Integrando Moodle e Juízes Online no Apoio a Atividades de Programação

José Osvaldo M. Chaves, Angélica F. Castro, Rommel W. Lima, Marcos Vinicius A. Lima, Karl H. A. Ferreira

Programa de Pós-Graduação em Ciência da Computação (PPgCC)
Universidade do Estado do Rio Grande do Norte (UERN)
Universidade Federal Rural do Semi-Árido (UFERSA)
BR 110 – Km 46 – Bairro Costa e Silva, 59.625-620, Mossoró – RN, Brasil

oswaldo.mesquita@gmail.com, angelica@ufersa.edu.br,
rommelwladimir@uern.br, {marcos.engsoft, karlhansimuller}@gmail.com

Abstract. *This paper presents a tool that integrates Moodle and Online Judges. The tool automates the process of elaboration, submission and evaluation of programming activities, reducing the overhead of the teacher's tasks and giving the necessary support to students. Besides to support in the teaching modality in classroom or distance. The tool allows to carry out activities in various programming languages, among them Java, C and C ++.*

Resumo. *Com o objetivo de contribuir para a melhoria das condições de ensino e aprendizagem de programação, este trabalho apresenta uma ferramenta que integra os Juízes Online ao Moodle. A ferramenta automatiza o processo de elaboração, submissão e avaliação de atividades de programação, diminuindo a sobrecarga de tarefas do professor e dando o suporte necessário aos alunos, podendo, ainda, ser utilizada na educação presencial ou a distância. A ferramenta permite a realização de atividades em várias linguagens de programação, dentre elas Java, C e C++.*

1. Introdução

As Tecnologias de Informação e Comunicação (TIC) têm contribuído de maneira significativa para a ampliação das estratégias de ensino e aprendizagem, proporcionando diferentes ferramentas e artefatos que apoiam o processo de aprendizagem [Amaral *et al*. 2011]. Dessa forma, o sistema educacional tem caminhado rumo a novas possibilidades e sendo enriquecido com novas técnicas que visam um melhor desenvolvimento intelectual dos alunos.

A utilização dessas novas tecnologias tem sido de suma importância para auxiliar professores e alunos. Podem-se citar exemplos de sua utilização em disciplinas de programação, disciplinas que são essenciais em cursos de computação. Essas ferramentas de auxílio aplicadas a estas disciplinas podem facilitar o aprendizado de programação, reduzindo os índices de reprovação e o mau desempenho em outras matérias que tenham programação como pré-requisito.

Os índices de reprovação e o mau desempenho nessas disciplinas são alguns dos problemas que podem provocar a evasão do curso. De acordo com o jornal Folha de São Paulo, que analisou dados do Ministério da Educação (MEC), existe uma alarmante taxa

DOI: 10.5753/CBIE.SBIE.2013.244

de 28%, em média, de evasão nos cursos de Ciência da Computação [Takahashi 2009] nas instituições de ensino superior brasileiras.

Observa-se, nos primeiros semestres dos cursos de graduação em computação, uma quantidade relevante de alunos que reprova, desiste ou obtém um baixo rendimento nas disciplinas que tem foco em programação. Isso é ocasionado, na maioria dos casos, devido à complexidade do processo que é aprender e desenvolver lógica de programação ou à sintaxe da linguagem de programação associada [Mota *et al*. 2009].

No sentido de melhorar esta situação, muitas ferramentas de TIC têm sido propostas para auxiliar o professor no ensino de programação. Um exemplo a ser citado é a utilização dos Ambientes Virtuais de Aprendizagem (AVA), e um forte representante desta categoria é o Moodle [Kumar *et al*. 2011]. Porém, mesmo com o advento dessas ferramentas, algumas barreiras ainda são encontradas pelo professor no ensino de programação, como, por exemplo, a dificuldade de avaliar todos os exercícios de uma turma extensa de alunos em pouco tempo.

No que diz respeito à avaliação de atividades de programação, os sistemas de Juízes *Online* [Zhigang *et al*. 2012] são utilizados em maratonas de programação, e conhecidos por seu amplo repositório de questões e pelo seu processo de avaliação automática de código-fonte. A avaliação feita por estes juízes gera respostas como: certo, errado, erro de compilação, erro em tempo de execução, dentre outras.

Em geral, quer seja na modalidade de Educação a Distância (EaD) ou na modalidade de educação presencial, os sistemas de auxílio existentes atualmente fornecem um ambiente que permite ao aluno criar seus algoritmos e codificá-los em alguma linguagem de programação, porém, é de responsabilidade do professor a elaboração, submissão, avaliação das atividades e fornecimento de *feedback* a todos os seus alunos, o que acaba causando uma sobrecarga de tarefas ao professor.

Visando contribuir com a melhoria do processo de ensino e aprendizagem de programação, este artigo apresenta o Módulo de Integração com os Juízes *Online* (MOJO), uma ferramenta que integra os Juízes *Online* ao Moodle. O ambiente automatiza o processo ESA (Elaboração, Submissão e Avaliação) de atividades de programação propostas pelo professor. O objetivo é fornecer uma ferramenta de suporte que facilite o gerenciamento de recursos e permita realizar o devido acompanhamento dos alunos, além de fornecer o devido *feedback* da atividade.

Para um melhor entendimento, o texto está organizado da seguinte forma: a seção 2 apresenta os ambientes envolvidos na integração, além de citar os trabalhos relacionados. A seção 3 apresenta uma visão da ferramenta e sua arquitetura, e na seção 4 é descrito como ocorre o processo ESA com o MOJO. A seção 5 apresenta os resultados e as discussões sobre estes resultados, e por último, são apresentadas, na seção 6, as considerações finais e o que se pretende realizar em trabalhos futuros.

2. Ambientes da Integração e Trabalhos Correlatos

Nesta seção são descritos os ambientes a serem integrados pela ferramenta desenvolvida, além de enumerar os trabalhos relacionados a ela.

2.1. *Modular Object-Oriented Dynamic Learning Environment* (Moodle)

O Moodle faz parte do grupo de AVAs, estes ambientes permitem, por meio da *Internet*, a produção de conteúdos e canais variados de comunicação, bem como o gerenciamento de dados e controle total de informações. Além disso, possui uma comunidade formada por professores, pesquisadores, e, principalmente, programadores que mantém um Portal na *Web* funcionando como uma central de informações, discussões e colaborações.

O Moodle oferece a professores e alunos um ambiente capaz de reunir a maioria das informações e eventos relevantes, associados a uma disciplina de determinado curso. É importante observar que esta é a plataforma oficial do Ministério da Educação (MEC) para as escolas públicas brasileiras [Martins e Giraffa 2008], e que se mostrou, também, uma importante ferramenta auxiliar no ensino superior, podendo ser utilizada tanto na modalidade de ensino à distância como na modalidade de ensino presencial.

A plataforma conta com uma diversidade de recursos educacionais, permitindo larga flexibilidade para configuração e utilização. Por meio do seu desenvolvimento extremamente modular é possível, ainda, a inclusão de novos recursos que melhor se adaptem às necessidades de quem utiliza o ambiente.

2.2. Juízes *Online*

Os Juízes *Online* são sistemas que compilam, executam e testam códigos-fonte com base em dados padronizados para julgar se estão corretos [Zhigang *et al*. 2012]. Basicamente este processo de avaliação automática ocorre da seguinte maneira: o juiz recebe o código-fonte, durante a execução do código, o juiz utiliza dados formatados como a entrada do programa, processa esses dados e realiza a comparação dos resultados obtidos com os resultados esperados, dando uma resposta apropriada com base nessas comparações (certo, errado, problema de execução, dentre outras). Este método é utilizado em muitos concursos de programação, tais como ACM *Collegiate Programming Contest International* [ACM-ICPC 2013] e *TopCoder* [Inc. 2013].

Os Juízes *Online* podem ser facilmente encontrados na *Internet* e como exemplos podem ser citados o *Timus Online Judge* [Timus Online Judge Team 2013], o SPOJ Brasil [Sphere Research Labs 2013] e o URI *Online Judge* [URI Erechim 2013], estes dois últimos utilizados no desenvolvimento deste trabalho. Nestes sistemas são disponibilizados vários problemas a serem submetidos para resolução. Estes sistemas são tão atraentes que muitos alunos começam a praticar suas habilidades de programação neles [Zhigang *et al*. 2012]. A seguir são apresentadas as principais características dos dois juízes utilizados neste trabalho.

a) SPOJ Brasil

O SPOJ Brasil é a versão brasileira do *Sphere Online Judge* (SPOJ). Suas principais características são: (i) Possui interface amigável; (ii) Disponibiliza fóruns de discussão, ranking de classificação, além de *links* de notícias e informações; (iii) Não disponibiliza funcionalidades via *Web Service*; (iv) Seus problemas suportam várias linguagens de programação, dentre elas C, C++ e Java.

b) URI Online Judge

O URI *Online Judge* também conta com uma versão em português. Suas principais características são: (i) Possui interface amigável; (ii) Disponibiliza fóruns de discussão, ranking de classificação, além de tutoriais e materiais extras; (iii) Possui repositório de questões dividido e classificado por categorias; (iv) Não disponibiliza funcionalidades via *Web Service*; (v) Seus problemas suportam apenas as linguagens C++ e Java.

Nesta primeira versão do MOJO optou-se por utilizar apenas estes dois juízes, devido suas interfaces amigáveis, número de questões e por possuírem versão em português.

2.3. Trabalhos Relacionados

O uso de ferramentas de TIC para dar suporte à educação, mais especificamente em disciplinas de programação, já vem sendo explorado há alguns anos. Esta seção relaciona algumas dessas ferramentas.

Entre os ambientes que fornecem apoio à submissão e avaliação automática de atividades de programação destacam-se a WEB-CAT [Edwards 2004] e a PROGTEST [Souza *et al*. 2012]. WEB-CAT é um ambiente *Web* que visa incentivar o desenvolvimento orientado a testes, apoiando a submissão e avaliação automática de atividades de programação. As ferramentas de teste e de análise são disponibilizadas na WEB-CAT por meio de *plugins*. Cada *plugin* possui critérios de avaliação e relatórios próprios, associados á uma linguagem de programação particular. Dentre as linguagens suportadas, encontram-se *plugins* para Java, C, C++ e Pascal.

A PROGTEST (semelhantemente a WEB-CAT) é um ambiente *Web* automatizado que apoia a submissão e avaliação de atividades práticas de programação, e também se basea em atividades de teste de *software*. Atualmente, dá suporte a apenas duas linguagens de programação (Java e C) e utiliza um programa referência (programa oráculo) que deve ser fornecido pelo professor para avaliação das atividades dos alunos, além de utilizar diferentes ferramentas para testes.

Embora esses trabalhos tragam importantes estudos para auxiliar no ensino de programação, eles não estão integrados a um ambiente que forneça outros recursos educacionais como, por exemplo, suporte a gestão de conteúdo e ferramenta de discussão, importantes recursos, principalmente no que diz respeito a disciplinas ministradas a distância. E em alguns casos específicos, além do professor ter que criar programas referências para auxiliar na correção das atividades, tem-se ainda a limitação de se trabalhar restrito a poucas linguagens de programação.

Uma iniciativa que utiliza Juízes *Online* é a de [Santos e Ribeiro 2011] que propõe o *JOnline* um sistema que possui um Juiz *Online* próprio no qual é possível acessar problemas para serem resolvidos e submeter códigos-fonte para a visualização do resultado. Além da programação colaborativa o sistema ainda adiciona funcionalidades didáticas ao juiz – auxiliando o aluno no processo de aprendizagem. Porém, mesmo com estas vantagens, o *JOnline* também não está integrado a um ambiente como, por exemplo, o Moodle que forneça outros importantes recursos.

Em um contexto semelhante ao abordado neste artigo, algumas iniciativas foram realizadas no sentido de integrar recursos de apoio a disciplinas de programação ao ambiente Moodle, como é o caso da iniciativa de Sirotheau *et al*. (2011), do BOCA-

LAB [França e Soares 2011] e do *Onlinejudge* para Moodle [Onlinejudge 2013]. Em Sirotheau *et al*. (2011), com o objetivo de contribuir para uma melhor compreensão do estudante no aprendizado de programação, a ferramenta *JavaTool* [Mota *et al*. 2009] foi integrada ao Moodle juntamente com o avaliador automático de Moreira e Favero (2009), propiciando uma maneira de visualizar e simular programas no Moodle e permitindo a combinação de técnicas para avaliação da complexidade do código. Desta forma, colaborando para uma melhor avaliação e *feedback* das atividades.

O BOCA-LAB foi desenvolvido no Departamento de Engenharia de Teleinformática (DETI) da Universidade Federal do Ceará (UFC) e surgiu da estensão de um sistema utilizado em maratonas de programação – o BOCA [Campos e Ferreira 2004]. O BOCA-LAB foi integrado ao Moodle por meio de *Web Services*. A ferramenta é capaz de compilar e executar programas escritos em diversas linguagens de programação. Os programas submetidos são então avaliados quanto a erros de compilação e execução em um processo automático.

O *Onlinejudge* para Moodle é composto por *plugins* e também foi desenvolvido para gerenciar a submissão e avaliação de códigos-fontes no Moodle. Ele pode ser integrado a outras duas aplicações: a *Sandbox* [Sandbox 2013] e a *Ideone* [Sphere 2013]. A *Sandbox* se restringe a submissões que executem em C/C++ no ambiente Linux. Já a *Ideone* permite escrever códigos-fonte em aproximadamente 40 linguagens de programação diferentes, sendo executados diretamente a partir do navegador. Entretanto, a *Ideone* é uma aplicação comercial e permite submissão de apenas 2000 códigos-fonte por mês em uma conta gratuita [Zhigang *et al*. 2012]. Sem a integração o *Onlinejudge* suporta apenas as linguagens C e C++.

Todos os trabalhos citados contêm importantes contribuições para o ensino de programação. Porém, mesmo com esse auxílio, o professor ainda sofre com a sobrecarga de atividades no que diz respeito ao processo de elaboração, submissão, avaliação e *feedback* de uma turma extensa de alunos.

Neste artigo, em complemento aos demais trabalhos, é apresentado um ambiente que, integrando os Juízes *Online* ao Moodle, forneça: (1) auxílio necessário ao professor no que diz respeito ao processo ESA de atividades de programação; (2) maior agilidade nas tarefas do professor; (3) acompanhamento de resultados através de uma mesma interface disponível no ambiente virtual; (4) *feedback* mais rápido ao aluno; (5) suporte a diversas linguagens de programação; (6) reuso das atividades.

3. Visão da Ferramenta

Na arquitetura geral de integração entre os ambientes, apresentada na Figura 1, o Moodle fornece a interface e o conjunto de funcionalidades necessárias à gestão e ao acompanhamento das atividades de programação, e o MOJO fica responsável pela interação e comunicação com os Juízes *Online*.

Para garantir a integração entre os dois ambientes (Moodle e Juízes *Online*), a arquitetura interna do MOJO é composta por dois módulos que se comunicam entre si: o Módulo Principal e o Módulo de Carga e Atualização, além de um Repositório de Integração, detalhados nas subseções seguintes.

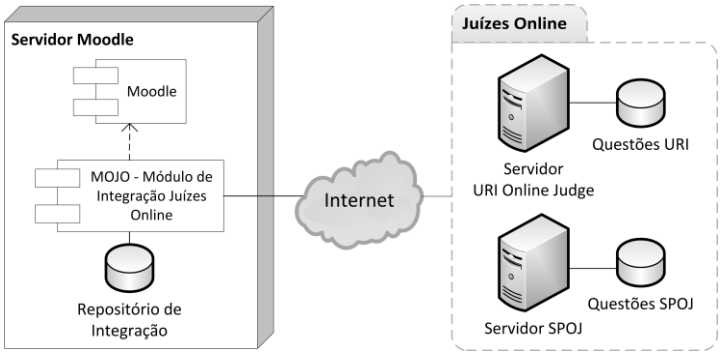

Figura 1. Arquitetura Geral de Integração entre os Ambientes

3.1. Módulo Principal (MOP)

O MOP, como o próprio nome sugere, é o principal módulo da ferramenta, é ele quem gerencia, controla e fornece todas as funcionalidades necessárias ao MOJO para disponibilização das questões e dos resultados no Moodle. É o MOP quem controla as ações do Módulo de Carga e Atualização.

3.2. Módulo de Carga e Atualização (MOCA)

O MOCA é o responsável pela carga de questões no repositório local (Repositório de Integração) da ferramenta e por sua constante atualização. Em um primeiro momento, o MOJO irá fazer uma carga inicial em seu repositório. Esta carga é feita pelo MOCA que interage diretamente, por meio de requisições *Web*, com o repositório dos Juízes *Online*.

O MOCA também monitora o repositório de questões dos juízes a fim de saber se novas atividades foram adicionadas; em caso afirmativo, é realizada a atualização no Repositório de Integração do MOJO.

3.3. Repositório de Integração

O Repositório de Integração consiste da base de dados exclusiva criada para o MOJO, pois foi identificada a necessidade de tabelas específicas para suas funcionalidades. Estas tabelas interagem diretamente com as tabelas do Moodle propiciando a real integração alvo desta pesquisa. É neste repositório onde dados como, por exemplo, as questões dos juízes ficarão armazenadas, assim como também o resultado das mesmas. A Figura 2 ilustra a arquitetura interna do MOJO com todos os seus componentes.

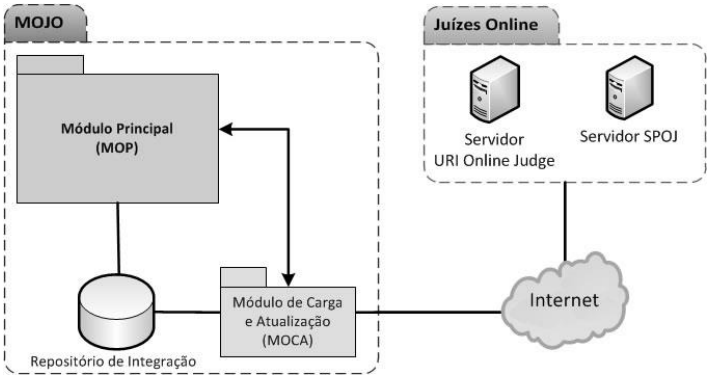

Figura 2. Arquitetura Interna do MOJO

4. Processo de Avaliação com o MOJO

Com o repositório carregado com as questões dos juízes, o MOJO vai propiciar uma forma de colaboração, ainda que indireta, entre o Moodle e os Juízes *Online*, uma vez que estes não interagem diretamente entre si. Neste processo, cada envolvido (professor, aluno e Juiz *Online*) tem um papel diferente e importante. Essa colaboração vai facilitar o processo ESA de atividades de programação da seguinte forma:

a) Facilitando a **Elaboração** de atividades: o professor vai poder selecionar uma questão já existente no repositório do MOJO, sem a necessidade dele mesmo elaborar a questão. Vale ressaltar que ele estará livre da tarefa de elaborar a questão, mas ele ainda precisará analisar as questões e observar qual a melhor a ser aplicada aos seus alunos;

b) Facilitando a **Submissão** de atividades: após o professor selecionar a questão no MOJO, ela automaticamente é submetida para resolução dos alunos no Moodle;

c) Facilitando a **Avaliação** de atividades: após submissão da solução da questão pelo aluno, o MOJO entra em contato automaticamente com o juiz da questão que vai avaliar e retornar o resultado da avaliação.

Para um melhor entendimento, a Figura 3 ilustra o fluxo do processo ESA de atividades de programação com o MOJO.

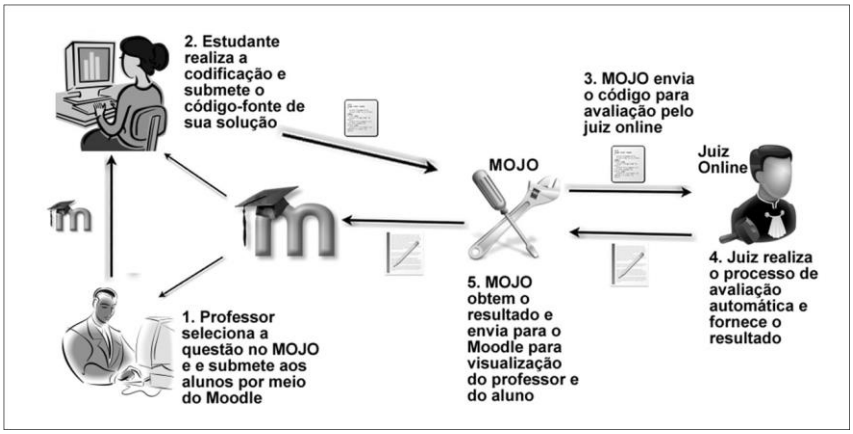

Figura 3. Fluxo do Processo ESA com o MOJO

Conforme a Figura 3, todo o processo se inicia com o professor definindo a questão (selecionada no MOJO) e a submetendo no Moodle para resolução pelos alunos (passo 1). Após submissão da questão, o aluno poderá visualizá-la e preparar sua solução (passo 2).

Após submissão da solução pelo aluno, o MOJO entra em contato com o juiz responsável pela questão e envia o código-fonte para avaliação (passo 3). Em seguida, o Juiz *Online* realiza os devidos processos de avaliação automática para a solução proposta e devolve o resultado da avaliação (passo 4).

No último passo do processo o MOJO obtém o resultado da avaliação e o disponibiliza no Moodle para visualização pelo professor e pelo aluno (passo 5). Com a possibilidade de visualizar os resultados, o professor poderá fazer o devido acompanhamento de seus alunos.

Vale ressaltar que o professor terá acesso aos códigos submetidos pelos alunos, e as atividades ficam armazenadas no repositório local para reutilização no futuro.

4.1. Tecnologias Utilizadas no Desenvolvimento da Ferramenta

Pelo fato de nenhuma API (*Application Programming Interface*) ou serviço *Web* ser fornecido por qualquer um dos juízes utilizados nessa versão do MOJO, foi necessário fazer implementações bem específicas para fornecer um ambiente coeso. Para tais implementações e visando facilitar a integração com a plataforma Moodle, foi utilizada a linguagem de programação PHP [The PHP Group 2013], mesma linguagem na qual o Moodle foi desenvolvido.

Para a base de dados optou-se por utilizar o PostgreSQL [The PostgreSQL Global Development Group 2013], pela característica de ser *open source* (*software* livre). Ainda no que diz respeito à base de dados, percebeu-se a necessidade de criação de novas tabelas para o MOJO, dando origem ao seu Repositório de Integração, com a finalidade de armazenar informações individualizadas dos alunos, permitindo o correto direcionamento do *feedback* gerado pela ferramenta.

5. Resultados e Discussões

Para coleta dos primeiros resultados da ferramenta, foi realizada uma conversa informal com 7 (sete) professores que lecionam alguma disciplina de programação, onde foi apresentado o MOJO e seus principais objetivos. Por meio dessa conversa, foram obtidos os seguintes resultados:

i) 71,4% dos professores afirmaram que a ferramenta pode reduzir o tempo gasto pelo professor no processo ESA de atividades de programação;

ii) 28,6% dos professores sentiram falta da possibilidade de elaborarem suas próprias atividades e as terem avaliadas automaticamente;

iii) 42,8% dos professores fizeram observações sobre a legibilidade do código;

iv) 28,6% dos professores fizeram observações sobre a clareza nos resultados retornados pela ferramenta (resposta dos juízes);

v) 85,7% dos professores concordaram que a ferramenta pode ajudar a fornecer um *feedback* mais rápido ao aluno;

vi) 100% dos professores concordaram que com o a diminuição na sobrecarga de tarefas do professor, será possível fazer o devido acompanhamento dos alunos, principalmente àqueles que sentem mais dificuldades. Neste aspecto todos também concordaram com a criação de um indicador que determine quais alunos sentem mais dificuldades do que outros.

Pode-se perceber, com base nos resultados dessas observações, que o *feedback* dado pela ferramenta deve ser melhorado, assim como deve-se atentar a possíveis problemas de legibilidade do código. Observa-se, ainda, que alguns professores preferem elaborar suas próprias atividades, mas que a avaliação automática destas também pode ser bem útil. A ideia de criar um indicador de dificuldades também deve ser levada em conta em melhorias futuras da ferramenta.

Ainda foram realizados testes em laboratório quanto à velocidade que é dado o resultado das questões pelos juízes, nos testes realizados para ambos os juízes utilizados na ferramenta, o tempo de resposta era de apenas alguns segundos ou até mesmo menos de um segundo (para problemas mais básicos), mas deve-se frisar que foram testes realizados em laboratórios com um número reduzido de usuários.

Para uma melhor avaliação do desempenho da ferramenta junto a professores e alunos, já se trabalha, prioritariamente, no intuito de disponibilizar o MOJO em uma turma regular, com o objetivo de verificar a aplicação prática da ferramenta e possíveis ajustes a serem realizados.

6. Considerações Finais e Trabalhos Futuros

A integração dos Juízes *Online* com o Moodle, por meio do MOJO, tem como objetivo diminuir a sobrecarga de trabalho do professor no processo ESA de atividades de programação. Como resultado, espera-se a melhoria na qualidade do ensino e aprendizagem de programação, tendo em vista que o tempo do professor com outras tarefas pode ser reduzido, e com isso ele poderá fazer uso desse ganho de tempo para realizar um melhor acompanhamento de seus alunos.

Além disso, sabe-se que outros fatores podem ser responsáveis pelas reprovações e desistências nos cursos, porém, considera-se que, com o aumento da disponibilidade de tempo, o professor poderá atuar de maneira a evitar o desestímulo ou eventual sensação de abandono por parte do aluno, evitando, assim, a evasão do curso.

Como trabalhos futuros pretende-se integrar ao MOJO um Laboratório Virtual de Programação, a exemplo do VPL [VPL 2013], que além de auxiliar o aluno na edição e avaliação de seus códigos, também permitirá ao professor elaborar suas próprias atividades e avaliá-las automaticamente. Com este laboratório virtual também se pretende melhorar os problemas de legibilidade e do *feedback* retornado na avaliação da atividade. A inclusão de outros Juízes *Online* também está prevista, ampliando ainda mais o repositório de questões do MOJO.

Referências

ACM-ICPC. (2013) "*The ACM-ICPC International Collegiate Programming Contest*". Disponível em: <http://icpc.baylor.edu/>. Acesso em 20 de jul de 2013.

Amaral E. M. H; Ávila, B; Zednik, H; Tarouco, L. (2011) "Laboratório Virtual de Aprendizagem: Uma Proposta Taxonômica". In: RENOTE. Revista Novas Tecnologias na Educação, v. 9 n. 2.

Campos, C. P; Ferreira, C. E. (2004) "BOCA: Um sistema de apoio para competições de programação". In: Workshop de Educação em Computação, Anais da Sociedade Brasileira de Computação, Salvador-BA, 2004.

Edwards, S. H. (2004) "*Using software testing to move students from trial-and-error to reflection-in-action*". SIGCSE Bulletin, v. 36, n. 1, março 2004, pp 26–30.

França, A. B; Soares, J. M. (2011) "Sistema de apoio a atividades de laboratório de programação via Moodle com suporte ao balanceamento de carga". In: Anais do XXII Simpósio Brasileiro de Informática na Educação, Aracaju-SE, 2011.

Inc., T. (2013) "TopCoder". Disponível em: <http://www.topcoder.com/>. Acesso em 20 de jul de 2013.

Kumar S; Gankotiya, A. K; Dutta, K. (2011) "*A Comparative Study of Moodle with other e-Learning Systems*". In: International Conference on Eletronics Computer Technology – ICECT, 3. Kanyakumari: IEEE, 2011.

Martins, C; Giraffa, L. M. M. (2008) "Capacit@ndo: uma proposta de formação docente utilizando o Moodle". In: RENOTE. Revista Novas Tecnologias na Educação, v. 6, n. 1, pp 1-8.

Moreira, M. P; Favero, E. L. (2009) "Um Ambiente Para Ensino de Programação com Feedback Automático de Exercícios". In: Workshop Sobre Educação em Computação, Anais da Sociedade Brasileira de Computação. Belém-PA, 2009.

Mota, M. P; Pereira, L. W. K; Favero, E. L. (2009) "*JavaTool*: Uma Ferramenta Para Ensino de Programação". In: Anais do XX Simpósio Brasileiro de Informática na Educação. Florianópolis-SC, 2009.

Onlinejudge. (2013) "*Onlinejudge*". Disponível em: https://github.com/hit-moodle/onlinejudge. Acesso em 21 de jul de 2013.

Sandbox. (2013) "*Sandbox*". Disponível em: https://github.com/openjudge/sandbox. Acesso em 21 de jul de 2013.

Santos, J. C. S; Ribeiro, A. R. L. (2011) "*JOnline*: proposta preliminar de um juiz online didático para o ensino de programação". In: Anais do XXII Simpósio Brasileiro de Informática na Educação. Aracaju-SE, 2011.

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

Souza, D. M; Maldonado, J. C; Barbosa, E. F. (2012) "Aspectos de Desenvolvimento e Evolução de um Ambiente de Apoio ao Ensino de Programação e Teste de Software". In: Anais do XXIII Simpósio Brasileiro de Informática na Educação, Rio de Janeiro-RJ, 2012.

Sphere Research Labs. (2013) "IDE ONE". Disponível em <http://ideone.com/>. Acesso em 22 de jul de 2013.

Sphere Research Labs. (2013) "SPOJ Brasil". Disponível em <http://br.spoj.pl/>. Acesso em 16 de jul de 2013.

Takahashi, F. (2009) "Matemática e ciências da computação têm alta taxa de abandono". In: Folha de São Paulo. São Paulo, 06 abr. 2009. Disponível em: http://www1.folha.uol.com.br/folha/educacao/ult305u546576.shtml. Acesso em 12 de jul de 2013.

The PHP Group. (2013) "PHP: *Hypertext Preprocessor*". Disponível em: <http://www.php.net>. Acesso em 17 jul de 2013.

The PostgreSQL Global Development Group. (2013) "PostgreSQL". Disponível em: <http://www.postgresql.org>. Acesso em 17 jul de 2013.

Timus Online Judge Team. (2013) "*Timus Online Judge*". Disponível em: <http://acm.timus.ru/>. Acesso em 21 de jul de 2013.

URI Erechim. (2013) "*URI Online Judge*". Disponível em: <http://www.urionlinejudge.com.br>. Acesso em 16 de jul de 2013.

VPL. (2013) "*Virtual Programming Lab*". Disponível em: <http://vpl.dis.ulpgc.es/ >. Acesso em 21 de jul de 2013.

Zhigang, S; Xiaohong, S; Ning, Z; Yanyu, C. (2012) "*Moodle Plugins for Highly Efficient Programmin Courses*". In: Proceeding of 1st Moodle Research Conference, Heraklion, Crete-Greece, 2012.