Uma Ferramenta para Auxiliar na Elaboração, Submissão e Correção de Atividades Práticas em Disciplinas de Programação

José Osvaldo M. Chaves[1], Angélica F. Castro[2], Rommel W. Lima[1], Marcos Vinicius A. Lima[1], Karl H. A. Ferreira[1]

[1]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)
Laboratório de Redes e Sistemas Distribuídos (LORDI)
BR 110 – Km 46 – Bairro Costa e Silva, 59.625-620, Mossoró – RN, Brasil

[2]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)
Departamento de Ciências Exatas e Naturais (DCEN)
BR 110 – Km 47 – Bairro Costa e Silva, 59.625-900, Mossoró – RN, Brasil

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

Abstract. *Disciplines that require a lot of programming needs an involved teacher who often cannot perform efficient monitoring of the student. The long wait for the student to ask questions or present their results are elements that don't contribute to motivation. Due to the overload of activities the teacher, the use of automated tools support monitoring becomes an interesting alternative. Thus, in order to contribute to the improvement of teaching and learning in programming disciplines, this article proposes to automate the process of preparing, submitting programming practices and their evaluations based on a process similar to that used in programming marathons, presenting an environment that integrated the Moodle supports programming teaching both in person and distance.*

Resumo. *Disciplinas de programação exigem um grande envolvimento do professor que, muitas vezes, não consegue realizar um acompanhamento eficiente do aluno. A longa espera do aluno para tirar dúvidas ou para apresentar seus resultados são elementos que podem contribuir para a desmotivação do mesmo. Devido à sobrecarga de atividades do professor, o uso de ferramentas automatizadas de apoio ao acompanhamento se torna uma boa alternativa. Dessa maneira, visando contribuir com a melhoria das condições de ensino e aprendizagem em disciplinas de programação, este artigo propõe a automatização do processo de elaboração, submissão de atividades práticas de programação e de suas avaliações com base num processo semelhante ao adotado em maratonas de programação, apresentando um ambiente que, integrado, ao Moodle apoie o ensino de programação tanto presencialmente como à distância.*

1. Introdução

A disciplina de programação é uma das disciplinas essenciais, principalmente, aos estudantes de computação, pois constitui a base para muitas áreas em que a informática pode ser aplicada. Um bom aprendizado dessa disciplina torna o indivíduo apto a utilizar a lógica de programação na resolução de diversos problemas, fator importante em disciplinas mais avançadas.

As dificuldades encontradas no aprendizado de programação refletem em altos índices de reprovação e consequentemente em mau desempenho do aluno em outras matérias que têm programação como base. Este é um grave problema enfrentado pelas instituições de ensino superior no Brasil. Para ter uma ideia da gravidade do problema, segundo dados do Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira (INEP), entre os anos de 2001 a 2005 os cursos de graduação na área de computação apresentaram índice médio de 28% de evasão e os cursos de ciência da computação atingiram 32% [Silva Filho *et al.* 2007].

Durante os primeiros anos dos cursos de graduação em computação, por exemplo, é observada uma quantidade bem relevante de alunos que reprovam, desistem ou obtém um baixo rendimento nas disciplinas que focam o estudo de programação. Isso é ocasionado, na maioria dos casos, devido ao processo complexo e exigente que é aprender e desenvolver lógica de programação [Ferradin e Stephani 2005].

A dificuldade para se aprender programação pode ser consequência de vários fatores, tais como, por exemplo, uma fraca base matemática, dificuldades para a compreensão do problema e entendimento do assunto. Segundo Prior (2003), a habilidade de se programar computadores não pode ser adquirida sem um significante esforço em atividades práticas de laboratório.

Neste sentido, muitas ferramentas foram propostas para auxiliar o professor no ensino de programação, como, por exemplo, em Moreira e Favero (2009). Porém, mesmo com o advento dessas ferramentas, algumas barreiras ainda são encontradas pelo professor nesse processo, como por exemplo, a dificuldade de avaliar todos os exercícios de uma turma extensa em pouco tempo.

Em geral, quer seja na modalidade de Educação a Distância (EaD) ou na modalidade de educação presencial, os sistemas existentes atualmente fornecem um ambiente que permite ao aluno criar seus algoritmos e codificá-los em alguma linguagem de programação, porém, para o professor torna-se difícil fornecer um *feedback* rápido e disponibilizar as devidas correções a seus alunos.

Um ambiente muito comum em competições de programação são os chamados Juízes *Online* [Kurnia *et al.* 2001], que tem como principal função a avaliação de códigos-fonte. A avaliação feita por esses juízes gera respostas como: certo, errado, saída mal formatada, erro de compilação, erro em tempo de execução, dentre outras [Campos e Ferreira 2004].

As ferramentas de auxílio existentes somadas às pesquisas na área contribuem não apenas para minimizar os problemas de evasão e dificuldade do aprendizado de programação, como também podem melhorar a qualidade do processo de ensino.

Contudo, muito ainda pode ser feito para que as inovações aconteçam com mais qualidade e credibilidade.

No cenário da tecnologia educacional, no que diz respeito ao ensino da disciplina de programação, os ambientes de auxílio existentes não são completos. Entretanto, é possível fazer a integração de dois ou mais ambientes distintos de uma maneira complementar para um propósito comum. Desta forma, fazendo surgir um novo sistema, mas é nessa integração que se encontra um grande desafio e embora seja um processo mais complexo, este é o cenário encontrado mais comumente.

O texto está organizado da seguinte forma: a seção 2 aborda os trabalhos relacionados, apresentando soluções que buscam auxiliar o ensino de disciplinas que envolvam práticas de programação. A seção 3 apresenta as características do Moodle e porque ele é uma boa opção para fazer a integração com outros ambientes. Na seção 4 são apresentados os Juízes *Online*, suas características e alguns exemplos de juízes. A seção 5 é responsável por mostrar a problemática e a justificativa que impulsionaram o desenvolvimento da pesquisa. A seção 6 apresenta e descreve o módulo e a arquitetura de integração e, por último, são apresentadas, na seção 7, as considerações finais abordando o que se espera como resultados e o que se pretende realizar em trabalhos futuros.

2. Trabalhos Relacionados

O uso de ambientes virtuais para dar suporte à educação, mais especificamente às atividades práticas de programação, vem sendo explorado há alguns anos. Em um contexto aproximado à pesquisa apresentada neste artigo, algumas iniciativas foram realizadas no sentido de integrar recursos de apoio a disciplinas de programação ao Moodle [MOODLE 2013], como o BOCA-LAB [França e Soares 2011] e a iniciativa de Sirotheau *et al.* (2011).

O BOCA-LAB foi desenvolvido no Departamento de Engenharia de Teleinformática (DETI) da Universidade Federal do Ceará (UFC) e surgiu da adaptação de um sistema utilizado em maratonas de programação – o BOCA [Campos e Ferreira 2004]. O BOCA-LAB foi integrado ao Ambiente Virtual de Aprendizagem (AVA) Moodle, onde o Moodle forneceu a interface e o conjunto de funcionalidades necessárias à gestão e ao acompanhamento das atividades associadas ao laboratório de programação. A integração dos dois ambientes foi realizada por meio de *Web Services* (WS). 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.

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.* 2008], que propicia uma maneira de visualizar e simular programas, também foi integrada ao Moodle juntamente com o avaliador automático de Moreira e Favero (2009), permitindo a combinação de algumas técnicas para avaliação da complexidade do código. Desta forma, colaborando para uma melhor avaliação e *feedback* das atividades.

Uma importante iniciativa é a de Souza *et al.* (2012), que mostra a evolução da PROGTEST [Souza *et al.* 2011], um ambiente *Web* automatizado que apoia a submissão e avaliação de trabalhos práticos de programação, baseada em atividades de teste de *software*. A PROGTEST, atualmente, dá suporte a apenas duas linguagens de programação (Java e C) e utiliza um programa referência (programa oráculo) fornecido pelo professor para avaliação dos trabalhos dos alunos [Souza *et al.* 2012], além de utilizar diferentes ferramentas para testes, tais como JUnit [Beck e Gamma 2010] e CUnit [CUnit Project 2012].

Embora todos os trabalhos aqui citados contenham importantes contribuições para auxiliar no ensino das disciplinas de programação, eles ainda exigem que o professor gaste certo tempo para a elaboração das atividades que serão submetidas aos alunos, ou seja, por mais auxílio que o professor tenha com essas ferramentas, ele ainda teria que dedicar uma boa parte de seu tempo para idealizar tais atividades. E em alguns casos específicos, como é o caso da ferramenta PROGTEST, além do professor ter que criar programas referências para auxiliar na correção das questões, tem-se ainda a limitação de se trabalhar restrito a poucas linguagens de programação.

Neste artigo, em complemento aos trabalhos aqui relacionados, é proposto um ambiente que forneça o auxílio necessário ao professor no que diz respeito à elaboração, submissão e correção de atividades práticas de programação, resultando em maior agilidade nas atividades do professor, um ganho de tempo na elaboração das questões submetidas aos alunos e um *feedback* mais rápido ao aluno. Desta forma, a ferramenta propõe melhorar o ensino e aprendizagem de disciplinas de programação, pois o professor poderá utilizar-se do ganho de tempo para dar uma maior atenção aos seus alunos.

3. *Modular Object Oriented Distance Learning* (Moodle)

O Moodle foi desenvolvido pelo australiano Martin Dougiamas em 1999, possui tradução para mais de 40 idiomas, e é classificado como um Ambiente Virtual de Aprendizagem (AVA) de *software* livre, ou seja, pode ser baixado, utilizado e/ou modificado por qualquer indivíduo em todo o mundo [Alves e Brito 2005].

O Moodle tem uma comunidade de usuários colaborativa e conta, atualmente, com mais de um milhão de participantes espalhados por mais de 200 países, inclusive no Brasil. Essa comunidade, formada por professores, pesquisadores, administradores de sistema, designers instrucionais e, principalmente, programadores, mantém um Portal na *Web* que funciona como uma central de informações, discussões e colaborações [Pulino 2004].

A plataforma em si é diversificada em recursos educacionais e permite larga flexibilidade para configuração e utilização. O seu desenvolvimento extremamente modular permite a fácil inclusão de novos recursos que melhor o adaptem às reais necessidades de quem o utiliza. Vale ressaltar que o Moodle é a plataforma oficial do Ministério da Educação (MEC) para as escolas públicas brasileiras [Martins e Giraffa 2008], podendo ser utilizada tanto na modalidade de ensino à distância como na modalidade de ensino presencial.

Neste contexto, a ferramenta oferece a professores e alunos um ambiente capaz de reunir a maioria das informações e eventos relevantes, associados a uma disciplina. O grande potencial oferecido para a criação de novas funcionalidades e sua ampla utilização justificam a integração do Moodle com outras ferramentas.

4. Juízes *Online*

A maioria dos programas de natureza algorítmica necessita apenas obter como entrada um padrão de dados devidamente formatados e, a partir desses dados, realizar o devido processamento. Após o processamento, os resultados são apresentados de maneira formatada em uma saída padronizada. Dessa maneira, é possível que a avaliação de programas seja feita automaticamente utilizando uma ferramenta que gere os dados de entrada e outra que obtenha e verifique os resultados obtidos [Kurnia *et al.* 2001].

O processo de avaliação automática é feito pelos Juízes *Online*. Estes sistemas recebem o código-fonte enviado pelo usuário e posteriormente compilam e executam esse código. Durante a execução do programa, os Juízes *Online* utilizam dados formatados como a entrada do programa, processam esses dados e realizam a comparação dos resultados obtidos com os resultados esperados.

Os Juízes *Online* são muito utilizados em maratonas de programação e podem ser encontrados na Internet, como exemplos podem ser citados o SPOJ Brasil [Sphere Research Labs 2012] e URI *Online Judge* [URI Erechim 2012]. Nesses sistemas são disponibilizados vários problemas a serem resolvidos e submetidos. Dessa maneira, um usuário seleciona a linguagem de programação a ser utilizada na escrita do código e envia a sua solução do problema para ser avaliada. Além disso, também são disponibilizados fóruns de discussão, ranking de usuários e algumas informações em forma de estatísticas para cada problema (por exemplo, total de submissões e a quantidade de pessoas que resolveram o problema)**.**

5. Problemática e Justificativa

Conforme dito anteriormente, na introdução deste artigo, no que diz respeito ao ensino de programação, as ferramentas existentes não promovem um auxílio completo a professores e alunos. Uma opção para minimizar este problema é fazer a integração entre dois ou mais ambientes de maneira complementar, visando obter um ambiente coeso e completo para este fim. Porém, é nesta integração que reside um complexo desafio computacional, pois muitas vezes é necessário trabalhar com, por exemplo, tecnologias e linguagens de programação diferentes.

O que se tem observado é que, em um contexto geral, as ferramentas de auxílio existentes já são produzidas para um fim específico, sem que sejam levadas em consideração futuras integrações entre ambientes ou a expansão de suas funcionalidades. Mesmo nos Ambientes Virtuais de Aprendizagem atuais, que apresentam um conjunto de ferramentas de propósito geral e podem ser empregados para diversos cursos, esses ambientes raramente são concebidos com a perspectiva de extensão ou de integração com outras plataformas. Uma exceção a este modelo é o Moodle que possui documentação específica para a agregação de novas funcionalidades. Um forte argumento, se não o maior, que apoia a integração entre os Juízes *Online* e o Moodle é o

fato de evitar a reimplementação de todos os recursos de gerenciamento de disciplinas e cursos [Ihantola *et al.* 2010].

5.1. Elaboração de Práticas

Um dos desafios mais cansativos, e demorados, enfrentados por um professor que ministra alguma disciplina que envolva prática de programação é a elaboração de atividades práticas para seus alunos, pois o professor muitas vezes trabalha com mais de uma turma e cada turma contendo vários alunos e, além disso, ainda precisa gerenciar seu tempo para correção de trabalhos, provas, seminários, dentre outras tarefas.

Neste contexto, a utilização dos Juízes *Online* se mostra uma alternativa válida, pois esses sistemas contam com uma base de dados de questões pré-definidas com toda informação necessária à sua realização, além de procedimentos específicos para a avaliação dos códigos submetidos para cada questão. Desta maneira, o professor contaria com o auxílio dos Juízes *Online* para a submissão das questões e para suas respectivas avaliações. Segundo Carter *et al.* (2003), uma ferramenta que automatize esse processo poupa tempo do professor, que poderá utilizar este tempo para realizar atividades que não podem ser automatizadas.

6. Módulo e Arquitetura de Integração

Conforme mostrado na seção anterior, a elaboração das práticas pode se tornar um desafio para o professor, e com o auxílio dos Juízes *Online* esse problema pode ser resolvido.

Neste sentido, este artigo apresenta uma ferramenta que funciona como um novo módulo integrado ao Moodle para auxiliar o professor no processo de elaboração, submissão de questões e no rápido *feedback* sobre o resultado das mesmas, proporcionando um ambiente coeso onde estejam integrados o Moodle e os Juízes *Online*. Para um melhor entendimento, a seguir é mostrada uma figura contendo a arquitetura do ambiente.

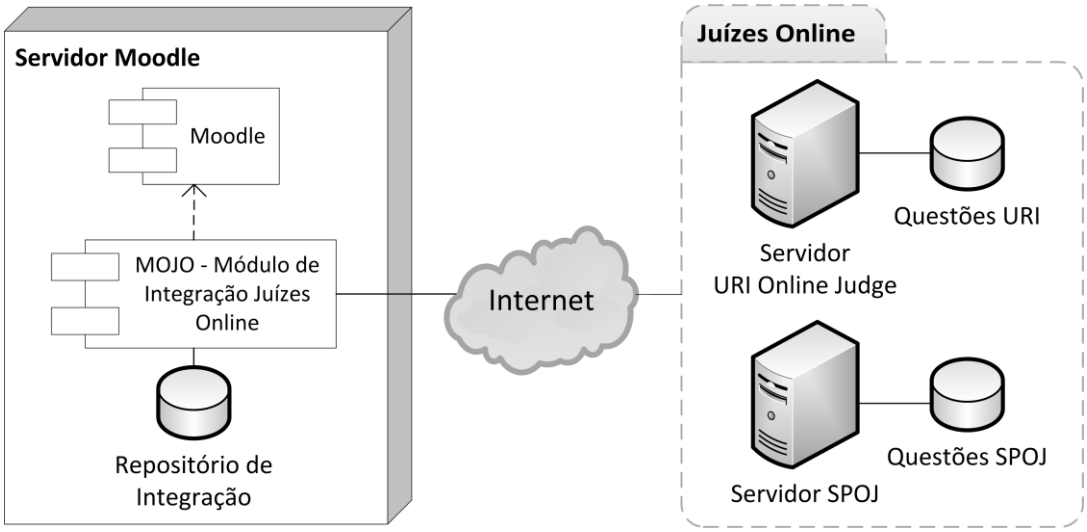

Figura 1. Arquitetura do ambiente

Conforme mostrado na Figura 1, observa-se que de um lado temos os Juízes *Online* em seus respectivos servidores, com seus respectivos repositórios de questões. Do outro lado, no servidor Moodle, temos o módulo de integração com os juízes e que conta com seu próprio repositório local de questões.

6.1. Módulo de Integração com os Juízes *Online* (MOJO)

O MOJO é a ferramenta propriamente dita (encapsulada em um módulo) responsável pela integração, que vai fazer o meio termo entre o Moodle e os Juízes *Online*. Ele é o responsável pela comunicação entre o Moodle e os Juízes *Online* envolvidos nas operações.

Em um primeiro momento, o MOJO irá fazer uma carga inicial em seu repositório local de questões. O módulo fará isso por meio de requisições *Web* onde serão obtidas as questões de cada um dos Juízes *Online*. Com as questões armazenadas em sua base de dados, o MOJO fornecerá as informações necessárias para utilizar esses dados como atividades no Moodle.

Com o módulo carregado com as questões dos juízes o professor poderá visualizá-las em tela e selecionar a questão apropriada para submeter a seus alunos. E para a correção das soluções enviadas pelos alunos, O MOJO fará nova requisição *Web* ao juiz responsável pela questão, enviando o código do aluno e recuperando o resultado da avaliação feita pelo juiz. Por fim, disponibilizando este resultado no Moodle.

A seção seguinte descreve em um passo-a-passo como será o funcionamento da ferramenta, após a carga inicial do módulo.

6.2. Funcionamento da Ferramenta

O funcionamento da ferramenta ocorre de forma distinta para cada um dos diferentes envolvidos no processo: o professor, o aluno e o juiz o*nline*. Este funcionamento é explicado a seguir.

- Para o professor:
  - *i)* O professor, por meio do Moodle, terá acesso ao módulo de integração com os Juízes *Online*, onde ele encontrará as diversas questões, dos juízes utilizados, em um repositório local que anteriormente foi carregado com as questões de cada juiz.
  - *ii)* O professor então seleciona a questão contendo em sua descrição todas as informações necessárias à sua realização, e a submete para resolução pelos alunos.
- Para o aluno:
  - *i)* O aluno irá acessar o Moodle e verificar a existência de novas questões.
  - *ii)* Existindo uma nova questão, o aluno realiza a codificação do problema, e depois de terminada, envia sua resposta para avaliação.
- Para o juiz *online* responsável pela questão:

i) Após a submissão da resposta do aluno, o módulo de integração entra em contato com o juiz *online*, o qual a questão respondida pertence, e envia a resposta para avaliação.

ii) O juiz *online* responsável pela questão realiza os devidos processos de avaliação para a questão, e devolve o resultado da avaliação realizada.

Em posse do resultado da avaliação o professor poderá dar o devido *feedback* ao aluno. É importante lembrar que o professor terá acesso aos códigos submetidos pelos alunos para consulta e que as questões ficarão armazenadas no repositório local para posterior reutilização

7. Considerações Finais

A integração dos Juízes *Online* com o Moodle visa diminuir consideravelmente a sobrecarga de trabalho na correção de códigos-fonte por parte dos professores, bem como reduzir o tempo necessário para correção e apresentação dos resultados das atividades desenvolvidas pelos alunos. Como resultado, espera-se a melhoria na qualidade do ensino e aprendizagem das disciplinas de programação, tendo em vista que, o tempo do professor com atividades de administração e de gestão de recursos pode ser reduzido e, com isso, espera-se uma maior disponibilidade para dar uma maior atenção ao aluno.

O professor ficará livre da criação de questões caso ele opte por utilizar uma das questões pré-definidas no juiz *online*, a proposta é exatamente esta: o professor verificará no juiz *online* se existe uma questão que ele julgue interessante para submeter a seus alunos. É importante lembrar que nos Juízes *Online* existem diversas questões dos mais diferentes níveis de dificuldade, questões que se encaixam bem no aprendizado de turmas onde esteja sendo cursada uma disciplina de programação. Mas ainda assim, o professor pode optar por não utilizar as questões disponibilizadas pelos Juízes *Online* e aí é onde se encontra uma das propostas de trabalhos futuros: uma maneira de o professor elaborar suas próprias questões e submetê-las aos alunos, utilizando um laboratório virtual de programação.

Ainda como trabalhos pretende-se integrar ao MOJO um editor de código-fonte para que o aluno possa desenvolver seu código no próprio módulo, além de uma ferramenta de avaliação automática, que auxilie o aluno na edição e avaliação de seus códigos, onde o próprio aluno poderá obter um *feedback* antes mesmo de submeter seu código para avaliação pelo juiz.

Agradecimentos

Os autores agradecem a CAPES e a FAPERN pela concessão das bolsas de pesquisa e pelo apoio financeiro para realização da mesma, e em especial ao Programa de Pós-Graduação em Ciência da Computação – PPgCC da Universidade do Estado do Rio Grande do Norte – UERN e Universidade Federal Rural do Semi-Árido – UFERSA, por toda infraestrutura oferecida.

Referências

Alves, L; Brito, M. (2005) "O Ambiente Moodle como Apoio ao Ensino Presencial". Disponível em: <http://www.abed.org.br/congresso2005/por/pdf/085tcc3.pdf>. Acesso em 12 de mar de 2013.

Beck, K; Gamma, E. (2010) "*JUnit Cookbook"*. Disponível em: <http://junit.sourceforge.net/doc/cookbook/cookbook.htm>. Acesso em 13 de mar de 2013.

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.

Carter, J; English, J; Ala-Mutka, K; Dick, M; Fone, W; Fuller, U; Sheard, J. (2003) "*ITICSE working group report: How shall we assess this?*". SIGCSE Bulletin, v. 35, n. 4, december 2003, pp 107–123.

CUnit Project. (2012) "*CUnit - A Unit testing Framework for C*". Disponível em: <http://cunit.sourceforge.net/>. Acesso em 13 de mar de 2013.

Ferradin, M; Stephani, S. L. (2005) "Ferramenta para o ensino de programação via Internet". In: I Congresso Sul Catarinense de Computação: UNESC – Criciúma, 2005.

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.

Ihantola, P; Ahoniemi, T; Karavirta, V; Seppälä, O. (2010) "*Review of recent systems for automatic assessment of programming assignments*". Proceedings of the 10th Koli Calling International Conference on Computing Education Research, Koli Calling '10, Koli, Finland. 2010.

Kurnia, A; Lim, A.; Cheang, B. (2001) "*Online Judge*". Computer & Education, v. 36, n. 4, maio 2001, pp 299-315.

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. 7, p. 1-8, 2008.

MOODLE. (2013) "*Modular Object-Oriented Dynamic Learning Environment*". Disponível em: <http://moodle.org/>. Acesso em 16 de mar de 2013.

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. (2008) "*JavaTool*: Uma Ferramenta Para Ensino de Programação". In: Anais do XX Simpósio Brasileiro de Informática na Educação. Florianópolis-SC, 2009.

Prior, J. C. (2003) "*Online assessment of SQL query formulation skills*". In Proceedings of the Fifth Australasian Conference on Computing Education. Adelaide, Australia. 2003.

Pulino, F. A. R. (2004) “Introdução ao Moodle - Ambiente de Aprendizagem (Módulo 1)”. Disponível em: <http://ead.faculdademarista.com.br/file.php/1/modulo01-moodle_1.pdf>. Acesso em 15 de mar de 2013.

Silva Filho, R. L. L; Motejunas, P. R; Hipólito, O; Lobo, M. B. C. M. (2007) “A evasão no ensino superior brasileiro”. Disponível em <http://www.scielo.br/pdf/cp/v37n132/a0737132.pdf>. Acesso em 13 de dez de 2012.

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. (2011) “*ProgTest: An environment for the submission and evaluation of programming assignments based on testing activities*”. Proceedings of the 24th IEEE-CS Conference on Software Engineering Education and Training, CSEET ’11, Honolulu, HI, USA. 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) “SPOJ Brasil”. Disponível em <http://br.spoj.pl/>. Acesso em 16 de mar de 2013.

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