas sobre os erros gerados pelo programa do aluno, permitindo que este consiga evoluir na correção do seu programa. O aperfeiçoamento da interação entre aluno e professor também foi foco desse trabalho. O programa permite que o aluno possa adicionar marcações em seu programa que serão exibidas ao professor, facilitando o entendimento, de ambas as partes, sobre em qual linha exata do código o aluno está tendo dúvidas e/ou dificuldades.

2. Trabalhos Relacionados

JOnline é um Juiz *Online* didático para o ensino de programação [Santos e Ribeiro 2011]. A proposta deste trabalho foi a de utilizar uma plataforma já existente de Juiz *Online* denominada BOCA [Campos e Ferreira 2004] e agregar a ela novas funcionalidades, entre as quais está a apresentação de dicas em língua portuguesa, programação colaborativa, entre outros recursos.

Outro trabalho interessante proposto é o MOJO, uma ferramenta para auxiliar o professor em disciplinas de programação [Chaves *et al* 2013]. No MOJO o foco é o professor. Ele visa diminuir consideravelmente a sobrecarga de trabalho do docente na correção e apresentação dos resultados das atividades desenvolvidas pelos alunos. Como o foco é no professor, o MOJO recebe a resposta do Juiz *Online* e apresenta para o professor, para que este então dê um *feedback* para os seus alunos.

Lançado em 2005, o BOSS é um sistema de submissão e avaliação de código fonte que suporta a avaliação de trabalhos por meio da coleta de submissões, executando testes automatizados para correção e qualidade, verificando plágios, e fornecendo uma interface para obtenção de *feedbacks* [Joy, Griffiths e Boyatt 2005].

Como diferencial neste trabalho, para a interação entre alunos e professor, foi desenvolvida uma funcionalidade que permite registrar dúvidas e comentários diretamente nas linhas do código fonte do aluno, permitindo que ambos possam visualizar e discutir sobre o código fonte, diferente dos demais trabalhos relacionados que fazem uso de *chats* simples e fóruns. Outro diferencial neste trabalho é a forma como os exercícios são cadastrados pelo professor, que permite atingir um alto nível de detalhamento na correção das respostas submetidas pelos alunos, através do cadastro de classes de teste e também de massa de dados para testes de entrada e saída de informação.

3. O feeper

O *feeper (http://feeper.unisinos.br)* consiste em uma ferramenta *web* para apoiar a aprendizagem de programação em sala de aula, assim como em extraclasse. Na ferramenta, são disponibilizados exercícios práticos que são organizados de forma a desafiar o aluno de forma gradual na resolução e também acompanhar a sua evolução desde o início do uso desta ferramenta.

Para responder aos exercícios, o aluno utiliza um editor de código fonte, podendo criar múltiplas classes e também, caso necessário, realizar o *upload* de classes existentes. Através deste editor, é possível adicionar dúvidas para enviar ao professor, registrar anotações vinculadas às linhas do código fonte, e marcar o código fonte como favorito para uma futura consulta em uma listagem de favoritos.

Durante o desenvolvimento destes exercícios, o aluno pode submeter a sua resposta para validação. Esta validação é feita de forma imediata, utilizando a tecnologia do Juiz *Online* para executar testes automatizados a fim de determinar se a resposta do aluno está correta ou não. Caso ocorram problemas durante esta validação, seja de compilação ou de falha na execução dos testes, o Juiz *Online* emite uma resposta personalizada. Quando um erro de compilação acontece, o Juiz *Online* captura o erro gerado pelo compilador para exibir ao aluno. Quando um erro de execução acontece, uma mensagem personalizada, editada pelo professor, é emitida. Estes testes são divididos em duas partes, onde primeiro é executado o teste caixa preta, inserindo dados conhecidos, coletando as saídas geradas pelo programa e comparando com as saídas conhecidas e esperadas. O segundo modo de testes é a execução de classes de teste, onde para cada classe de teste cadastrada para o exercício é possível emitir uma mensagem personalizada de erro de execução e também uma mensagem personalizada de compilação.

A Figura 1 ilustra o processo executado pelo Juiz *Online*, exibindo as etapas de verificação, validação, execução de testes pré-cadastrados e emissão da resposta final sobre o código fonte do aluno.

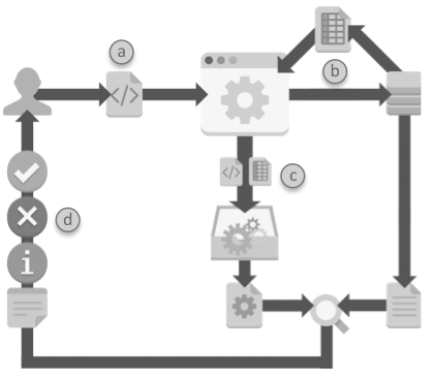

Figura 1. Fluxograma do Juiz *Online*

Figura 2. Tela de resultados com detalhamento de execução

O processo inicia com a submissão do código fonte pelo aluno (conforme item (a) da Figura 1). O Juiz *Online* então solicita ao banco de dados os testes existentes para o exercício que o aluno está tentando responder (b). Os testes e os códigos são reunidos e executados um a um e, a cada execução, o Juiz Online compara a resposta gerada pelo programa do aluno com as respostas esperadas (c). Havendo falha na execução, então

uma mensagem de erro é emitida ao aluno e o erro gerado pelo compilador é exibido. Havendo uma divergência entre a resposta gerada e a resposta esperada, então uma mensagem personalizada de alerta é emitida ao aluno, contendo orientações para que este consiga resolver o problema gerado (d). Essa mensagem de erro deve ser cadastrada pelo professor no seu módulo. Do contrário, então o programa é considerado correto e a resposta de sucesso é emitida ao aluno.

O aluno pode a qualquer momento registrar dúvidas para enviar ao professor. Estas dúvidas são vinculadas a uma linha de um código fonte (conforme linha 10 da Figura 1), permitindo que o aluno sinalize para o professor em qual linha do seu código ele está tendo dificuldades. O professor, ao receber a mensagem, consegue visualizar esta marcação, e responder ao aluno. Este recurso visa facilitar a comunicação entre o aluno e o professor, permitindo também que o professor ao revisar os códigos dos alunos, possa registrar os seus comentários e observações para os alunos.

Para auxiliar o professor, o seu módulo conta com funcionalidades para acompanhar as tentativas dos alunos de solucionar os exercícios, permitindo que o professor interaja com o aluno, registrando comentários nos códigos dos alunos de forma simples e direta. Todas estas mensagens enviadas e recebidas estão disponíveis dentro da caixa de mensagens, desenvolvida como forma de centralizar todas as conversas entre professor e alunos. O professor pode também gerenciar a sua turma, para adicionar novos exercícios e liberá-los gradativamente conforme necessidade. Estes exercícios são mantidos pelo professor, onde ele poderá montar toda a inteligência para a correção do exercício, fornecendo mensagens que irão ajudar seus alunos quando estes encontrarem problemas para resolver o exercício.

Em um painel de resultados, o professor consegue visualizar o andamento dos exercícios, onde através de uma grade que exibe todos os alunos e todos os exercícios da turma é possível visualizar o *status* do exercício para cada aluno, se o estudante já conseguiu resolver o exercício ou se exercício está com algum outro problema. Ao clicar nesta sinalização, a ferramenta apresenta um histórico de respostas submetidas pelo aluno, onde todos os detalhes são exibidos, como a data da submissão, os códigos fontes utilizados e o *feedback* que o sistema apresentou para o aluno. A Figura 2 ilustra a tela de resultados com o detalhamento de execução de determinado exercício para um estudante em específico.

4. Avaliação

Uma avaliação experimental foi realizada em uma turma com 24 alunos de uma disciplina de programação em laboratório do primeiro semestre dos cursos de tecnologia da informação de uma universidade da grande Porto Alegre. O *feeper* foi utilizado pelos alunos e pelo professor durante dois meses. Após este período, para avaliar a experiência no uso da ferramenta, foi aplicado um questionário diferente para alunos e professor. As perguntas foram montadas utilizando a escala de *Likert*, onde os alunos deveriam escolher um valor entre 1 a 5 (discordo totalmente a concordo totalmente). Os dados foram analisados com a medida por moda (*Mo*) ou a resposta mais frequente, assim como a média (μ). Dos 24 alunos que utilizaram a ferramenta, 17 responderam esse questionário final.

O questionário mostrou que as principais funcionalidades obtiveram uma boa aceitação. A funcionalidade para envio (e recebimento) de mensagens ao professor foi

muito bem avaliada, ficando empatada entre muito útil e útil (*Mo*=5; μ=3,7). A grande maioria dos alunos julgou que o retorno fornecido pelo *feeper* após corrigir seus códigos fonte foi útil (*Mo*=4; μ=3,6), porém as mensagens que validam as respostas objetivando ajudar a solucionar o erro teve dois picos na avaliação, no primeiro sendo avaliadas como úteis e no segundo pico avaliadas como um nível abaixo de intermediário (*Mo*=4; μ=3,4). É importante lembrar que as mensagens de *feedback* são cadastradas pelo professor e que, possivelmente, elas não foram tão efetivas pois era a primeira experiência do professor.

5. Conclusão

Este artigo apresentou uma ferramenta *online* para ser utilizada como apoio, para alunos e professores, em disciplinas de programação. Dentre as funcionalidades disponíveis, ganham destaque o recurso de *feedback* personalizado, com base nos cenários de teste cadastrados pelo professor. Esta mensagem é apresentada de forma imediata ao aluno, assim que este submeter sua resposta. Outro recurso importante é a facilidade criada para troca de mensagens entre professor e aluno, a fim de comentar sobre partes específicas dos códigos dos programas dos seus alunos. Quando o estudante manda mensagens para o professor, esta mensagem é vinculada a um trecho de código fonte, o que permite destacar a linha a qual está ocorrendo sua dúvida. Para o professor, a facilidade de acompanhar a sua turma através de um painel de resultados dos exercícios, possibilita que este visualize um histórico de todas as respostas do aluno, com a data da resposta, os códigos utilizados e o *feedback* emitido pelo Juiz *Online*.

Os resultados deste trabalho, que foram obtidos através de uma avaliação da experiência de uso da ferramenta, serviram para identificar se os objetivos foram ou não atingidos. A pesquisa indicou que as principais funcionalidades propostas foram úteis para os avaliados, e que de fato agregaram para a sua experiência de uso. Com a avaliação também foi possível identificar a necessidade de automatizar a criação de cenários de teste, para diminuir o trabalho do professor.

Agradecimentos

O presente trabalho foi realizado com o apoio do CNPq.

Referências

Campos, C. P.; Ferreira, C. E., (2004) "BOCA: um sistema de apoio a competições de programação", Workshop de Educação em Computação, SBC, Salvador, BA.

Chaves, J. O.; Castro, A.; Lima, R.; Lima, M. V.; Ferreira, K., (2013) "MOJO: uma ferramenta para auxiliar o professor em disciplinas de programação", Congresso Brasileiro de Ensino Superior a Distância, Belém, PA.

Joy, M.; Griffiths, N.; Boyatt, R., (2005) "The BOSS Online Submission and Assessment System", ACM Journal on Educational Resources in Computing, v. 5, n. 3, pp. 1–28, setembro 2005.

Santos, J. C. S.; Ribeiro, A. R. L., (2011) "JOnline: proposta preliminar de um juiz online didático para o ensino de programação", XXII Simpósio Brasileiro de Informática na Educação, pp. 964–967, São Cristóvão, SE.