

**UNIVERSIDADE FEDERAL DE SERGIPE
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA
DEPARTAMENTO DE COMPUTAÇÃO**

JOANNA CECÍLIA DA SILVA SANTOS

**JONLINE: JUIZ ONLINE PARA ENSINO DE
PROGRAMAÇÃO**

Trabalho de Conclusão de Curso
submetido ao Departamento de
Computação da Universidade
Federal de Sergipe como requisito
parcial para a obtenção do título de
Bacharel em Engenharia de
Computação.

Orientador: Admilson de Ribamar Lima Ribeiro

**SÃO CRISTÓVÃO
2013**

JOANNA CECILIA DA SILVA SANTOS

**JONLINE: JUIZ ONLINE PARA ENSINO DE
PROGRAMAÇÃO**

Trabalho de Conclusão de Curso submetido ao corpo docente do Departamento de Computação da Universidade Federal de Sergipe (DCOMP/UFS) como parte dos requisitos para obtenção do grau de Bacharel em Engenharia de Computação.

São Cristovão, 23 de setembro de 2013.

BANCA EXAMINADORA:

Profº Admilson de Ribamar Lima Ribeiro, Doutor
Orientador
DCOMP/UFS

Profº Sérgio Queiroz de Medeiros, Doutor
Universidade Federal de Sergipe

Profº Tarcísio da Rocha, Doutor
Universidade Federal de Sergipe

Dedico este trabalho à minha querida mãe.

AGRADECIMENTOS

Agradeço à minha mãe, Maria Clarice, pelo incentivo, educação, apoio, cuidados e, sobretudo, em me fazer compreender que a maior importância da vida não são os bens materiais, mas sim, o conhecimento, que nos permite uma participação consciente na sociedade. Sem ela não teria sido capaz de chegar até onde cheguei. Também agradeço aos demais familiares e amigos pelo apoio e companheirismo. Agradeço ainda aos professores que fizeram parte de minha formação acadêmica, de modo particular ao professor Admilson que, ao longo dos últimos anos, esteve me orientando na monitoria e nas iniciações tecnológica e científica, contribuindo significativamente para o meu amadurecimento enquanto estudante.

“Que os vossos esforços desafiem as impossibilidades, lembrai-vos de que as grandes coisas do homem foram conquistadas do que parecia impossível.”
(Charles Chaplin)

“Devemos mudar nossa atitude tradicional em relação à construção de programas. Em vez de imaginar que nossa principal tarefa é instruir o computador sobre o que ele deve fazer, vamos imaginar que nossa principal tarefa é explicar a seres humanos o que queremos que o computador faça.”
(Donald E. Knuth)

SANTOS, Joanna C. S. **JOnline: juiz online para ensino de programação**. 2013. Trabalho de Conclusão de Curso – Curso de Engenharia de Computação, Departamento de Computação, Universidade Federal de Sergipe, São Cristóvão, 2013.

RESUMO

Juízes online são muito utilizados em competições de programação para avaliar automaticamente se um programa resolve um dado problema corretamente. Eles despertam nos discentes o interesse em resolver problemas desafiadores através da construção de programas. Dessa maneira, embora os juízes online sejam voltados ao uso em competições de programação, é possível também utilizá-los como apoio em disciplinas básicas de computação que envolvem a construção de programas. No entanto, para que eles possam ser usados para esse propósito, é preciso agregar novas características a esses sistemas. Sendo assim, neste trabalho descreve-se um juiz online denominado JOnline que, além da avaliação de código-fonte, possui funcionalidades que visam auxiliar o discente no desenvolvimento de seus programas e, conseqüentemente, em seu aprendizado de programação de computadores. Essas funcionalidades são: apresentação de casos de testes que geraram resultados errados, organização de problemas por assunto e grau de dificuldade, votação do nível de dificuldade dos problemas pelos usuários, além da programação colaborativa permitindo a edição simultânea de um mesmo código-fonte por um ou mais usuários. Para agregar essas funcionalidades, o JOnline é desenvolvido utilizando serviços Web RESTful.

Palavras-chave: Juiz online, serviços Web, aprendizado de programação.

ABSTRACT

Online judges are systems largely used in programming competitions to evaluate if a program is able to solve a specific problem correctly. They rouse in students the interest in solving challenging problems through the development of computer programs. Therefore, although online judges are designed to be used in programming competitions, it is also possible to use them to support basic computing disciplines that involve the development of programs. However, it is required to add new features to these systems so they can be used for that purpose. Thus, this work describes an online judge called JOnline, which, besides source code evaluation, has features designed to assist the students in their developing computer programs and consequently in their computer programming learning process. These features are: presentation of the test cases that generated incorrect results, organization of the problems by topic and difficulty, user difficulty rating, as well as collaborative programming, which allows simultaneous editing of the same source code by one or more users. To assemble these functionalities, JOnline is developed using RESTful Web services.

Keywords: *Online judge, Web services, computer programming learning.*

LISTA DE FIGURAS

Figura 1 - Exemplo de mensagens SOAP	22
Figura 2 - Exemplo de WSDL.....	23
Figura 3 - Arquitetura do JOnline	34
Figura 4 - Modelo entidade relacionamento do banco de dados do JOnline.....	35
Figura 5 - Diagrama de casos de uso de um usuário com o papel “aluno”	36
Figura 6 - Diagrama de casos de uso de um professor e um administrador.....	37
Figura 7 - Página inicial do JOnline	46
Figura 8 - Página de cadastro de usuários no JOnline.....	47
Figura 9 - Página de gerenciamento de conta do usuário	47
Figura 10 - Formulário para recuperação de senha	48
Figura 11 - E-mail enviado ao usuário para informar sua nova senha	48
Figura 12 - Interface para gerenciamento de contas de usuários do JOnline	49
Figura 13 - Busca de problemas por nome	49
Figura 14 - Descrição do problema “População de coelhos”	50
Figura 15 - Página de submissão de código-fonte.....	50
Figura 16 - Aviso mostrado ao usuário indicando que há um erro de compilação no seu código-fonte.....	51
Figura 17 - Resultado correto para o problema “População de coelhos”	52
Figura 18 - Resultado incorreto com o caso de teste que causou a falha	52
Figura 19 - Saída mal formatada para um programa escrito em C.....	52
Figura 20 - Listagem de problemas disponíveis no JOnline	53
Figura 21 - Caixa de diálogo para buscar problemas	54
Figura 22 - Visualização do problema "O troco"	54
Figura 23 - Votação do nível de dificuldade de um problema	55
Figura 24 - Interface Web de gerenciamento de problemas.....	55
Figura 25 - Formulário Web para cadastro de problemas	56
Figura 26 - Cadastro de casos de teste para um problema	57
Figura 27 - Formulário para gerar arquivos com os resultados esperados	57
Figura 28 - (a) Listagem de disciplinas. (b) Detalhes da disciplina de Algoritmos	58
Figura 29 - Mensagem de aviso indicando que o aluno não está inscrito na turma.....	58

Figura 30 - Listagem de arquivos disponibilizados pelo professor.....	59
Figura 31 - Listagem de fóruns da turma	59
Figura 32 - Mensagem de erro avisando que o título do fórum deve ser preenchido	60
Figura 33 - Estudante “Olga” visualizando um fórum criado pela estudante “Joanna”	60
Figura 34 - Listagem de avisos de uma turma. Destaque para o quadro de avisos mostrado ao usuário	61
Figura 35 - Lista de tarefas para a turma de Algoritmos 2012/2 – A0.....	61
Figura 36 - Gerenciamento de disciplinas	62
Figura 37 - Listagem das turmas de um professor. Destaque para o atalho colocado no bloco "Gerenciar" do sistema	62
Figura 38 - Listagem de tarefas com as opções de edição, exclusão e cadastro	63
Figura 39 - Listagem de recursos com a opção de exclusão e cadastro	63
Figura 40 - Listagem de fóruns	63
Figura 41 - Listagem de avisos com a opção de edição, exclusão e cadastro.	64
Figura 42 - Visualização dos projetos do usuário Blaise Pascal	64
Figura 43 - Visualização de um projeto do usuário Blaise Pascal	65
Figura 44 - Caixa de diálogo para compartilhamento de um projeto	65
Figura 45 - Edição de um mesmo código-fonte por dois usuários	66
Figura 46 - Caixa de diálogo usada para inserir dados de entrada do programa.....	66
Figura 47 - Exemplo de resultado da execução de um projeto.....	67
Figura 48 - Erros de compilação mostrados na execução de um projeto	67
Figura 49 - Possível implementação de <i>streaming</i> de vídeo adaptativo	70

LISTA DE QUADROS

Quadro 1 - Exemplo de requisição HTTP para um serviço Web RESTful.....	24
Quadro 2 - Diferenças entre serviços Web RESTful e baseados no SOAP.....	25
Quadro 3 - Síntese das vantagens e desvantagens dos serviços Web baseados no SOAP e baseados no REST	26
Quadro 4 - Estrutura das <i>tags template:get</i> , <i>template:put</i> e <i>template:insert</i> criadas	40
Quadro 5 - Exemplo de página modelo usando as <i>tags</i> criadas	40
Quadro 6 - Exemplo de página que utiliza um modelo de página criado com as <i>tags</i> implementadas	41
Quadro 7 - Argumentos disponíveis para o <i>safeexec</i>	43
Quadro 8 - <i>Script</i> colocado no carregamento de uma página Web para configurar o MobWrite e o CodeMirror.....	45
Quadro 9 - <i>Script</i> para capturar áudio e vídeo do usuário.....	71

LISTA DE ABREVIATURAS E SIGLAS

ACM	<i>Association for Computing Machinery</i>
AJAX	<i>Asynchronous Javascript and XML</i>
API	<i>Application Programming Interface</i>
AVA	<i>Ambientes Virtuais de Aprendizagem</i>
BOCA	<i>BOCA Online Contest Administrator</i>
CAPTCHA	<i>Completely Automated Public Turing test to tell Computers and Humans Apart</i>
COOJ	<i>Course-Oriented Online Judge</i>
CRUD	<i>Create, Read, Update and Delete</i>
CSS	<i>Cascading Style Sheets</i>
DCOMP	<i>Departamento de Computação</i>
EaD	<i>Educação à Distância</i>
HTML	<i>HyperText Markup Language</i>
HTTP	<i>HyperText Transfer Protocol</i>
ICPC	<i>International Collegiate Programming Contest</i>
IOI	<i>International Olympiad in Informatics</i>
JSON	<i>JavaScript Object Notation</i>
JSP	<i>Java Server Pages</i>
JSTL	<i>JSP Standard Tag Library</i>
LMS	<i>Learning Management System</i>
PDF	<i>Portable Document File</i>
QoS	<i>Quality of Service</i>
REST	<i>Representational State Transfer</i>
SBC	<i>Sociedade Brasileira de Computação</i>
SIGAA	<i>Sistema Integrado de Gestão de Atividades Acadêmicas</i>
SOA	<i>Service Oriented Architecture</i>
SOAP	<i>Simple Object Access Protocol</i>
SPOJ	<i>Sphere Online Judge</i>
TI	<i>Tecnologia da Informação</i>
UDDI	<i>Universal Description, Discovery and Integration</i>
UFS	<i>Universidade Federal de Sergipe</i>
UML	<i>Unified Modeling Language</i>
URI	<i>Uniform Resource Identifier</i>
URL	<i>Uniform Resource Locator</i>
UUID	<i>Universally Unique Identifier</i>
UVA	<i>Universidad de Valladolid</i>
W3C	<i>World Wide Web Consortium</i>
WADL	<i>Web Application Description Language</i>
WSDL	<i>Web Services Description Language</i>
XML	<i>Extensible Markup Language</i>

LISTA DE NOTAÇÕES E SÍMBOLOS

™	Marca comercial
®	Marca registrada

SUMÁRIO

1	INTRODUÇÃO	14
1.1	Objetivos.....	15
1.2	Metodologia.....	16
1.3	Organização do trabalho	17
2	FUNDAMENTAÇÃO TEÓRICA.....	18
2.1	Juízes <i>Online</i> e competições de programação	18
2.2	Ambientes virtuais de aprendizagem e Sistemas de Gestão de Aprendizagem	19
2.3	Serviços Web	20
3	TRABALHOS RELACIONADOS	27
3.1	Juízes Online	27
3.2	Ambientes <i>online</i> de aprendizagem de programação	31
3.3	Comparativo dos trabalhos relacionados.....	31
4	PROJETO DO JONLINE	33
5	ASPECTOS DE IMPLEMENTAÇÃO.....	39
5.1	Tecnologias do lado do servidor	39
5.2	Tecnologias do lado do cliente	44
6	RESULTADOS	46
6.1	Módulo usuários	46
6.2	Módulo submissão	48
6.3	Módulo problemas	53
6.4	Módulo disciplinas.....	58
6.5	Módulo de programação colaborativa.....	64
7	CONCLUSÃO.....	68
	APÊNDICE	70
	REFERÊNCIAS	73

1 INTRODUÇÃO

O aprendizado de programação de computadores é de suma importância para a formação de profissionais na área de computação. Além de capacitar o indivíduo para utilizar a lógica de programação na resolução de problemas, a programação é a base para muitos campos em que a computação é aplicada, fator relevante em disciplinas avançadas (MOREIRA; FAVERO, 2009).

De acordo com Pimentel et al. (2003) uma quantidade relevante de discentes das disciplinas iniciais focadas ao estudo de programação possuem dificuldade em assimilar e utilizar os conceitos abstratos de programação e, como consequência, é observado um alto número de reprovações nessas disciplinas introdutórias. Esses fatos são ocasionados devido ao processo complexo e exigente que é aprender e desenvolver lógica de programação para a maioria dos alunos (FERRANDIN; STEPHANI, 2005). Dessa forma, estabelecer maneiras de facilitar o ensino de programação torna possível a melhoria na qualidade de ensino de cursos em computação. Consequentemente, isso permite a redução da quantidade de reprovações em disciplinas que possuem como pré-requisito o conhecimento de programação.

Uma etapa importante para que um discente possa desenvolver a habilidade de programar bem é praticar, ou seja, escrever programas para resolver algum problema (CHEANG et al., 2003). Sendo assim, os docentes atribuem tarefas de implementação para os alunos resolverem e que, posteriormente, serão corrigidas pelo docente. Entretanto, tendo em vista que a correção manual é um processo que consome muito tempo do professor, por vezes a quantidade de tarefas atribuídas é insuficiente para a prática dos conceitos aprendidos em sala. Uma maneira de resolver esse problema é construir um ambiente *online* que automaticamente avalie o código-fonte do aluno e apresente o resultado ao mesmo.

Um ambiente *online* muito comum utilizado em competições de programação que fazem a avaliação automática de códigos-fonte é o juiz *online*, a exemplo do SPOJ (*Sphere Online Judge*) (SPHERE ONLINE LAB, 2013) e UVA (Universidad de Valladolid) *Online Judge* (UNIVERSIDAD DE VALLADOLID, 2013). Um juiz *online* consiste de um servidor que contém descrições de problemas e dados que são utilizados no processo de julgamento do código-fonte enviados ao ambiente (REVILLA; MANZOOR; LIU, 2008). Nesses ambientes, quando um usuário submete para avaliação seu código-fonte contendo a solução para um

problema, o resultado é mostrado em poucos segundos. Esse resultado indica se a solução do usuário está correta ou se houve algum erro em tempo de compilação ou execução. Também é mostrado ao usuário quanto tempo foi gasto na solução do problema e a quantidade de memória consumida após a execução do programa.

De acordo com Revilla, Manzoor e Liu (2008), juízes *online* são bem sucedidos em despertar nos discentes a curiosidade e interesse em resolver problemas desafiadores, podendo ainda atrair novos estudantes (por exemplo, alunos do ensino fundamental e médio) para o mundo da programação e informática em geral. Dessa maneira, embora os juízes *online* tenham sido projetados para uso em competições de programação e para treinar estudantes para essas competições, eles podem ser utilizados em disciplinas com ênfase em programação de computadores.

Para que o uso de juízes *online* no apoio a disciplinas seja possível, é necessário adicionar novas funcionalidades que facilitem o aprendizado de programação visto que poucas funcionalidades didáticas são encontradas nos mesmos. Por exemplo, após a submissão da resolução de um problema, o usuário obtém um resultado que apenas indica se a resposta está certa, errada ou se houve algum erro em tempo de execução ou compilação. Esse resultado, porém, não fornece mais informações que possam indicar onde o usuário errou, sendo uma responsabilidade do mesmo encontrar o seu erro e corrigi-lo.

1.1 Objetivos

Partindo do pressuposto que o uso de juízes *online* pode auxiliar no processo de aprendizado de programação, esse trabalho tem como principal objetivo o desenvolvimento de um juiz *online*, chamado JOnline, com funcionalidades que possam ajudar discentes em seu aprendizado.

Além das características básicas comuns aos juízes *online* existentes, o JOnline também agrega as seguintes funcionalidades:

- Programação colaborativa que permite que um ou mais usuários editem *online* um mesmo código-fonte simultaneamente. Além disso, os alunos podem se comunicar entre si usando *chat*.
- Mostrar os casos de testes que geraram resultados incorretos.

- Organização dos problemas por assunto abordado e grau de dificuldade. O grau de dificuldade do problema é resultado da média da nota atribuída pelos próprios usuários mediante uma votação.
- Professores podem criar disciplinas e turmas no JOnline e disponibilizar material, tarefas e avisos das turmas. Os estudantes podem se inscrever nas turmas para ter acesso ao seu respectivo conteúdo. Os participantes da turma podem ainda criar fóruns para discussões de assuntos relacionados à turma.

Os objetivos específicos desse trabalho são:

- Desenvolver um ambiente online que futuramente possa servir de apoio para disciplinas de programação ofertadas pelo Departamento de Computação (DCOMP) da Universidade Federal de Sergipe (UFS);
- Disponibilizar as funcionalidades do JOnline usando serviços Web RESTful;
- Disponibilizar o JOnline para que ele possa ser implantado em outras instituições acadêmicas;
- Publicar os resultados desse trabalho em anais de eventos;

1.2 Metodologia

Inicialmente foi feita uma revisão da literatura visando identificar os juízes *online* existentes e trabalhos que utilizam ambientes de aprendizado *online* em cursos de computação. Além disso, também foram estudados trabalhos sobre serviços Web tendo em vista que o JOnline foi projetado para possuir uma arquitetura orientada a serviços.

Após a revisão da literatura, foram escolhidas ferramentas e diversas API (*Application Programming Interface*) para auxiliar o desenvolvimento do JOnline. Posterior a essa identificação, foram feitos diagramas UML (*Unified Modeling Language*) (BOOCH; RUMBAUGH; JACOBSON, 2000) para identificar os aspectos do sistema e projetar a arquitetura do ambiente.

Feito o projeto do JOnline, deu-se início ao desenvolvimento do sistema seguido da sua implantação em um servidor Web localizado no Laboratório de Projetos do Departamento de Computação da UFS.

1.3 Organização do trabalho

Este trabalho, além deste primeiro capítulo, possui mais seis capítulos. O capítulo 2 trata da fundamentação teórica necessária para o entendimento do trabalho. No capítulo 3 são apresentados trabalhos relacionados sobre juízes *online*, ambientes de aprendizado *online* e serviços Web. No capítulo 4 o projeto do JOnline é apresentado. No capítulo 5 são discutidas questões relacionadas ao desenvolvimento do sistema. No capítulo 6 é mostrado o funcionamento do JOnline, mostrando o estado atual do mesmo. No capítulo 7, último deste trabalho, são feitas as considerações finais apresentando uma síntese dos resultados obtidos e trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Nas próximas subseções serão apresentados conceitos relacionados aos juízes *online*, ambientes de aprendizagem *online* e serviços Web.

2.1 Juízes *Online* e competições de programação

Nas competições de programação os alunos desenvolvem programas para resolver um conjunto de problemas. Dependendo da competição, os discentes podem estar competindo em equipe ou sozinhos. Essas competições são uma maneira de despertar nos discentes o interesse em aprender programação, possibilitando tanto que os alunos aprendam a trabalhar em equipe e sob pressão quanto o aprendizado de conceitos importantes de computação, tais como estruturas de dados, algoritmos e desenvolvimento de *software* (CAMPOS; FERREIRA, 2004).

Exemplos de competições de programação são a Olimpíada Internacional de Informática (IOI - *International Olympiad in Informatics*) e a Competição Internacional de Programação Universitária (ICPC – *International Collegiate Programming Contest*) organizada pela ACM (*Association for Computing Machinery*). Em ambas as competições, os estudantes dispõem de um computador para resolver um conjunto de problemas de natureza algorítmica em um determinado tempo.

A principal diferença entre essas duas competições é a forma de avaliação dos problemas. Enquanto que na IOI os alunos não possuem nenhum tipo de informação em tempo real para verificar se sua solução está correta, na ICPC as soluções são avaliadas em tempo real, ou seja, no instante em que o aluno submete seu código-fonte o sistema apresenta o resultado da submissão depois do aval de juízes humanos. Outra diferença é o fato de que os alunos que participam da IOI são de ensino médio e competem individualmente enquanto que na ICPC os alunos são de instituições de ensino superior e competem em equipes.

No Brasil, a Maratona de Programação, organizada pela Sociedade Brasileira de Computação (SBC) desde 1996, é utilizada como meio de classificação de equipes para a final mundial da ICPC (SBC, 2013). Nessa competição, as equipes compostas por três alunos de ensino superior, tentam resolver durante cinco horas os oito ou mais problemas da competição. As equipes dispõem de material impresso e um computador para desenvolver

suas soluções e conferir se as mesmas estão corretas. A equipe vencedora é aquela que resolve o maior número de problemas.

Nessas competições os programas são de natureza algorítmica, ou seja, apenas necessitam obter em sua entrada padrão dados devidamente formatados e, a partir desses dados, efetuar o processamento apresentando os resultados de maneira formatada em sua saída padrão. Sendo assim, é possível que a avaliação de programas seja feita de maneira automática utilizando um programa que gere os dados de entrada e outro que obtenha e verifique os resultados adquiridos (KURNIA; LIM; CHEANG, 2001).

Esse processo automático de avaliação é feito pelos chamados juízes *online*. Eles recebem código-fonte enviado pelo usuário e, posteriormente, compila e executa esse código. Durante a execução do programa, os juízes *online* utilizam dados formatados como a entrada do programa e, após o processamento, é feita a comparação dos resultados obtidos com os esperados. Dessa forma, um juiz *online* é um servidor que contém descrições de problemas e dados utilizados no processo de avaliação das submissões (REVILLA; MANZOOR; LIU, 2008).

A maioria dos juízes *online* baseia-se nas regras de julgamento da ICPC para avaliar os códigos-fonte submetidos ao sistema. Desse modo, quando um usuário envia um código-fonte contendo sua solução para um determinado problema, o resultado da submissão é mostrado em poucos segundos. Esse resultado indica se a solução do usuário está correta, incorreta ou se ocorreram erros em tempo de compilação ou execução.

2.2 Ambientes virtuais de aprendizagem e Sistemas de Gestão de Aprendizagem

A popularização do uso de computadores conectados à Internet favoreceu a adoção de Ambientes Virtuais de Aprendizagem (AVA) para auxiliar professores no decorrer de suas disciplinas. Esses ambientes virtuais oferecem suporte ao processo de ensino e de aprendizagem através da Internet, permitindo que alunos possam acessar materiais relacionados ao conteúdo da disciplina, verificar quais são as tarefas, participar de fóruns, entre outras atividades. De acordo com Barbosa (2006), tendo em vista a eficácia de tais sistemas *online*, universidades e empresas desenvolvem seus próprios ambientes de aprendizagem, a exemplo do Coursera (COURSERA, 2013).

Embora os ambientes virtuais de aprendizagem favoreçam o processo de ensino-aprendizagem, faltam aos mesmos ferramentas intuitivas para gestão dessa aprendizagem e,

assim, diversas vezes os docentes utilizam outros recursos computacionais (a exemplo de planilhas eletrônicas, anotações, entre outros) para realizar essa gestão (GOMES et al., 2009). Nesse contexto, um Sistema de Gestão de Aprendizagem (LMS - *Learning Management System*) resolve esse problema permitindo o gerenciamento de conteúdo de aprendizado de maneira interativa e gradativa. Um LMS possibilita o registro de atividades realizadas pelos alunos permitindo que os docentes possam acompanhar o aprendizado dos mesmos.

As principais características dos LMS são as seguintes (GOMES et al., 2009):

- o registro e monitoramento das atividades e acesso dos alunos;
- recursos interativos;
- gestão de conteúdo que permite docentes a criarem seus cursos de modo que os discentes possam facilmente encontrar o que desejam;
- sistema colaborativo de aprendizagem que permite o trabalho em grupo e o compartilhamento de conhecimentos;

O SIGAA (Sistema Integrado de Gestão de Atividades Acadêmicas) usado na UFS é um exemplo de sistema de gestão de aprendizagem. Esse sistema, desenvolvido pela Universidade Federal do Rio Grande do Norte, possui diversos módulos relacionados com o tipo de atividade oferecida pela instituição, a exemplo dos módulos de graduação, pós-graduação (*stricto e lato sensu*), ensino técnico, ensino médio e infantil, entre outros (SINFO, 2013). De acordo com a análise feita por Queiroz et al. (2012) esse sistema auxilia os discentes em seu processo de aprendizagem pois propicia uma melhor interação entre aluno e docente.

O *site* da empresa Capterra, especializada na comparação de programas com o propósito de auxiliar corporações a escolher o *software* mais adequado para seus negócios, publicou em 2012 uma lista dos LMS mais usados. De acordo com Capterra (2012) os cinco LMS mais populares são o Moodle (60 mil usuários), o SumTotal Systems (32 mil usuários), o Blackboard (20 mil), o Edmodo (10 mil usuários) e o Interactyx (10 mil usuários).

2.3 Serviços Web

Uma arquitetura orientada a serviços (SOA - *Service Oriented Architecture*) é um estilo de arquitetura de *software* que visa desenvolver módulos funcionais, chamados de serviços, que possuem baixo acoplamento e permitem a reutilização de código (SAMPAIO, 2006). Dessa forma, uma arquitetura orientada a serviços suporta a construção de sistemas

distribuídos em que partes do sistema estão localizadas em outros computadores distribuídos geograficamente. Essas partes se comunicam através de serviços que executam sob diferentes plataformas e que foram desenvolvidos usando diferentes linguagens de programação (SOMMERVILLE, 2007).

Nos últimos anos, tem sido notado o crescente uso de tecnologias Web para soluções de TI (Tecnologia da Informação). Dessa forma, o exemplo mais comum de serviços são os serviços Web. Esses tipos de serviço são uma tecnologia de integração completamente baseada em padrões Web e que, geralmente, utilizam o protocolo HTTP (*HyperText Transfer Protocol*) como transporte (STAL, 2002). Por serem independentes de linguagem e plataformas subjacentes, diversas empresas usam serviços Web para disponibilizar seus aplicativos.

A máquina que contém a implementação de um serviço Web é chamada de *host* de serviço Web e é responsável por publicar o serviço e processar as solicitações dos clientes fornecendo uma resposta (DEITEL; DEITEL, 2010). Esses serviços podem ser implementados baseados no protocolo SOAP (*Simple Object Access Protocol*) ou no estilo de arquitetura REST (*Representational State Transfer*).

O SOAP é um protocolo que define a estrutura de mensagens XML (*eXtensible Markup Language*) de requisição e de resposta dos serviços Web (W3C, 2012b). Esse protocolo inicia com um marcador (*tag*) chamado *Envelope* que engloba duas partes: cabeçalho (*Header*) e corpo (*Body*). No cabeçalho da mensagem são colocadas informações de controle e no corpo o conteúdo da mensagem tanto de quem envia quanto de quem recebe (SAMPAIO, 2006). Quando a mensagem é gerada pelo cliente, o *Body* especifica qual a operação do serviço a ser realizada e quais são os valores dos parâmetros. Quando se trata de uma mensagem de resposta do *host* de serviço, então o *Body* contém o resultado da operação.

Na figura 1 é possível visualizar exemplos de mensagens SOAP para invocar um serviço Web. Nessa figura, é mostrada uma mensagem SOAP para invocar a operação *somar* especificando que os parâmetros dessa operação, *a* e *b*, serão os valores 1 e 3.5, respectivamente. Nota-se ainda que a mensagem de resposta contém somente o valor da soma (4.5) dentro da marcação *return*.

Os provedores de serviços Web descrevem os serviços disponíveis usando WSDL (*Web Services Description Language*). Essa linguagem especifica o que o serviço pode realizar, como ele se comunica e qual o endereço para acessá-lo usando o formato XML (SOMMERVILLE, 2007). Na figura 2 é apresentado um exemplo de um WSDL que descreve um serviço chamado *CalculadoraService* que disponibiliza uma operação chamada *somar*, cujas mensagens SOAP para invocar essa operação podem ser vistas na figura 1.

```
<!-- Mensagem de requisição -->
<soapenv:Envelope
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:ser="http://services/">
  <soapenv:Header/>
  <soapenv:Body>
    <ser:somar>
      <a>1</a>
      <b>3.5</b>
    </ser:somar>
  </soapenv:Body>
</soapenv:Envelope>

<!-- Mensagem de resposta -->
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:somarResponse xmlns:ns2="http://services/">
      <return>4.5</return>
    </ns2:somarResponse>
  </S:Body>
</S:Envelope>
```

Figura 1 Exemplo de mensagens SOAP

Conforme observado na figura 2, WSDL possui os seguintes elementos para descrever serviços (W3C, 2012c):

- *Types* (tipos): contém as definições dos tipos de dados usados.
- *Message* (mensagem): uma definição abstrata que especifica os dados sendo transmitidos e quais são os tipos desses dados.
- *Operation* (operação): uma descrição abstrata de uma ação suportada pelo serviço.
- *Port Type* (tipo da porta): um conjunto abstrato de operações suportado por uma ou mais portas (*endpoints*).
- *Binding* (ligação): um protocolo concreto e uma especificação de formatos de dados para um tipo de porta específico.
- *Port* (porta): um único *endpoint* definido como uma combinação de uma ligação (*binding*) e um endereço de rede.
- *Service* (serviço): uma coleção de *endpoints* relacionados.

```

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<definitions targetNamespace="http://services/" name="CalculadoraService" xmlns="http://schemas.xmlsoap.org/wsdl/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/">
  <types>
    <xsd:schema>
      <xsd:import namespace="http://services/" schemaLocation="CalculadoraService_schema.xsd"/>
    </xsd:schema>
  </types>
  <message name="somar">
    <part name="parameters" element="tns:somar"/>
  </message>
  <message name="somarResponse">
    <part name="parameters" element="tns:somarResponse"/>
  </message>
  <portType name="Calculadora">
    <operation name="somar">
      <input wsam:Action="http://services/Calculadora/somarRequest" message="tns:somar"/>
      <output wsam:Action="http://services/Calculadora/somarResponse" message="tns:somarResponse"/>
    </operation>
  </portType>
  <binding name="CalculadoraPortBinding" type="tns:Calculadora">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="document"/>
    <operation name="somar">
      <soap:operation soapAction=""/>
      <input>
        <soap:body use="literal"/>
      </input>
      <output>
        <soap:body use="literal"/>
      </output>
    </operation>
  </binding>
  <service name="CalculadoraService">
    <port name="CalculadoraPort" binding="tns:CalculadoraPortBinding">
      <soap:address location="http://localhost:8080/Webservices/CalculadoraService?wsdl"/>
    </port>
  </service>
</definitions>

```

Figura 2 Exemplo de WSDL

Os serviços descritos usando WSDL são publicados em um Registro de Serviço usando o padrão UDDI (*Universal Description, Discovery and Integration*) (SOMMERVILLE, 2007). A principal entidade do UDDI é a entidade empresarial (*business entity*) que possui um nome, um identificador exclusivo, descrição, contatos (pessoas que podem ser usadas para contatar a entidade) e pode prover vários serviços (OASIS, 2013). Cada serviço possui um nome, uma descrição e várias implementações. Dessa forma, os clientes usam UDDI para descobrir serviços Web e verificar quais são as empresas que disponibilizam esses serviços.

Uma alternativa ao uso de serviços Web baseados no protocolo SOAP é o desenvolvimento de serviços Web baseados no REST, chamados de serviços Web RESTful. O REST é um estilo de arquitetura de *software* para sistemas de hipermídia distribuídos tais

como a World Wide Web (HAMAD; SAAD; ABED, 2009). Dessa forma, os princípios que guiam o desenvolvimento de serviços Web RESTful são (HANSEN, 2007):

- serviços sem estado: as requisições feitas pelo cliente contém todas as informações necessárias para o entendimento da requisição. Sendo assim, as requisições não levam em consideração nenhum tipo de contexto armazenado no servidor.
- interface uniforme: são permitidas somente as operações suportadas pelo protocolo HTTP.
- identificador único: as arquiteturas baseadas no REST são construídas a partir de recursos, ou seja, pedaços de informação que possuem um identificador único (URI – *Uniform Resource Identifier*)
- representação de recursos: os recursos são manipulados pelos serviços Web através de sua representação. Os recursos podem ser representados, por exemplo, usando XML.

Os serviços Web RESTful precisam somente do protocolo HTTP para troca das mensagens. Esse tipo de serviço expõe os recursos através da Web geralmente usando as quatro principais operações do HTTP: POST, GET, PUT e DELETE. Ao mapear essas quatro operações, é possível realizar as ações CRUD (*Create* – Criar, *Read* - Ler, *Update* – Atualizar, *Delete* - Apagar). Normalmente o mapeamento ocorre da seguinte forma: o método GET é utilizado para obter recursos (Ler), o POST é usado para criar um novo recurso (Criar), o PUT para atualizar algum recurso (Atualizar) e o DELETE para apagar um recurso (Apagar) (HAMAD; SAAD; ABED, 2009). No quadro 1 é possível ver um exemplo de requisição HTTP GET para um serviço que soma dois números, análogo ao mostrado na figura 2. Nessa requisição o cabeçalho *Accept* especifica que se espera que o serviço retorne os dados em formato XML.

```
GET /calculadora?a=1&b=3.5
Host: www.example.com
Accept: application/xml
```

Quadro 1 Exemplo de requisição HTTP para um serviço Web RESTful

Embora o REST não seja um padrão, serviços Web RESTful são implementados utilizando padrões Web e, assim, esses serviços podem ser utilizados diretamente de um navegador Web (no caso de requisições GET) ou usando um programa. Ao contrário dos serviços baseados no protocolo SOAP, os serviços Web RESTful não se limitam a retornar dados em XML, podendo também retorná-los em formato JSON (*JavaScript Object*

Notation), HTML (*HyperText Markup Language*), texto puro, arquivos de mídia, entre outros (DEITEL; DEITEL, 2010). As principais diferenças entre serviços Web baseados no SOAP e serviços Web RESTful podem ser vistas no quadro 2.

	REST	SOAP
Formato de mensagem	XML	XML dentro de um envelope SOAP
Definição da interface	nenhum ¹	WSDL
Transporte	HTTP	HTTP, FTP, MIME, JMS, SMTP, entre outros

Quadro 2 Diferenças entre serviços Web RESTful e baseados no SOAP.

Fonte: Hansen (2007)

Tendo em vista que os serviços Web podem ser construídos baseados no SOAP ou no estilo de arquitetura REST, diversos trabalhos discutem as vantagens de cada um, comparando os dois tipos de acordo com o desempenho, complexidade e aplicabilidade.

De acordo com Hamad, Saad e Abed (2009) o uso de serviços Web baseados no SOAP em dispositivos móveis resultam em maior tamanho de mensagem e tempo de resposta se comparado com o uso de serviços Web RESTful. Esse trabalho indica ainda que a principal causa do problema de desempenho de serviços Web baseados no SOAP nesses dispositivos é a necessidade de codificação e decodificação de mensagens SOAP que estão em formato XML.

Hamad, Saad e Abed (2009) discutem ainda que o uso de serviços Web baseados no SOAP apresenta problemas de *overhead*² em que se gastam mais *bytes* com marcadores XML do que com o conteúdo propriamente dito da mensagem. Além disso, à medida que o uso de serviços Web SOAP evoluiu, a arquitetura expandiu-se e lida também com funcionalidades mais complexas, como segurança e confiabilidade de mensagem. Dessa forma, levando em consideração o resultado das análises de desempenho feitas de dois serviços Web (um baseado no SOAP e outro RESTful), Hamad Saad e Abed (2009) recomendam o uso de serviços Web RESTful em dispositivos móveis.

Embora os serviços SOAP possuam os problemas destacados acima, esses tipos de serviços possuem a vantagem de serem padronizados pela W3C (*World Wide Web Consortium*) enquanto que os serviços Web RESTful não o são. Desse modo, os serviços baseados no SOAP podem ter clientes gerados automaticamente a partir da descrição do

¹ A Linguagem de Descrição de Aplicações Web (WADL - *Web Application Description Language*) foi submetida em 2009 para a W3C (*World Wide Web Consortium*) com intuito de padronizá-la. A WADL, semelhante à WSDL, tem a finalidade de fornecer uma descrição XML de aplicações Web baseadas em HTTP e, assim, podendo ser usada para descrever serviços Web RESTful (HADLEY, 2009).

² Termo que indica o uso desnecessário de processamento ou armazenamento, consumindo recursos do computador e gerando problemas de desempenho.

serviço feita usando WSDL. Além disso, os serviços Web baseados no SOAP possuem apoio de diversos padrões tais como o *WS-Security* (troca de mensagens seguras), *WS-Reliable Messaging* (assegura que as mensagens serão entregues somente uma vez), *WS-Addressing* (define informações de endereçamento) e *WS-Transactions* (coordenação de serviços distribuídos).

Os serviços Web RESTful possuem a vantagem de não se limitarem a retornar dados em formato XML, podendo também retorná-los em formato HTML, JSON, texto puro e arquivos de mídia (DEITEL; DEITEL, 2010). Entretanto, esses serviços têm a desvantagem de terem suas operações limitadas as operações suportadas pelo HTTP. Além disso, de acordo com Chen et al. (2010) a falta de padronização e de uma linguagem de descrição para serviços Web RESTful os tornam inadequados para uso em situações que demandam a invocação dinâmica de serviços Web, ou seja, em que as informações sobre o serviço são descobertas e processadas em tempo de execução.

No quadro 3 é possível visualizar um resumo das vantagens e desvantagens discutidas pelos trabalhos apresentados anteriormente.

	Serviços Web baseados no SOAP	Serviços Web RESTful
Vantagens	Padronizado pela W3C	Menor tempo de resposta em dispositivos móveis.
	Suporta questões de segurança de mensagens fim-a-fim, coordenação, confiabilidade e endereçamento usando padrões de apoio.	Simplicidade
		Leves (consomem menos <i>bytes</i>)
	Suporta a invocação dinâmica de <i>Web services</i>	Suporta o retorno de dados em outros formatos além do XML, tais como JSON, HTML, texto puro e arquivos de mídia.
	Suporta a criação de clientes gerados automaticamente a partir da descrição do serviço usando WSDL	Requisições podem ser armazenadas em <i>cache</i> .
Desvantagens	Complexidade	Não suporta a invocação dinâmica de serviços Web
	<i>Overhead</i>	Limitação de operações que podem ser realizadas (somente operações do protocolo HTTP)
	Necessidade de codificação e decodificação de mensagens XML.	Não há um padrão comum aceito para descrição formal desses serviços.

Quadro 3 Síntese das vantagens e desvantagens dos serviços Web baseados no SOAP e baseados no REST

3 TRABALHOS RELACIONADOS

Neste capítulo serão apresentados os principais juízes *online* existentes e trabalhos que usam os mesmos como apoio em disciplinas de cursos na área de Computação.

3.1 Juízes Online

Os juízes *online* são muito utilizados em maratonas de programação e podem ser encontrados na Internet. Entre eles pode-se citar o SPOJ (SPHERE ONLINE LAB, 2013), o juiz *online* URI (URI ONLINE JUDGE, 2013), o TopCoder (TOPCODER, 2013) e os juízes *online* da Universidade de Valladolid (UNIVERSIDAD DE VALLADOLID, 2013) e da Universidade de Pequim (FUCHEN; PENGCHENG; DI, 2013).

Nesses sistemas são disponibilizados vários problemas a serem resolvidos e que podem ser obtidos em algum formato de arquivo, a exemplo do formato PDF (*Portable Document Format*). Sendo assim, um usuário pode enviar sua solução para um determinado problema e escolher a linguagem utilizada na escrita do código-fonte do programa. Além disso, esses sistemas disponibilizam fóruns de discussão, *ranking* de usuários e estatísticas para cada problema (por exemplo, a quantidade de pessoas que resolveram o problema).

O *BOCA Online Contest Administrator* (BOCA) é um sistema que visa apoiar as competições de programação (CAMPOS; FERREIRA, 2004). Sua ideia principal é disponibilizar aos alunos e aos professores o sistema de apoio utilizado em maratonas de programação. Ele possui uma interface Web em que cada time participante da competição possui um *login* próprio e, utilizando esse *login*, pode submeter a resolução de um problema, modificar informações de cadastro, enviar dúvidas aos juízes, observar o placar da competição em tempo real, solicitar a impressão de códigos-fonte e pedir ajuda ao pessoal de *staff*. Além disso, é disponibilizada uma interface de administração da competição que permite o gerenciamento de todo o sistema.

Tendo em vista que o BOCA foi desenvolvido para funcionar na Web, ele pode ainda ser utilizado em competições distribuídas, ou seja, os times poderiam estar em qualquer lugar desde que estivessem conectados à Internet. Nesse caso, o processamento é feito todo

em um único local para garantir uma maneira igualitária de avaliar as equipes, pois esta avaliação é feita pelo mesmo grupo de juízes.

Embora os juízes *online* tenham sido projetados para apoiar competições de programação e para treinar estudantes para essas competições, eles podem ser usados também para o ensino de informática e para melhorar as habilidades de programação (codificação, depuração e teste) dos estudantes (REVILLA; MANZOOR; LIU, 2008). Dessa forma, existem trabalhos na literatura que utilizam juízes *online* na realização de atividades de disciplinas de programação e discutem as vantagens desse uso.

No trabalho de Kurnia, Lim e Cheang (2001) são apresentadas as desvantagens da avaliação manual de exercícios de alunos. A principal desvantagem da avaliação manual de exercícios apontada por esse trabalho é a possível inconsistência que pode ocorrer na avaliação dos exercícios devido a influências do estilo de programação e estado emocional dos avaliadores. Além disso, podem existir casos em que o avaliador atribui notas mais baixas para discentes cujos exercícios estão corretos e que, no entanto, resolveram de uma maneira diferente da solução projetada pelo avaliador. Outra desvantagem é o fato de que somente os alunos de uma determinada disciplina são avaliados, ou seja, os discentes que não estão cursando a disciplina e que desejam fazer os exercícios para estudos próprios não terão suas respostas corrigidas. Para superar esses problemas, Kurnia, Lim e Cheang (2001) utilizam um juiz *online* com o propósito de avaliar os exercícios dos alunos atribuindo-lhes notas automaticamente.

O modelo de avaliação automática proposto por Kurnia, Lim e Cheang (2001) funciona da seguinte forma: uma máquina dedicada possui um juiz *online* que, a cada submissão, registra informações sobre a submissão, avalia o programa e depois retorna os resultados ao usuário. Esse juiz foi usado como apoio para as disciplinas de Estruturas de Dados e Algoritmos, Programação Competitiva e para seleção para a maratona de programação em 1999. Esse juiz avalia códigos desenvolvidos usando as linguagens C++ e Java.

Em Yi et al. (2010) é descrito o projeto de um juiz *online* orientado a cursos (COOJ - *Course-Oriented Online Judge*) que combina as vantagens de uma plataforma de aprendizado colaborativo baseada na Web com as vantagens dos juízes *online* tradicionais. No COOJ é possível criar classes, semelhante às turmas no mundo real, sob responsabilidade de

um professor ou monitor. Os estudantes podem se inscrever em uma ou mais classes e formar grupos. Cada grupo possui um estudante como líder.

Sendo assim, no COOJ os usuários são separados em cinco papéis: administrador, professor, monitor, líder de grupo e estudantes. O administrador é responsável por todo o gerenciamento do sistema, sendo capaz de fazer atribuição de papéis a usuários. Um usuário com o papel professor pode adicionar testes para uma ou mais classes das quais é esteja responsável, verificar as submissões dos alunos, enviar avisos, se comunicar com os discentes, criar grupos nas classes e adicionar novos problemas ao juiz *online*. O monitor, papel atribuído pelo professor, pode verificar as soluções dos alunos e atribuir-lhes notas subjetivas que levam em consideração o estilo de programação do aluno (nome de variáveis, formatação do código-fonte, entre outros). O líder de grupo pode atribuir tarefas aos membros do grupo e monitorar o processo de realização dessas tarefas. O papel estudante corresponde aos usuários que usam o ambiente para aprendizado, submetendo código-fonte e participando das classes e grupos.

O COOJ também dispõe de um sistema de recomendação de problemas de acordo com as especialidades dos estudantes. Desse modo, professores e monitores podem verificar quais os tipos de problemas que os estudantes possuem mais interesse, podendo fazer ajustes na metodologia de ensino da disciplina.

A avaliação feita no COOJ é composta de duas partes: objetiva e subjetiva. A parte objetiva é oriunda do próprio juiz (avaliação automática). A parte subjetiva é dada como uma combinação do tempo de execução e memória consumida. Esse valor subjetivo pode ser alterado pelo professor ou monitor conforme achem necessário.

Li et al. (2012) apresenta um novo tipo de juiz *online* voltado para o curso de Estrutura de Dados. Nesse sistema, são adicionados três novos tipos de problemas que avaliam a habilidade do discente de implementar e aplicar as estruturas de dados. No primeiro tipo de problema, que verifica a habilidade do discente em implementar as estruturas de dados, é solicitado que os discentes apenas implementem as funções contidas na interface da estrutura de dados. Assim, o sistema se encarrega de invocar as funções implementadas e verificar se elas estão corretas usando casos de teste previamente definidos. No segundo tipo de problema, em que se verifica a habilidade do aluno em aplicar a estrutura de dados na

resolução do problema, o sistema disponibiliza a implementação da interface da estrutura de dados e o discente deve invocar as funções corretamente de modo que possa resolver o problema. O terceiro tipo de problema é igual aos problemas contidos nos juízes *online* tradicionais e, assim, são verificadas a habilidade de implementação e aplicação de estruturas de dados na resolução de problemas.

Além da adição de novos tipos de problemas, toda vez que um usuário é registrado no juiz *online* apresentado em Li et al. (2012), o sistema recomenda o primeiro tipo de problema para que os discentes iniciem o aprendizado de estruturas de dados a partir da implementação das interfaces dessas estruturas. Após o usuário resolver tais tipos de problemas, o sistema recomenda o segundo tipo de problema para que os discentes possam adquirir habilidade no uso das estruturas de dados. E, por fim, é recomendado o terceiro tipo de problema aos usuários. A principal vantagem dessa abordagem é que o juiz *online* conduz o discente a um aprendizado gradual de Estruturas de Dados através da resolução de problemas dos mais simples aos mais complexos.

Chen et al. (2012) apresenta uma plataforma de aprendizado de programação que usa um juiz *online* para correção em tempo real de exercícios atribuídos por professores. Nela os professores podem administrar os exercícios e verificar o progresso dos estudantes. Além disso, as soluções enviadas pelos discentes podem ter suas notas diminuídas ou aumentadas de acordo com as regras de avaliação da plataforma. Essa plataforma possui ainda a detecção de plágio e, assim, cada submissão tem seu código comparado com as submissões já enviadas ao sistema.

Um trabalho similar ao JOnline é o projeto EduJudge que incorpora processos de aprendizado nos campos de programação e matemática (REVILLA; MANZOOR; LIU, 2008). Esse projeto tem como principal objetivo dar um caráter mais pedagógico para o juiz *online* da Universidade de Valladolid de modo a torná-lo um ambiente educacional eficaz para o ensino superior e secundário. Para alcançar esse objetivo, o juiz *online* da Universidade de Valladolid foi melhorado e integrado com o Moodle e o sistema QUESTOURnament que permite o desenvolvimento de competições de programação. Além disso, um repositório de problemas, denominado crimsonHex, foi criado de modo a melhorar a acessibilidade e uso dos problemas (EACEA, 2010).

3.2 Ambientes *online* de aprendizagem de programação

Uma análise dos anais dos dois maiores eventos sobre Informática e Educação (Workshop sobre Educação em Computação e Simpósio Brasileiro de Informática na Educação) feita por com Pereira Júnior e Rapkiewicz (2004) revelou que a comunidade científica brasileira está constantemente procurando soluções para superar as dificuldades encontradas no processo de ensino e aprendizagem de disciplinas básicas de computação.

Nesse contexto, Ferrandin e Stephani (2005) descrevem o WEB-UNERJOL, uma ferramenta para ensino de programação usando a Web. O objetivo dessa ferramenta é diminuir as dificuldades encontradas pelos estudantes quando estão aprendendo programação e possibilitar interação entre professores e alunos fora da sala de aula. A ferramenta apresentada permite que professores possam gerenciar alunos, verificar as versões dos algoritmos enviados ao ambiente pelos discentes e fazer correções e comentários sobre estes algoritmos. Utilizando o WEB-UNERJOL, os alunos podem testar e armazenar versões dos seus algoritmos além de visualizar as correções feitas pelo docente.

Nobre e Menezes (2002) apresentam o AmCorA (Ambiente Cooperativo de Aprendizagem), desenvolvido no Departamento de Informática da Universidade Federal do Espírito Santo, que possui ferramentas apropriadas que apoia tanto professores quanto alunos no desenvolvimento de cursos. Entre as diversas ferramentas incluídas no ambiente podem-se citar as ferramentas Sábia (auxilia a revisão bibliográfica), o Timoneiro (constrói interfaces baseadas em mapas conceituais e fornece um navegador para esses mapas) e o Qsabe (atua como distribuidor inteligente de mensagens e cataloga dinamicamente, a partir de um serviço de perguntas e respostas, o perfil de colaboradores). Além dessas ferramentas, o ambiente disponibiliza a geração de relatórios que podem conter, entre outras informações, número de acessos, quantidade de documentos armazenados e informações para acompanhamento de usuários.

3.3 Comparativo dos trabalhos relacionados

Os trabalhos que mais se assemelham com o JOnline são os trabalhos apresentados por Kurnia, Lim e Cheang (2001), Yi et al (2010), Li et al (2012), Chen et al (2012) e o EduJudge (ECEA, 2010). Esses trabalhos, assim como o JOnline, utilizam um juiz *online* para

apoiar a realização de atividades de disciplinas em computação auxiliando no aprendizado de programação de computadores.

As principais diferenças entre o JOnline e os trabalhos relacionados apresentados é a disponibilização da programação colaborativa. Essa funcionalidade possibilita aos alunos facilidades para a construção de programas em grupo. Outra diferença é o uso de serviços Web RESTful facilitando a integração do JOnline com outras aplicações.

4 PROJETO DO JONLINE

O JOnline é composto por um juiz *online* que permite que os estudantes possam enviar seus códigos-fonte que resolvem algum problema disponível no ambiente para serem avaliados. Após a avaliação do programa, o usuário pode observar a quantidade de memória que o programa consumiu além da quantidade de tempo necessária pra executar os casos de teste do problema. Além da avaliação de programas, o JOnline acrescenta as seguintes funcionalidades:

- Resultados que facilitem aos estudantes a correção de seus erros: o JOnline apresenta os casos de testes (entradas) que geraram resultados errados (incompatíveis com a resposta esperada) ou falhas na execução normal do programa de modo a auxiliar os discentes a encontrarem seus erros.
- Programação colaborativa: essa funcionalidade permite que os estudantes criem projetos que podem possuir códigos-fonte em uma dada linguagem de programação. Cada projeto pode ser compartilhado com outros discentes tornando possível o desenvolvimento de programas em grupo. Dessa forma, os estudantes dispõem de uma interface gráfica que apresenta o código-fonte e um bate-papo para que possam discutir suas ideias.
- Adição de disciplinas e turmas: o JOnline permite o cadastro de disciplinas que podem possuir diversas turmas lecionadas por um professor. Uma turma pode ser usada pelo professor para disponibilizar materiais utilizados em sala (arquivos), enviar avisos aos discentes e atribuir tarefas com um prazo de entrega definido. As turmas dispõem ainda de fóruns permitindo que os participantes possam discutir assuntos acerca do conteúdo da disciplina.
- Gerenciamento de problemas: os professores podem cadastrar problemas e definir quais são as entradas de teste e as saídas esperadas do problema. Se o professor dispor de entradas de teste e da solução do problema escrita em alguma das linguagens suportadas pelo JOnline, então ele pode gerar os resultados esperados.
- Votação do nível de dificuldade dos problemas: cada problema disponibilizado pelo ambiente pode ser avaliado pelos usuários do JOnline de modo a indicar o seu grau de dificuldade. O nível de dificuldade é dado

numa escala de um a cinco em que o nível um indica um problema fácil e o nível cinco um problema desafiador.

- Classificação de problemas: para facilitar o acesso aos problemas pelos usuários, os problemas podem ser pesquisados por assunto abordado ou quanto ao seu grau de dificuldade.

Para agregar as funcionalidades acima, o JOnline é desenvolvido usando serviços Web RESTful visto que esse tipo de serviço possui a vantagem de ser mais leve (consomem menos *bytes* por mensagem) e de suportar o retorno de dados em diversos formatos (JSON, XML, HTML, texto puro, entre outros), conforme discutido no capítulo 3. Por conseguinte, a arquitetura do JOnline é composta de quatro partes: interface com o usuário (páginas Web), serviços Web, banco de dados e o núcleo. Na figura 3 é apresentada a arquitetura do ambiente.

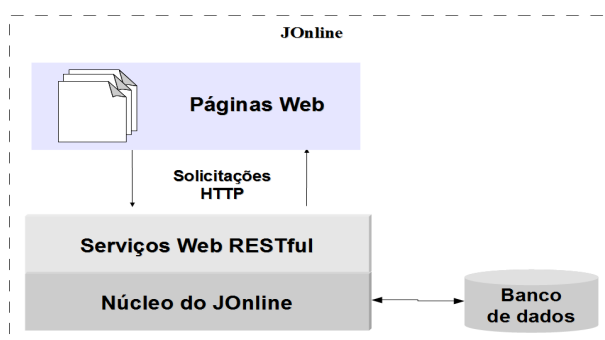


Figura 3 Arquitetura do JOnline

Conforme é possível observar na figura 3, os usuários do sistema interagem com o ambiente através de páginas Web. Nessas páginas são feitas requisições aos serviços Web RESTful que analisam essas requisições e as encaminham para o processamento pelo núcleo do JOnline. O núcleo do JOnline é responsável por se comunicar com o banco de dados e gerenciar as execuções dos programas dos usuários. O banco de dados do sistema segue o modelo relacional possuindo as tabelas apresentadas na figura 4.

O banco de dados mostrado na figura 4 foi modelado de modo a atender os requisitos de projeto descritos anteriormente. É importante ressaltar que a tabela *Token* do banco de dados é utilizada para controlar o acesso aos serviços Web do sistema. Sendo assim, quando um usuário se autentica no JOnline um *token* é criado e será usado durante a sessão do usuário como forma de identificação. O *token* é um Identificador Único Universal (UUID – *Universally Unique Identifier*) gerado automaticamente e é descartado toda vez que a sessão do usuário é finalizada.

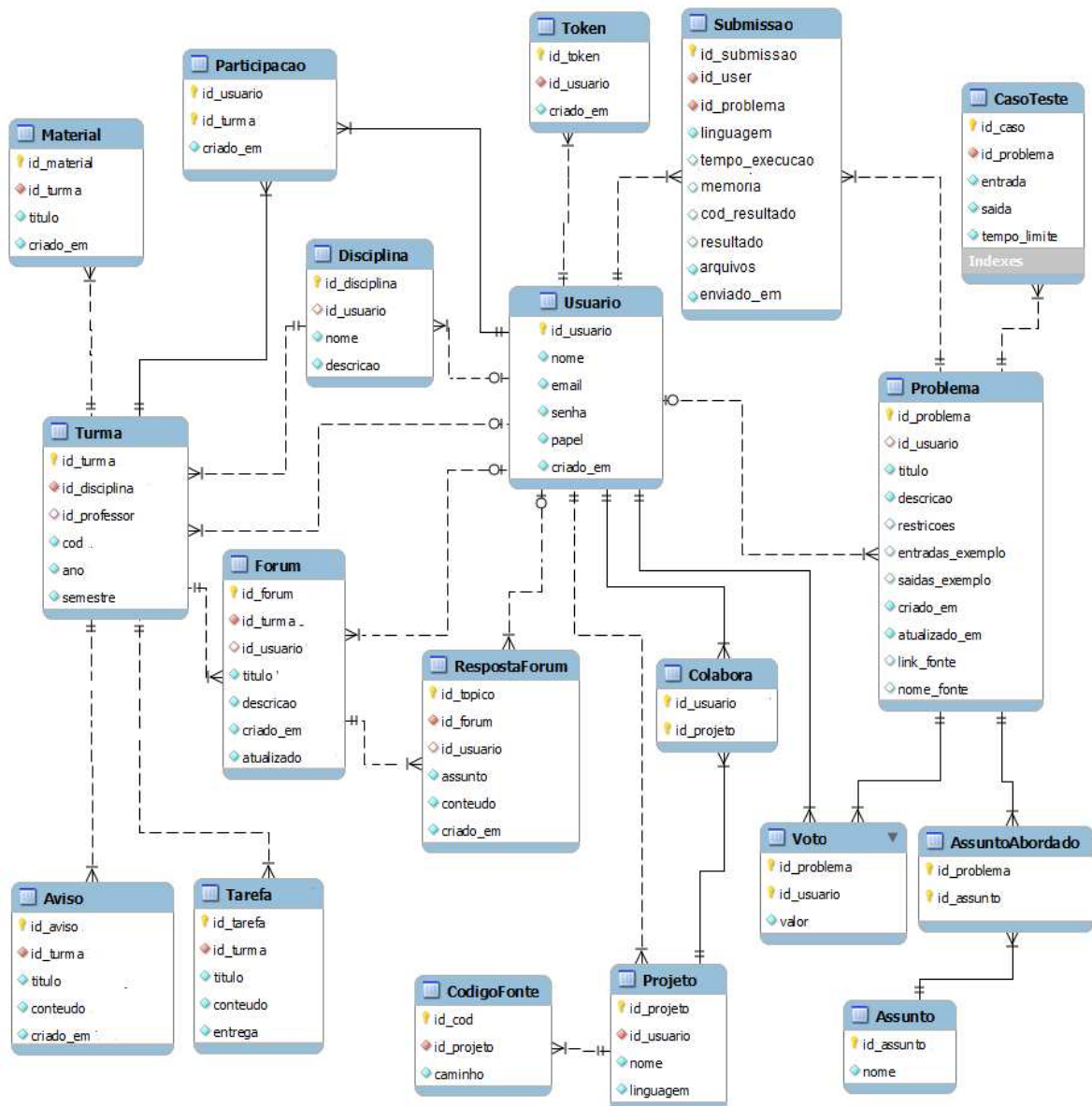


Figura 4 Modelo entidade relacionamento do banco de dados do JOnline

Os usuários do JOnline são classificados em três papéis: aluno, professor e administrador. Essa distinção de papéis é usada para verificar quais são as permissões que um usuário possui para cada um dos módulos do sistema.

O papel “estudante” é atribuído por padrão para todos os usuários que se cadastram no JOnline. Esse papel permite que o usuário possa enviar seus códigos-fonte visualizando o resultado da submissão. Os estudantes podem ainda se inscrever em disciplinas e, assim, obter material, visualizar tarefas e avisos além de participar dos fóruns da disciplina. Esse papel permite também que o usuário vote no grau de dificuldade dos problemas e possa utilizar o

ambiente para construir programas em grupo com outros discentes. Na figura 5 é apresentado o diagrama de casos de uso para um estudante.

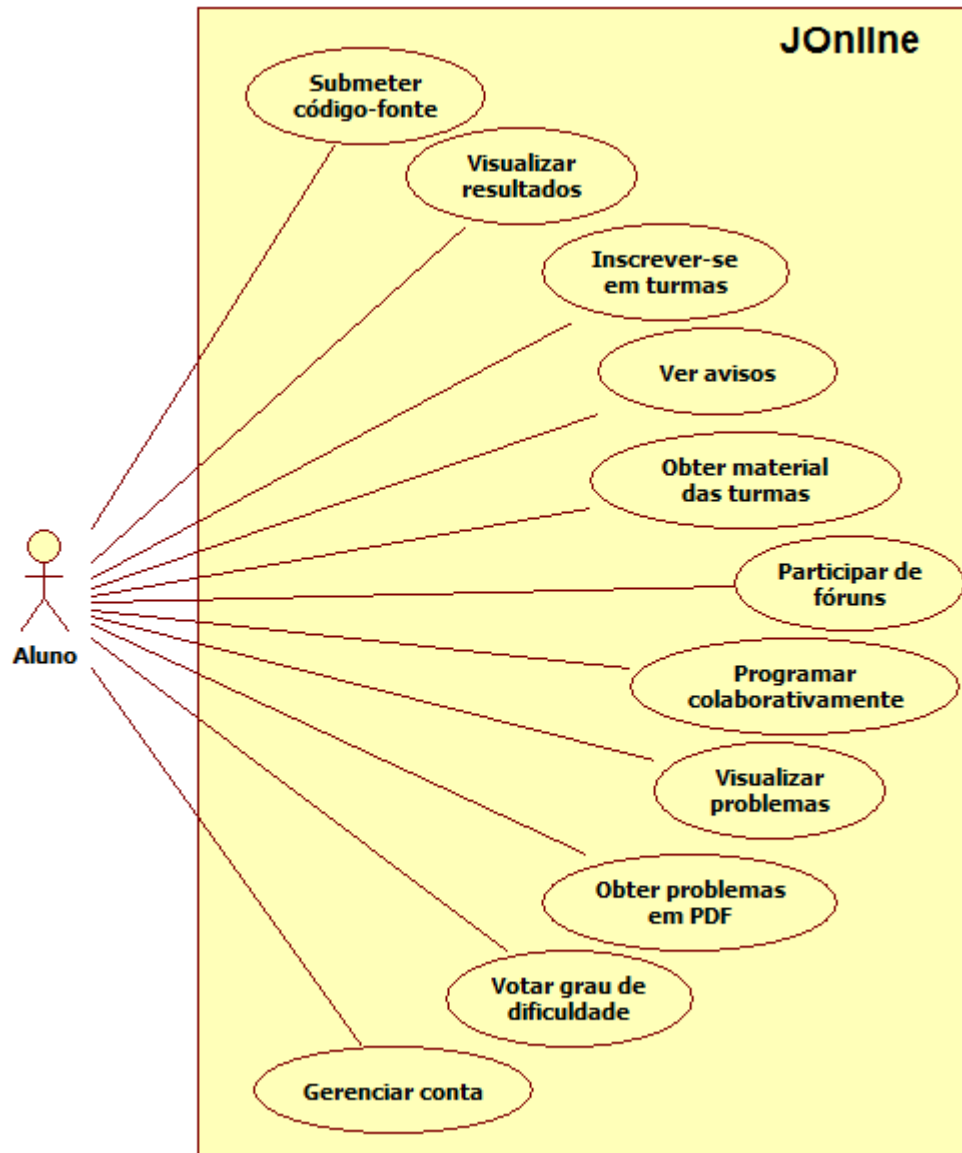


Figura 5 Diagrama de casos de uso de um usuário com o papel “aluno”

O tipo “professor”, além de possuir as permissões do papel estudante, está autorizado a gerenciar as disciplinas e os problemas criados pelo próprio usuário. Esse papel pode ainda gerenciar as turmas as quais o mesmo leciona.

O papel “administrador” é responsável por todo gerenciamento dos módulos do ambiente. Esse papel permite que um indivíduo gerencie problemas, disciplinas e turmas semelhante a um professor. No entanto, um administrador também tem a permissão de gerenciar usuários cadastrados no JOnline. Na figura 6 é apresentado um diagrama de casos

de uso UML indicando quais são as interações que um professor e um administrador efetuam com o JOnline.

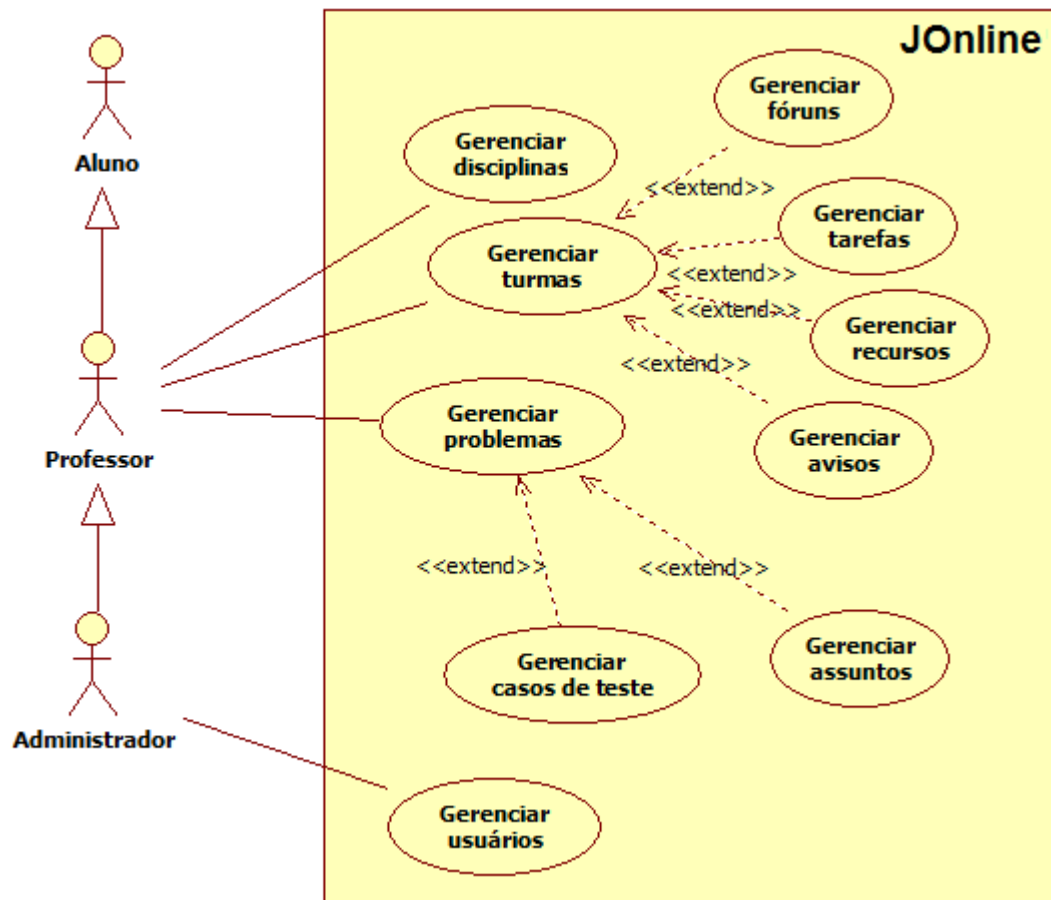


Figura 6 Diagrama de casos de uso de um professor e um administrador

Um indivíduo que não esteja cadastrado no JOnline pode somente acessar conteúdo sobre o ambiente, os problemas e as disciplinas disponíveis. As demais funcionalidades só podem ser realizadas por usuários autenticados ao sistema.

Tendo em vista que os programas dos usuários irão executar remotamente, é estabelecida uma fila de submissão de modo a garantir que somente um programa por vez seja executado no servidor. Isso evita que o tempo de execução medido para o programa do usuário seja afetado por outros programas que estejam executando no servidor.

Um fator que é levado em consideração no projeto do JOnline é a necessidade de criar um ambiente de execução seguro evitando que os usuários enviem código malicioso ao sistema. Para garantir isso, é possível usar um programa, executando com privilégios do usuário administrador do sistema operacional do servidor (*root*), que recebe como argumentos nomes de programas a serem executados e monitora o consumo de memória, o tempo de execução do programa do usuário e um possível comportamento malicioso. Esse programa

deve interromper a execução de programas que excedam a quantidade de tempo limite ou a quantidade de memória máxima permitida. Além disso, também devem ser encerrados programas com suspeita de comportamento malicioso.

5 ASPECTOS DE IMPLEMENTAÇÃO

Nesse capítulo são descritas questões de implementação apresentando as principais tecnologias usadas durante o desenvolvimento do JOnline. A discussão dessas tecnologias é dividida em duas seções: uma dedicada às que foram usadas no lado do servidor e outra para as utilizadas no lado do cliente.

5.1 Tecnologias do lado do servidor

A linguagem usada no desenvolvimento do JOnline no lado do servidor foi a linguagem de programação Java. Esta linguagem foi escolhida por possuir uma vasta quantidade de APIs para auxiliar o programador no desenvolvimento de suas aplicações. Dentre essas APIs se pode citar a Apache Taglibs, Apache Commons, Grizzly WebSockets, Itext, Jersey, MySQL Connector/J e SimpleCaptcha que foram utilizadas nesse trabalho.

A Apache Taglibs (APACHE SOFTWARE FOUNDATION, 2013b) é uma API que implementa a especificação JSTL (*JSP Standard Tag Library*) (ORACLE, 2013a). Essa especificação padroniza o uso de diversas marcações (*tags*) customizadas que encapsulam funcionalidades comumente usadas durante o desenvolvimento de páginas JSP (*Java Server Pages*), tais como iteração por listas, condicionais e internacionalização. Dessa forma, essas *tags* foram usadas nesse trabalho para disponibilizar o JOnline em mais de um idioma e separar a lógica de negócio da visualização das páginas.

Além das *tags* existentes na especificação JSTL, também foram criadas *tags* próprias para tornar a criação de páginas do sistema um trabalho menos propenso a erros, evitar código-fonte duplicado e facilitar a manutenção dos *layouts* das páginas Web. O objetivo dessas *tags* é criar um modelo (*template*) de página global de modo que as demais páginas incluam esse modelo e somente adicionem o conteúdo específico da sua página. Para isso, foram criadas três *tags* (*template:get*, *template:insert* e *template:put*), baseadas na implementação apresentada por Geary (2000). Os atributos existentes para cada uma dessas *tags* podem ser vistos no quadro 4.

Tag	Atributos
Get	name – nome de identificação para o bloco de conteúdo.
	value – valor padrão caso nenhum valor tenha sido especificado
Insert	template – URI do modelo que a página está utilizando
Put	name – nome do bloco de conteúdo do modelo ao qual essa <i>tag</i> irá inserir dados
	content – esse atributo contém uma URI para outra página JSP que possui o conteúdo a ser inserido no modelo. Caso esse atributo não exista na <i>tag</i> então o conteúdo inserido são os dados existentes no corpo dessa <i>tag</i> .

Quadro 4 Estrutura das tags *template:get*, *template:put* e *template:insert* criadas

A tag *template:get* define locais da página modelo em que o conteúdo pode ser adicionado dinamicamente. Dessa maneira, páginas que usem esse modelo devem utilizar as tags *template:put* e *template:insert* para inserir conteúdo dentro desses locais. A tag *template:insert* é usada para especificar qual o modelo que será utilizado, enquanto que o *template:put* especifica qual é a parte do modelo que o conteúdo deverá preencher.

No quadro 5 é possível observar um exemplo de uso dessas tags. Nesse código-fonte a página modelo *template.jsp* (localizada no diretório /WEB-INF/) define três locais em que se pode ser adicionado conteúdo. Esses locais foram nomeados como *title*, *empty_slot_head* e *main_content*. Em particular, o bloco *title* tem como valor padrão o nome 'JOnline' se nenhum valor for atribuído ao mesmo.

```
<%@taglib uri='/WEB-INF/template.tld' prefix='template'%>
<html>
  <head>
    <title><template:get name='title' value='JOnline' /></title>
    <template:get name='empty_slot_head' />
  </head>
  <body>
    <div>
      <div id="nav"><!-- Barra de navegação superior --></div>
      <div id="content"><template:get name='main_content' /></div>
    </div>
    <div id="footer"><!-- Rodapé da página --></div>
  </body>
</html>
```

Quadro 5 Exemplo de página modelo usando as tags criadas

No quadro 6 é mostrada uma página que referencia a página modelo criada e usa as *tags* `template:insert` e `template:put` para inserir conteúdo na mesma. No trecho de código mostrado nessa tabela, visualiza-se que somente o conteúdo *main_content* foi preenchido.

```
<%@taglib uri='/WEB-INF/template.tld' prefix='template'%>
<template:insert template='/WEB-INF/template.jsp'>
    <template:put name='main_content'>Olá mundo!</template:put>
</template:insert>
```

Quadro 6 Exemplo de página que utiliza um modelo de página criado com as *tags* implementadas

A API Apache Commons (APACHE SOFTWARE FOUNDATION, 2013a) contém diversos componentes reusáveis em Java. Desses componentes, foram utilizados no JOnline o de envio de e-mails e o de recebimento de arquivos pelo servidor Web.

Para criar o *chat* da programação colaborativa (que será mostrado na seção 6.5) foi necessário estabelecer uma maneira de permitir que o servidor encaminhasse mensagens de *chat* de um cliente para os demais participantes do *chat*, ou seja, o servidor deve ser capaz de enviar dados a qualquer momento para um ou mais clientes conectados. Essa característica não é suportada pelo protocolo HTTP por este ser um protocolo de requisição e resposta.

Uma maneira de atingir esse objetivo é utilizar uma tecnologia do HTML5 chamada de WebSockets. Essa tecnologia estabelece uma conexão TCP persistente entre servidor e navegador Web permitindo que ambas as partes enviem e recebam dados a qualquer momento (HICKSON, 2012).

A especificação da API de WebSockets exige que tanto o navegador Web usado pelo usuário quanto o servidor Web da aplicação implementem as interfaces presentes nessa especificação. Dessa forma, foi usada a API Grizzly Websockets (PROJECT GRIZZLY, 2013) que lida com as questões do protocolo do WebSockets e fornecem interfaces Java para o programador implementar sua lógica de negócio.

No capítulo 6 deste trabalho será mostrado que os problemas disponíveis no JOnline podem ser obtidos pelos usuários em formato PDF. Sendo assim, foi necessário utilizar uma API para gerar arquivos nesse formato. A API utilizada no JOnline foi a Itext® (1T3XT BVBA, 2013).

Conforme descrito no capítulo 4 deste trabalho, o JOnline disponibiliza as suas funcionalidades em serviços Web RESTful. Para desenvolver esses serviços foi utilizada a API Jersey (ORACLE, 2013b) que disponibiliza um conjunto de classes tanto para o lado do

servidor (criação de serviços) quanto para o lado cliente (consumir serviços). No JOnline a API foi usada somente para criar os serviços Web RESTful.

Os serviços Web do JOnline processam as mensagens HTTP e realizam alterações nos dados do sistema. Esses dados estão armazenados em um banco de dados relacional gerenciados pelo MySQLTM. Por conseguinte, foi necessária a utilização do MySQL Connector/J (ORACLE, 2013c) para estabelecer a comunicação entre a aplicação e o banco de dados.

No formulário de cadastro de usuários do JOnline, foi colocado um campo de CAPTCHA (*Completely Automated Public Turing test to tell Computers and Humans Apart*) de modo verificar se o cadastro da conta de usuário está sendo realizado por um ser humano e não por um robô e, assim, é evitada a criação de contas de usuários falsas por programas mal intencionados. Esse CAPTCHA foi implementado usando a biblioteca SimpleCaptcha (CHILDERS, 2011) que pode gerar tanto áudio quanto imagens para serem utilizadas nessas verificações. No JOnline foi utilizada somente a opção de geração de imagens com caracteres distorcidos para fazer essa verificação.

Além das tecnologias citadas acima, foi preciso criar mecanismos que provesses a execução de programas de usuários de modo que estes não prejudiquem a integridade o servidor. Esse mecanismo de segurança foi alcançado através do uso de um programa, chamado *safeexec*, que cria um ambiente de execução segura. O *safeexec* é um programa de código aberto escrito em C que faz parte do sistema BOCA (CAMPOS; FERREIRA, 2004). Desse modo, após os programas serem devidamente compilados e gerados os executáveis, estes são passados como parâmetros para o *safeexec* que se encarrega de executá-los e monitorar o tempo e memória consumidos pelo programa.

O *safeexec* é um programa que pode ser executado usando linha de comando do sistema operacional e pode ser configurado através de parâmetros especificados durante a execução do mesmo. Os argumentos que podem ser passados a esse programa podem ser vistos no quadro 7. Observando esse quadro nota-se que o *safeexec* permite o isolamento dos programas enviados para execução das pastas do servidor através do comando *chroot*³ disponível nos sistemas operacionais do tipo Unix.

³ Utilizando esse comando é possível bloquear programas de acessarem diretórios fora de uma hierarquia de pastas específica do sistema de arquivos

Opções	Função	Valor padrão
-c	Tamanho máximo de arquivos de despejo	0
-f	Tamanho máximo do arquivo (executável)	131072 Kbytes
-F	Número máximo de arquivos	64
-d	Quantidade máxima de memória que pode ser consumida pelo processo	131072 Kbytes
-m	Quantidade máxima de memória residente	131072 Kbytes
-u	Quantidade máxima de filhos criados pelo processo	64
-t	Valor máximo de tempo de CPU	5 segundos
-T	Valor de tempo máximo real	30 segundos
-C	Diretório atual	Diretório atual
-R	Diretório raiz (caso a opção <i>chroot</i> esteja habilitada)	Nenhum
-n	Habilitar modo <i>chroot</i>	0
-r	Número de execuções	1
-i	Arquivo da entrada padrão	Não definido
-o	Arquivo da saída padrão	Não definido
-e	Arquivo da saída de erro padrão	Não definido
-U	Identificador do usuário do sistema operacional ao qual o programa irá executar	-1
-G	Identificador do grupo do usuário do sistema operacional	-1
-p	Habilita a apresentação do tempo gasto na execução	1
-a	Desabilita a verificação de todos os processos	Habilitado
-s	Desabilita a verificação da existência de processos filhos isolados	Habilitado
-K	Encerra todos os processos que possuam o mesmo par de identificações de grupo e usuário que estejam executando antes da execução atual	Habilitado
-q	Modo silencioso (não mostra mensagens geradas pelo <i>safeexec</i>)	Desabilitado

Quadro 7 Argumentos disponíveis para o *safeexec*

Nesse contexto, ao executar um programa no JOnline um processo é criado no Java (usando a class *Process*) e este invoca o *safeexec* passando os argumentos necessários para execução do usuário em modo *chroot*. É importante frisar que as linguagens Java e Python

necessitam de uma máquina virtual e um interpretador, respectivamente, para executar programas escritos nessas linguagens. Dessa maneira, para programas de usuários escritos nessas linguagens, o *safeexec* é usado para executar a máquina virtual do Java ou o interpretador.

5.2 Tecnologias do lado do cliente

Para auxiliar o desenvolvimento do *layout* das páginas Web e a manipulação de elementos HTML dinamicamente, foram utilizados o *framework* Bootstrap (BOOTSTRAP, 2013) e a biblioteca JavaScript jQuery (JQUERY FOUNDATION, 2013).

O Bootstrap é um *framework* largamente utilizado em sistemas Web que disponibiliza arquivos de estilo (CSS – *Cascading Style Sheets*) e *scripts* escritos em JavaScript que customizam a aparência dos componentes HTML (botões, formulários, *links*, entre outros). Esse *framework* é escalável e, por possuir o seu código-fonte aberto, pode ser alterado de acordo com as necessidades do desenvolvedor. Desse modo, é possível adicionar outros componentes a esse *framework*.

No JOnline foram acrescentados ao Bootstrap o tema Cerulean (PARK, 2013), o BootBox (PAYNE, 2013) e o Bootstrap Datetime Picker (ARRUDA, 2013). O Bootbox é uma biblioteca JavaScript que automatiza o processo de criação de caixas de diálogo. O Bootstrap Datetime Picker, por sua vez, é um *plugin* que formata o campo de texto como uma data e ainda permite que o usuário possa escolher uma data a partir de uma caixa de diálogo com um calendário.

A biblioteca jQuery escrita em JavaScript lida com questões de heterogeneidade entre navegadores Web fornecendo uma maneira uniforme para manipulação de elementos HTML, gerenciamento de eventos e realização de requisições AJAX (*Asynchronous Javascript and XML*) de maneira simplificada. As diferenças entre navegadores Web são encapsuladas com o uso de seletores que especificam quais são os elementos HTML que devem ser escolhidos para serem feitas alguma alteração.

Embora as bibliotecas acima simplifiquem a manipulação de elementos HTML e a definição de *layout*, nenhuma delas trata diretamente com a questão de criar uma área de texto que colore um determinado conjunto de palavras (semelhante aos editores de texto existentes em ambientes de desenvolvimento integrado). À vista disso, foi utilizado o editor de texto CodeMirror (CODEMIRROR, 2013) que suporta a coloração de palavras reservadas para

diversas linguagens de programação. O CodeMirror foi configurado através de parâmetros passados a uma função JavaScript e colocado na página de edição de código-fonte do módulo de programação colaborativa.

Por fim, o MobWrite (MOBWRITE, 2013) foi usado no módulo de programação colaborativa para prover a sincronização e a resolução de conflitos durante a alteração de código-fonte por mais de um usuário. A utilização desse serviço de colaboração ocorre da seguinte maneira: todos os usuários que desejam compartilhar algum conteúdo devem ter em suas páginas Web uma área de texto (*textarea*) com o mesmo identificador e necessitam especificar, utilizando JavaScript, que essa área de texto deve ser compartilhada e qual o servidor Web que hospeda a aplicação MobWrite.

O MobWrite lida somente com a questão de sincronização de conteúdo sendo necessário, desse modo, integrar o uso do MobWrite com o editor de texto CodeMirror citado anteriormente. Para isso foi utilizada uma extensão do MobWrite disponível no CodePad (2013). Feito isso, pode-se usar o código mostrado no quadro 8 para que o MobWrite seja utilizado no compartilhamento de conteúdo de maneira colaborativa.

```
//configuração do CodeMirror
editor = CodeMirror.fromTextArea(document.getElementById(textAreaID), {
  lineNumbers: true, //mostra o número das linhas ao lado do código
  indentUnit: 4, //quatro espaços são usados na indentação
  tabSize: 4, //uma tabulação corresponde a quatro espaços
  mode: language //linguagem de programação
});
//identificador exclusivo para o editor de texto
editor.id='IDENTIFICADOR';
//URL que contém a aplicação MobWrite executando
mobwrite.syncGateway = 'http://mobwrite3.appspot.com/scripts/q.py';
//Inicia o compartilhamento de conteúdo
mobwrite.share(editor);
```

Quadro 8 Script colocado no carregamento de uma página Web para configurar o MobWrite e o CodeMirror

6 RESULTADOS

Neste capítulo é mostrado em detalhes cada módulo do sistema e o seu funcionamento. Sendo assim, é dada ênfase nas funcionalidades disponibilizadas pelo JOnline para cada tipo de usuário do sistema (aluno, professor e administrador). Alguns dados apresentados nesse capítulo são baseados em informações cadastradas no SIGAA da UFS.

6.1 Módulo usuários

Ao acessar a página inicial do JOnline (figura 7) uma pessoa pode inserir seus dados de acesso e escolher o idioma do ambiente (atualmente estão disponíveis os idiomas português e inglês). Conforme explicado no capítulo 4, um usuário anônimo pode ainda visualizar alguns conteúdos do ambiente, tais como os problemas, disciplinas e tutoriais.

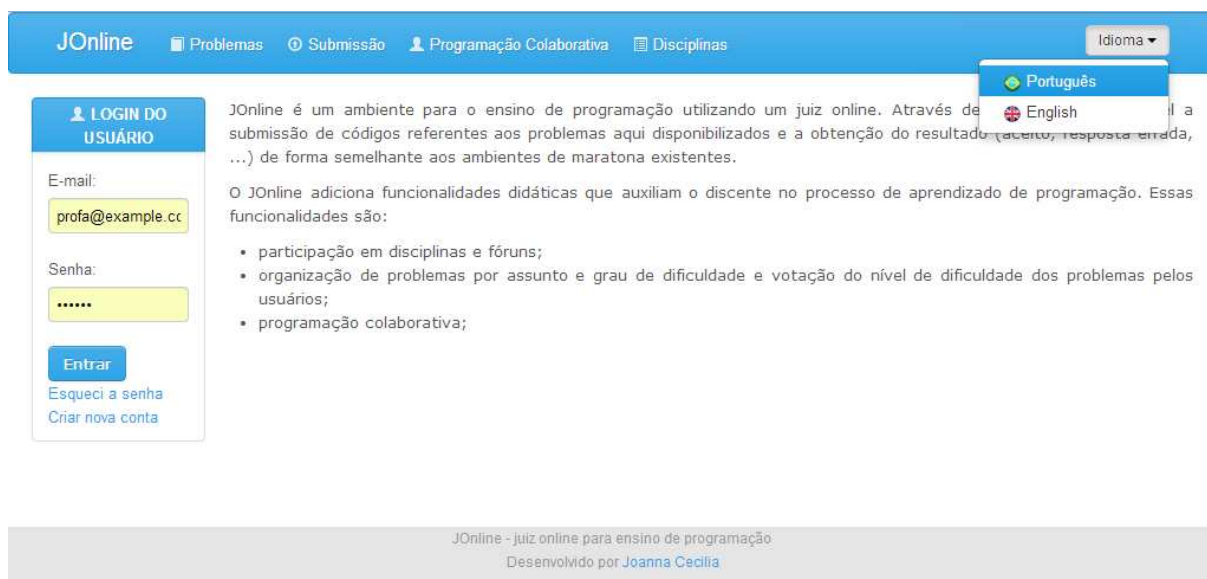


Figura 7 Página inicial do JOnline

As pessoas que ainda não estão cadastradas no JOnline, podem efetuar esse cadastro acessando a opção “Cria nova conta”. Feito isso, é solicitado que o indivíduo insira suas informações básicas (nome completo e e-mail) e os caracteres mostrados na imagem (CAPTCHA). Por questões de segurança, é exigido que a senha escolhida tenha mais de seis caracteres. Caso alguma informação não seja preenchida de maneira adequada, é apresentada uma mensagem de erro ao usuário. Na figura 8 é mostrada a interface Web para criar uma

nova conta no sistema com uma mensagem de erro devido ao não preenchimento do campo de e-mail. Por padrão, cada nova conta inserida no sistema tem o papel “estudante”

Figura 8 Página de cadastro de usuários no JOnline

Feito o cadastro, o usuário pode atualizar suas informações de conta usando a opção “Minha conta” disponível no bloco localizado do lado esquerdo do sistema. Além da redefinição de suas informações, o usuário pode optar por excluir a sua conta, tendo seus dados removidos do banco de dados do sistema. Na figura 9 é apresentada a página na qual um usuário pode gerenciar sua própria conta.

Figura 9 Página de gerenciamento de conta do usuário

Se o usuário esquecer sua senha ele poderá recuperá-la através da opção “Solicitar nova senha” mostrada no bloco “Login do usuário”. Para realizar essa solicitação, o usuário deve especificar seu e-mail cadastrado no JOnline e o CAPTCHA como mostrado na figura 10. Com isso, é enviada uma nova senha gerada aleatoriamente contendo seis caracteres

alfanuméricos para o e-mail do usuário. Na figura 11 é possível ver o e-mail enviado ao usuário para informar a senha gerada pelo sistema.



Formulário para recuperação de senha. O campo "E-mail" contém o endereço "jc_joanna@yahoo.com.br". Abaixo dele, há uma instrução "Insira os caracteres mostrados ao lado" seguida por uma imagem de um código de verificação (captcha) que mostra a sequência "gmmhk6". Abaixo do captcha, há um campo de entrada para a nova senha e um botão azul rotulado "Nova senha".

Figura 10 Formulário para recuperação de senha

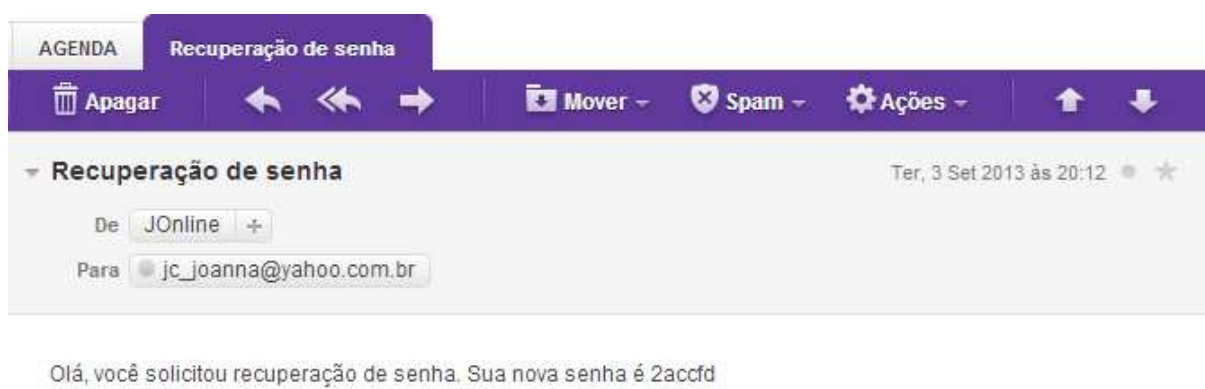


Figura 11 E-mail enviado ao usuário para informar sua nova senha

Quando o usuário tem o papel de administrador, é dada a permissão de alteração de papéis e exclusão de usuários. Para isso, o administrador deve acessar a opção “Usuários” dentro do bloco “Gerenciar” mostrado no lado esquerdo da página. Nessa opção, é apresentada uma lista dos usuários cadastrados sendo possível realizar alterações, conforme é possível ver na figura 12. Nessa figura é mostrado ainda um aviso com um indicativo de que a solicitação de alteração de papel de um usuário foi processada com sucesso.

6.2 Módulo submissão

Para que um usuário tenha sua solução para um dado problema do ambiente avaliada, é necessário que o mesmo tenha feito sua autenticação no sistema (mostrado na seção anterior). Usuários que não estão autenticados não possuem acesso a esse módulo e, assim, é apresentada uma mensagem de erro sugerindo que o usuário forneça seus dados de acesso ou ainda crie uma conta (caso ainda não possua uma).

Gerenciar	Nome completo	E-mail	Papel	Criado em
<input checked="" type="checkbox"/>	Ada Lovelace	ada@example.com	Professor	03-09-2013
<input checked="" type="checkbox"/>	Alan Turing	alanturing@example.com	Administrador	03-09-2013
<input checked="" type="checkbox"/>	Ana Santos	olhodecoruja@hotmail.com	Estudante	03-09-2013
<input checked="" type="checkbox"/>	Blaise Pascal	bpascal@example.com	Professor	03-09-2013
<input checked="" type="checkbox"/>	Charles Babbage	babbage@example.com	Estudante	03-09-2013
<input checked="" type="checkbox"/>	Donald Knuth	knuth@example.com	Professor	03-09-2013
<input checked="" type="checkbox"/>	Edsger Dijkstra	eddijs@example.com	Professor	03-09-2013
<input checked="" type="checkbox"/>	Jhon Von Neumann	neumann@example.com	Estudante	03-09-2013
<input checked="" type="checkbox"/>	Joanna Cecília da Silva Santos	jc_joanna@yahoo.com.br	Estudante	03-09-2013
<input checked="" type="checkbox"/>	Linus Torvalds	linux@example.com	Estudante	03-09-2013

Figura 12 Interface para gerenciamento de contas de usuários do JOnline

Para enviar sua solução para ser avaliada, o usuário deve primeiramente especificar qual dos problemas ele está resolvendo através de uma busca por nome dos problemas. Na figura 13 é possível ver o usuário buscando problemas que possuam no título a palavra “coelhos”. Essa busca apresentou como resultado o problema com o título “população de coelhos” cuja descrição pode ser vista na figura 14. Esse problema pode ser resolvido calculando uma sequência de números, chamada de sequência de Fibonacci, em que os dois primeiros termos da sequência são iguais a 1 e cada termo subsequente é resultado da soma dos dois termos precedentes ao número (KOSHY, 2001).

Figura 13 Busca de problemas por nome

População de coelhos

[Submeter solução](#) [Download](#)
Descrição:

Suponha que em um ambiente totalmente isolado haja inicialmente um coelho macho e uma coelho fêmea e que se queira saber qual a maior quantidade de pares que poderão existir ao final de um ano. Considere as seguintes regras para o aumento populacional dos coelhos:

- Cada par de coelhos gera no máximo outro par no final de um mês;
- Cada fêmea já pode reproduzir após dois meses depois de seu nascimento;
- Os pares não se alteram, ou seja, só há cruzamento entre os pares;
- O primeiro par de coelhos completou dois meses de idade no final do primeiro mês;

Dessa forma, escreva um programa que dada uma sequência de valores correspondendo o mês, imprima pra cada um deles a quantidade de coelhos existentes seguida de uma quebra-de-linha (\n). O final da sequência de números é indicado por um valor menor ou igual a zero.

Figura 14 Descrição do problema “População de coelhos”

Após escolher o problema, o usuário deve especificar a linguagem de programação utilizada, bem como os códigos-fonte para serem avaliados, como mostrado na figura 15. Conforme é possível observar na figura 15, o JOnline permite que o usuário possa enviar mais de um código-fonte ao ambiente. Isso é útil quando, por exemplo, um usuário separa as assinaturas das funções das suas respectivas implementações em arquivos diferentes.

Submissão

Problema

Linguagem

Código-fonte

coelhos_2.h

coelhos_2.c

Figura 15 Página de submissão de código-fonte

Atualmente, esse módulo suporta a execução de programas escritos em cinco linguagens de programação: C, C++, Java, Pascal e Python. É necessário destacar que no caso de programas escritos usando Python e Java, o usuário deve ter um arquivo nomeado main.py e Main.java, respectivamente, para indicar qual é o código-fonte que possui a função principal. Esse requisito não é necessário se a submissão somente tiver um código-fonte.

Após preencher as informações necessárias, a pessoa visualizará se a sua solução está ou não correta. A solução do usuário está incorreta quando uma das seguintes situações ocorre: os códigos-fonte não compilaram com sucesso, o programa apresentou um resultado

diferente do esperado, houve uma falha durante a execução do programa (término inesperado do programa), o programa demorou muito tempo executando ou a quantidade de memória consumida excedeu o limite máximo.

No primeiro caso (erros de compilação) as mensagens de erro geradas pelo compilador são capturadas e apresentadas ao usuário. Na figura 16 é possível ver um erro de compilação para um código-fonte escrito usando a linguagem C. Esse código-fonte, mostrado no quadro 9, está com um erro de compilação devido à ausência de um ponto-e-vírgula ao fim de uma instrução presente na função *fibonacci*.

Problema: População de coelhos

Linguagem: C

Resultado:

coelhos_sem_cabecalho.c: Na função 'fibonacci':
coelhos_sem_cabecalho.c:15:1: erro: expected ';' before '}' token

Figura 16 Aviso mostrado ao usuário indicando que há um erro de compilação no seu código-fonte

```
#include <stdio.h>
int fibonacci(int n);
int main(){
    int N, i;
    scanf("%d", &N);
    while(N>0){
        scanf("%d", &i);
        printf("%d\n", fibonacci(i));
        N--;
    }
}
int fibonacci(int n){
    if(n==0 || n==1)
        return 1;
    return fibonacci(n-1) + fibonacci(n-2)
}
```

Quadro 9 Código-fonte em linguagem C com erro de compilação

Após corrigir o erro de compilação citado anteriormente, o usuário novamente envia a sua solução e poderá verificar que a mesma está correta, conforme mostrado na figura 17. Além disso, é mostrada também a quantidade de tempo (em segundos) e memória (em *bytes*) gasta durante a execução do programa.

Problema: População de coelhos

Linguagem: C

Resultado:

```
Correto
Tempo: 0 s
Memória: 453 bytes
```

Figura 17 Resultado correto para o problema “População de coelhos”

Nas situações em que o programa compila e, no entanto, está incorreto é apresentado ao usuário qual foi o caso de teste que causou isto. Mostrar esse caso de teste facilita que o usuário encontre o seu erro. Na figura 18 é possível ver uma submissão em que houve uma falha durante a execução e o caso de teste que causou essa falha.

Problema: População de coelhos

Linguagem: C

Resultado:

```
Programa falhou ao executar o caso de teste 2:
2
2
3
```

Figura 18 Resultado incorreto com o caso de teste que causou a falha

Em alguns casos a saída do programa só difere dos resultados esperados por uma diferença de espaços em branco ou quebras de linha. Nesse caso em específico é dito ao usuário que a saída do programa está mal formatada e novamente é mostrado o caso de teste que gerou essa saída incorreta, como pode ser visto na figura 19.

Problema: População de coelhos

Linguagem: C

Resultado:

```
Saída mal formatada para o caso de teste # 1:
2
0
1
```

Figura 19 Saída mal formatada para um programa escrito em C

6.3 Módulo problemas

Qualquer pessoa (independente se já efetuou ou não a sua autenticação ao sistema) pode visualizar os problemas cadastrados no ambiente. Desse modo, todos podem visualizar uma listagem de problemas disponível, vendo algumas informações do problema, tais como título, assuntos abordados (*tags*), nível de dificuldade, porcentagem de respostas corretas enviadas até o momento e as datas de criação e atualização do problema. Na figura 20 é possível observar a listagem de problemas disponíveis no JOnline.

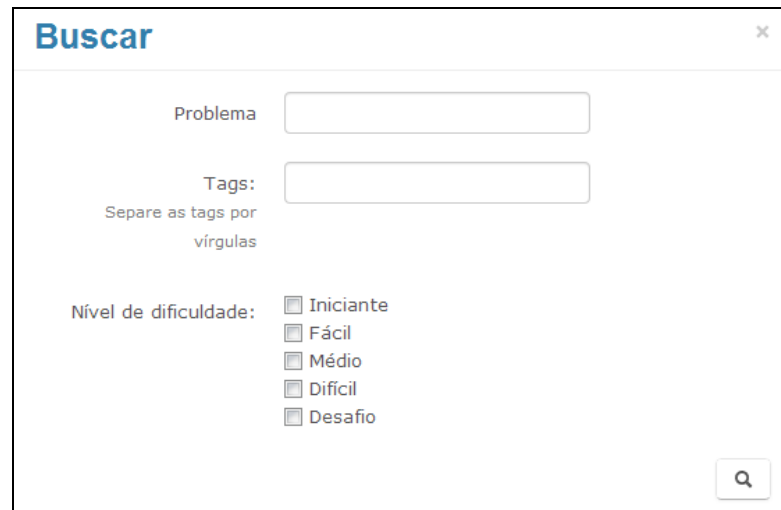
Problema	Tags:	Nível de dificuldade:	Respostas corretas (%)	Criado em:	Última atualização:
Amplitude térmica	condicional, loop	Nenhum voto até o momento	Nenhuma submissão até o momento	09-09-2013	09-09-2013
Conversão de tempo	manipulação de dados	Médio	66,67% (2/3)	05-09-2013	07-09-2013
Duração de uma ligação	condicional	Nenhum voto até o momento	Nenhuma submissão até o momento	09-09-2013	09-09-2013
Escrita de números por extenso	funções, loop	Nenhum voto até o momento	Nenhuma submissão até o momento	05-09-2013	05-09-2013
Impressão em binário	loop	Nenhum voto até o momento	Nenhuma submissão até o momento	09-09-2013	09-09-2013
Inversão de Números	manipulação de dados	Nenhum voto até o momento	Nenhuma submissão até o momento	09-09-2013	09-09-2013
Número de Euler	fatorial, loop, recursividade	Nenhum voto até o momento	Nenhuma submissão até o momento	09-09-2013	09-09-2013
Números quadrados perfeitos	condicional	Nenhum voto até o momento	Nenhuma submissão até o momento	09-09-2013	09-09-2013

Figura 20 Listagem de problemas disponíveis no JOnline

Na figura 20 é possível observar que há um botão para busca de problemas. Ao acionar esse botão, uma caixa de diálogo é mostrada em que o usuário poderá encontrar problemas de acordo com alguns critérios de busca: nome do problema, assunto abordado e nível de dificuldade (figura 21).

Ao escolher um problema, é mostrada ao usuário uma página com o detalhamento do problema. Cada problema possui um título, uma descrição, entradas e saídas de exemplo e restrições que especificam quais os tipos de entrada que serão usados para testar os programas. Um problema pode ter sido obtido de uma fonte externa e, nesse caso, é colocado um *link* para a origem do problema. Nessa página de visualização do problema, como

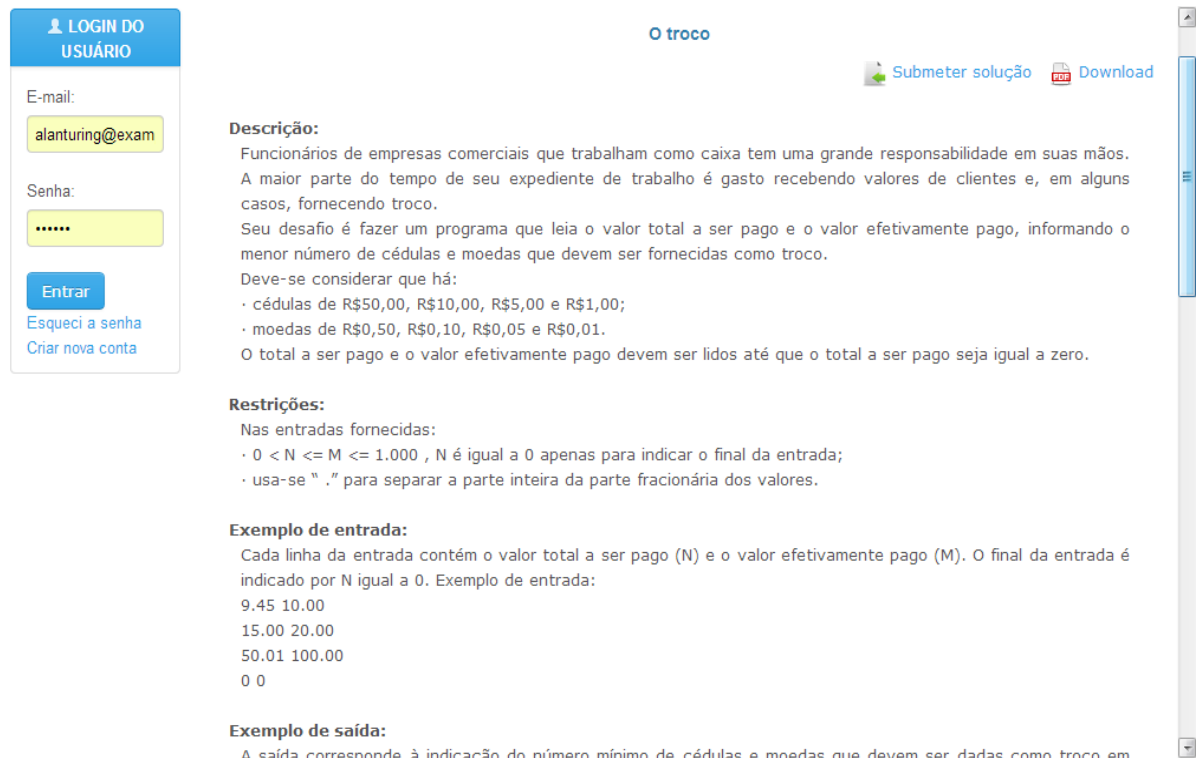
mostrado na figura 22, o usuário pode optar por obter um documento PDF sobre o problema ou ainda submeter sua solução para o mesmo.



A caixa de diálogo intitulada "Buscar" possui um campo de texto para "Problema", um campo de texto para "Tags" com o subtítulo "Separe as tags por vírgulas", e uma seção "Nível de dificuldade" com cinco opções de seleção: "Iniciante", "Fácil", "Médio", "Difícil" e "Desafio". Um botão de lupa está localizado no canto inferior direito.

Figura 21 Caixa de diálogo para buscar problemas

Quando o usuário está autenticado no sistema, ele pode votar o nível de dificuldade do problema através da caixa de diálogo apresentada na figura 23. Cada problema pode ter os seguintes níveis de dificuldades: iniciante, fácil, médio, difícil e desafio. Uma pessoa pode votar várias vezes para o mesmo problema, sendo que o voto mais recente substitui os votos anteriores



A interface de visualização do problema "O troco" apresenta uma barra lateral esquerda com o login do usuário "alanturing@exam" e uma seção de login. O conteúdo principal do problema inclui:

- Descrição:** Funcionários de empresas comerciais que trabalham como caixa tem uma grande responsabilidade em suas mãos. A maior parte do tempo de seu expediente de trabalho é gasto recebendo valores de clientes e, em alguns casos, fornecendo troco. Seu desafio é fazer um programa que leia o valor total a ser pago e o valor efetivamente pago, informando o menor número de cédulas e moedas que devem ser fornecidas como troco. Deve-se considerar que há:
 - cédulas de R\$50,00, R\$10,00, R\$5,00 e R\$1,00;
 - moedas de R\$0,50, R\$0,10, R\$0,05 e R\$0,01.
 O total a ser pago e o valor efetivamente pago devem ser lidos até que o total a ser pago seja igual a zero.
- Restrições:**
 - Nas entradas fornecidas:
 - $0 < N \leq M \leq 1.000$, N é igual a 0 apenas para indicar o final da entrada;
 - usa-se "." para separar a parte inteira da parte fracionária dos valores.
- Exemplo de entrada:** Cada linha da entrada contém o valor total a ser pago (N) e o valor efetivamente pago (M). O final da entrada é indicado por N igual a 0. Exemplo de entrada:


```
9.45 10.00
15.00 20.00
50.01 100.00
0 0
```
- Exemplo de saída:** A saída corresponde à indicação do número mínimo de cédulas e moedas que devem ser dadas como troco em

Na barra lateral direita, há links para "Submeter solução" e "Download".

Figura 22 Visualização do problema "O troco"

Os problemas disponíveis no sistema podem ser gerenciados por dois tipos de usuários: administradores e professores. Entretanto, os professores só possuem permissão de gerenciar os problemas que criaram. Para efetuar esse gerenciamento, os usuários com esses papéis acessam a opção “Problemas” contido no bloco “Gerenciar” mostrado no lado esquerdo de cada página do JOnline.

Figura 23 Votação do nível de dificuldade de um problema

A interface de gerenciamento é composta por duas abas: uma contém uma lista de problemas que o usuário pode alterar e a outra contém um formulário Web para cadastro de novos problemas no sistema (figura 24). Nessa interface também é disponibilizada a opção de busca de problemas (conforme já discutido anteriormente).

Problema	Tags:	Nível de dificuldade:	Respostas corretas (%):	Criado em:	Última atualização:
Amplitude térmica	condicional, loop	Nenhum voto até o momento	Nenhuma submissão até o momento	09-09-2013	09-09-2013
Conversão de tempo	manipulação de dados	Médio	66,67% (2/3)	05-09-2013	07-09-2013
Duração de uma ligação	condicional	Nenhum voto até o momento	Nenhuma submissão até o momento	09-09-2013	09-09-2013
Escrita de números por extenso	funções, loop	Nenhum voto até o momento	Nenhuma submissão até o momento	05-09-2013	05-09-2013
Impressão em binário	loop	Nenhum voto até o momento	Nenhuma submissão até o momento	09-09-2013	09-09-2013
Inversão de Números	manipulação de dados	Nenhum voto até o momento	Nenhuma submissão até o momento	09-09-2013	09-09-2013
Número de Euler	fatorial, loop, recursividade	Nenhum voto até o momento	Nenhuma submissão até o momento	09-09-2013	09-09-2013

Figura 24 Interface Web de gerenciamento de problemas

O cadastro de problemas, apresentado na figura 25, consiste de um formulário Web em que o usuário preenche todas as informações sobre o mesmo. Para auxiliar a especificação dos assuntos abordados pelo problema, foi colocada a funcionalidade de autocompletar nesse campo de formulário. Desse modo, são apresentadas sugestões de assuntos abordados já existentes no ambiente à medida que o usuário digita as palavras nesse campo.

Informações sobre o problema

Título	<input type="text" value="O troco"/>
Descrição	<p>Funcionários de empresas comerciais que trabalham como caixa tem uma grande responsabilidade em suas mãos. A maior parte do tempo de seu expediente de trabalho é gasto recebendo valores de clientes e, em alguns casos, fornecendo troco.</p> <p>Seu desafio é fazer um programa que leia o valor total a ser pago e o valor efetivamente pago, informando o menor número de cédulas e moedas que devem ser fornecidas como troco. Deve-se considerar que há:</p> <ul style="list-style-type: none"> • cédulas de R\$50,00, R\$10,00, R\$5,00 e R\$1,00; • moedas de R\$0,50, R\$0,10, R\$0,05 e R\$0,01.
Restrições	<div style="border: 1px solid #ccc; height: 60px;"></div>
Exemplos de entradas	Cada linha da entrada contém o valor total a ser pago (M) e o valor efetivamente pago (N). O final da entrada é indicado por M igual a 0. A Figura 17 apresenta um exemplo de entrada.
Exemplos de saída	A saída corresponde à indicação do número mínimo de cédulas e moedas que devem ser dadas como troco em cada um dos casos fornecidos na saída. Para cada caso, a saída é formada pelo valor da cédula (precedido de "R\$"), seguido de ":", espaço e do número de cédulas ou moedas que devem ser dados como troco. Separa-se os resultados de casos diferentes com uma linha em branco.
Link original do problema (opcional)	<input type="text" value="http://gravatai.ulbra.tche.br/~roland/pub/list"/>
Nome do site (opcional)	<input type="text" value="Lista de desafios"/>
Tags	<input type="text" value="alg"/> <div style="border: 1px solid #ccc; padding: 2px; margin-top: 5px; background-color: #f0f0f0;"> algoritmo guloso </div>

Figura 25 Formulário Web para cadastro de problemas

Para cadastrar um problema, também devem ser inseridos os arquivos de texto contendo as entradas de teste e as saídas esperadas para cada uma dessas entradas. Para cada par de entrada e saída de teste, deve ser especificado também o tempo máximo (em segundos) para que um programa execute esse par de teste. O formulário para efetuar esse cadastro pode ser visto na figura 26.

Casos de teste

Código-fonte	Entradas de teste	Saídas esperadas
	<div style="border: 1px solid #ccc; padding: 2px; display: inline-block;">Escolher arquivo</div> Nenhum arquivo selecionado	<div style="border: 1px solid #ccc; padding: 2px; display: inline-block;">Escolher arquivo</div> Nenhum arquivo selecionado
	Tempo máximo de execução: <input style="width: 100px;" type="text"/>	
	Clique aqui caso deseje gerar arquivos de saída	<div style="border: 1px solid #ccc; padding: 2px; display: inline-block;">+ Adicionar mais arquivos</div>

Figura 26 Cadastro de casos de teste para um problema

Se o usuário dispõe da solução do problema em alguma linguagem de programação suportada pelo ambiente, então ele pode gerar os resultados esperados a partir de arquivos de entrada e os códigos-fonte da solução, conforme mostrado na figura 27. Para gerar os arquivos com os resultados esperados, o usuário deve acessar a opção “Clique aqui caso deseje gerar arquivos de saída” mostrada na figura 26. Ao solicitar a geração dos resultados esperados, o usuário obterá um arquivo compactado em formato ZIP (PKWARE, 2013).

Linguagem Escolha a linguagem ▼

Código-fonte Escolher arquivo Nenhum arquivo selecionado

+ Adicionar mais arquivos

Entradas de teste Escolher arquivo Nenhum arquivo selecionado

+ Adicionar mais arquivos

Gerar

Figura 27 Formulário para gerar arquivos com os resultados esperados

6.4 Módulo disciplinas

O módulo de disciplinas pode ser usado por qualquer usuário (independente deste estar cadastrado ou não no JOnline) para visualizar as disciplinas existentes no ambiente. Também é permitido o livre acesso ao detalhamento de uma disciplina em específico. Na figura 28 são mostrados o detalhamento de uma disciplina e a listagem de disciplinas disponíveis. É importante ressaltar que na figura 28-b o bloco de autenticação do usuário (*login*) passa a ocupar toda a largura da tela. Isso sempre ocorre toda vez que a largura da janela do navegador Web diminui.

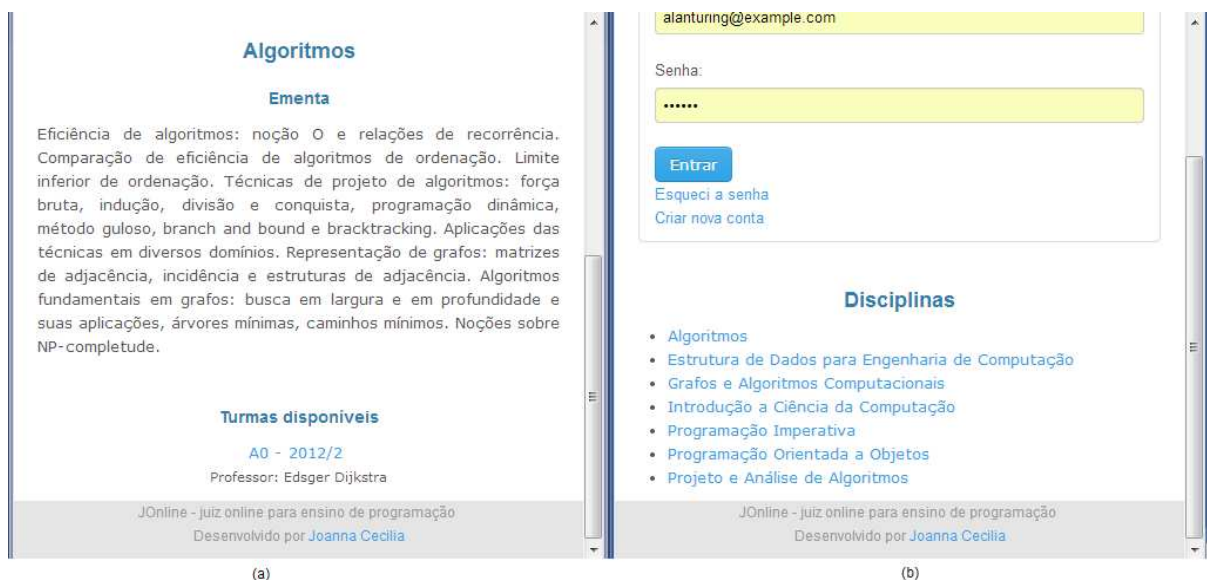


Figura 28 (a) Listagem de disciplinas. (b) Detalhes da disciplina de Algoritmos

Observando a figura 28 nota-se que cada disciplina é composta de várias turmas sob a responsabilidade de um professor. Para visualizar conteúdo relacionado a uma turma em específico, um usuário deve estar inscrito na mesma. Feita essa inscrição ele estará autorizado a acessar os materiais disponibilizados pelo professor, participar de fóruns além de visualizar outros participantes, tarefas e avisos da disciplina. A interface Web para que um usuário faça sua inscrição a uma turma pode ser vista na figura 29.

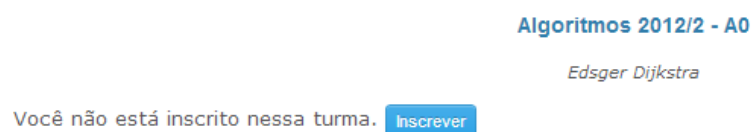


Figura 29 Mensagem de aviso indicando que o aluno não está inscrito na turma

Estando matriculado na disciplina o usuário poderá visualizar o conteúdo da disciplina através da página mostrada na figura 30. Nessa figura é possível ver a lista de arquivos e suas respectivas datas de publicação. A qualquer momento o discente pode cancelar sua inscrição à turma através do botão “Deixar turma” localizado no canto superior direito da página.



Figura 30 Listagem de arquivos disponibilizados pelo professor

Ao acessar a aba “Fóruns” da página da turma, o discente tem a opção de criar um novo fórum ou ainda ver uma lista dos fóruns já existentes, conforme nota-se na figura 31. Ao criar um fórum, o usuário deve especificar o título do fórum e a sua descrição. Caso alguma dessas duas informações não seja preenchida, o sistema mostra uma mensagem de erro ao usuário (figura 32).



Figura 31 Listagem de fóruns da turma

Após o fórum ter sido criado, qualquer participante da disciplina (incluindo o professor) poderá respondê-lo e visualizar as respostas já existentes. Além disso, cada usuário tem a opção de excluir os fóruns e respostas que criou (figura 33). O processo para criar uma resposta em um fórum é semelhante ao de criação de um fórum: deve-se especificar um título para a resposta e o seu conteúdo.

Para facilitar a comunicação entre discentes e docente, o JOnline disponibiliza a opção de cadastro de avisos da turma. Sendo assim, sempre que o professor necessitar que os estudantes tomem conhecimento de alguma situação, ele pode criar um aviso que será enviado como e-mail para todos os participantes da turma. Para permitir que os discentes visualizem os avisos de maneira rápida, é sempre mostrado um quadro com os avisos mais recentes no lado esquerdo das páginas do JOnline (figura 34).

Criar fórum

O título do fórum não foi preenchido ×

Título

Conteúdo

Alguém conseguiu resolver a questão 1 da lista?

Cancelar
Inserir

Figura 32 Mensagem de erro avisando que o título do fórum deve ser preenchido

Recursos
Fóruns
Avisos
Participantes
Tarefas

[Joanna] Dúvida na primeira questão da lista 1

Alguém conseguiu resolver a questão 1 da lista?

[Responder](#)

Criado em: 04/09/2013 08:11 Última atualização: 04/09/2013 08:11

Respostas

[Olga] Transforme a matriz num vetor

Eu tb fiz como a Ana explicou e deu certo.

[Excluir](#)

Criado em: 04/09/2013 09:00

[Ana] Considere a matriz como um vetor

Embora seja uma matriz, você pode lidar como ela como se fosse um vetor e aplicar a busca binária normalmente. Para isso, vc deve indexar a matriz desse jeito: $i*m + j$ em que i é o contador pras linhas e j pras colunas. Espero ter ajudado :D

Figura 33 Estudante “Olga” visualizando um fórum criado pela estudante “Joanna”



Figura 34 Listagem de avisos de uma turma. Destaque para o quadro de avisos mostrado ao usuário

Um docente pode ainda atribuir tarefas para cada turma ao qual é responsável especificando o prazo de entrega e o enunciado da tarefa. Uma vez cadastrada a tarefa, um e-mail é enviado a todos os participantes da turma para que os mesmos tomem conhecimento da nova tarefa. As tarefas podem ser visualizadas acessando a aba “Tarefas” na página da turma (figura 35).



Figura 35 Lista de tarefas para a turma de Algoritmos 2012/2 – A0

O cadastro de disciplinas e suas respectivas turmas são ações designadas aos usuários com o papel de administrador e professor. Desse modo, ambos os usuários podem fazer o gerenciamento de disciplinas e turmas (incluindo seus fóruns, avisos, tarefas e recursos). No entanto, um professor somente pode gerenciar as disciplinas e turmas as quais criou, ficando proibido de excluir disciplinas que possuam turmas cadastradas de outros professores. O administrador, por sua vez, pode gerenciar todas as disciplinas e turmas independente de quem as criou.

Com o intuito de facilitar o gerenciamento das disciplinas e suas respectivas turmas, é adicionado na lateral esquerda do ambiente um atalho para este gerenciamento. Esse atalho referencia uma página que contém a listagem das disciplinas que o usuário pode gerenciar, disponibilizando as opções de edição e exclusão da disciplina, conforme nota-se na figura 36.

















Disciplinas		
Gerenciar		Disciplina
 Editar	 Excluir	Algoritmos
 Editar	 Excluir	Estrutura de Dados para Engenharia de Computação
 Editar	 Excluir	Grafos e Algoritmos Computacionais
 Editar	 Excluir	Introdução a Ciência da Computação
 Editar	 Excluir	Programação Imperativa
 Editar	 Excluir	Programação Orientada a Objetos
 Editar	 Excluir	Projeto e Análise de Algoritmos
 Adicionar		

Figura 36 Gerenciamento de disciplinas

Também é colocado um atalho para que um professor visualize as turmas as quais é responsável. A figura 37 contém um exemplo de listagem das turmas do professor Edsger Dijkstra.

 EDSGER


[Minha Conta](#)

[Minhas turmas](#)

[Sair](#)

AVISOS

Nenhum aviso nos últimos dias

 **GERENCIAR**

[Problemas](#)

[Disciplinas](#)

[Turmas](#)

Minhas turmas

[Algoritmos \(A0 - 2012/2\)](#)

Figura 37 Listagem das turmas de um professor. Destaque para o atalho colocado no bloco "Gerenciar" do sistema

Professores e administradores visualizam a página da turma de maneira diferente da qual um estudante visualiza. As diferenças estão no fato de que esses tipos de usuários podem excluir a turma além de gerenciar os recursos, fóruns, avisos e tarefas, conforme é observado nas figuras 38 a 41.

Algoritmos 2012/2 - A0
Edsger Dijkstra

[✕ Excluir](#)

[Recursos](#) [Fóruns](#) [Avisos](#) [Participantes](#) **Tarefas**

- [Lista 1](#) [Prazo de entrega: 11-09-2013] [✎](#) [✕](#)
- [Exercícios de ordenação](#) [Prazo de entrega: 18-09-2013] [✎](#) [✕](#)

[+ Adicionar](#)

Figura 38 Listagem de tarefas com as opções de edição, exclusão e cadastro

Algoritmos 2012/2 - A0
Edsger Dijkstra

[✕ Excluir](#)

Recursos [Fóruns](#) [Avisos](#) [Participantes](#) [Tarefas](#)

- [Graph Theory.pdf](#) [04-09-2013] [✕](#)
- [revisao.pdf](#) [04-09-2013] [✕](#)
- [SLIDES-A.pdf](#) [04-09-2013] [✕](#)

[+ Adicionar](#)

Figura 39 Listagem de recursos com a opção de exclusão e cadastro

Algoritmos 2012/2 - A0
Edsger Dijkstra

[✕ Excluir](#)

[Recursos](#) **Fóruns** [Avisos](#) [Participantes](#) [Tarefas](#)

[07-09-2013]	Material da aula passada
[04-09-2013]	Dúvida na primeira questão da lista 1
[04-09-2013]	Complexidade $O(n \log n)$ para o exercício da aula passada

[+ Criar fórum](#)

Figura 40 Listagem de fóruns



Figura 41 Listagem de avisos com a opção de edição, exclusão e cadastro.

6.5 Módulo de programação colaborativa

Conforme explicado no capítulo anterior, o JOnline permite que um ou mais usuários possam desenvolver programas de maneira colaborativa. Para que isso ocorra, o usuário deve, primeiramente, acessar a opção “Programação Colaborativa” disponível na barra de navegação superior do sistema. Feito isso, é mostrada a opção de criar um novo projeto, gerenciar os projetos já criados e ver uma lista de projetos que estão compartilhados com o usuário. Na figura 42 é mostrada a página mostrada ao usuário Blaise Pascal quando é feito o acesso ao módulo de programação colaborativa.



Figura 42 Visualização dos projetos do usuário Blaise Pascal

Ao criar um projeto, o usuário deve atribuir-lhe um nome e especificar a linguagem de programação utilizada nos códigos-fonte do projeto. Um projeto pode possuir um ou mais códigos-fonte e, assim, o usuário pode utilizar o botão de “Download” para obter um arquivo compactado contendo todos esses códigos-fonte.

Ao visualizar um projeto em específico, como mostrado na figura 43, o usuário pode optar por compartilhá-lo com outros usuários ou acessar os códigos-fonte que fazem parte do projeto. O compartilhamento do projeto ocorre através da especificação do e-mail do outro

usuário que terá acesso ao projeto. Esse usuário terá a permissão de adicionar códigos-fonte ao projeto e ainda alterá-los. Na figura 44 é possível ver a caixa de diálogo utilizada para compartilhar um projeto.



Figura 43 Visualização de um projeto do usuário Blaise Pascal



Figura 44 Caixa de diálogo para compartilhamento de um projeto

Ao abrir um dos códigos-fonte do projeto, o usuário tem a possibilidade de modificá-lo. As modificações feitas podem ser vistas por outros usuários colaboradores do projeto. Desse modo, para permitir a interação entre colaboradores do projeto, foi colocado um *chat* ao lado do código-fonte. A figura 45 contém um exemplo de código-fonte sendo alterado por dois usuários simultaneamente.

Efetuada as alterações necessárias, o usuário pode optar por salvar as alterações ou ainda executar o projeto acionando o botão “Executar” mostrado logo acima da caixa de texto. Ao pressionar o botão “Executar”, uma caixa de diálogo é aberta para que o usuário insira as entradas do programa. Na figura 46 é possível visualizar as entradas utilizadas na execução do projeto mostrado na figura 43.

Após a inserção das entradas do programa, uma nova janela (*pop-up*) é aberta com o resultado da execução. Um exemplo de resultado de execução pode ser visto na figura 47. Esse resultado pode também conter erros de compilação ou em tempo de execução. Na figura 48 pode ser visto um exemplo de saída contendo os erros de compilação de um projeto.

The screenshot shows the JOOnline web interface. At the top is a blue navigation bar with links: JOOnline, Problemas, Submissão, Programação Colaborativa, and Disciplinas. A language dropdown is set to 'Idioma'. On the left, a sidebar for user 'JOANNA' includes links for 'Minha Conta', 'Minhas turmas', and 'Sair', along with an 'AVISOS' section showing a notice from 04/09/2013. The main area features a code editor with C code for a student management system, including functions for initializing a class, inserting students, and printing the list. To the right of the code editor is a chat window titled 'Chat (2)' showing a conversation between Joanna Santos and Blaise Pascal. The chat messages include greetings and discussions about code errors and creating functions. At the bottom right of the chat is an 'Enviar' button.

Figura 45 Edição de um mesmo código-fonte por dois usuários

The screenshot shows a dialog box titled 'Executar' with a close button in the top right corner. Inside the dialog, there is a section labeled 'Entradas:' followed by a text area containing the input data: '2', 'Joanna 34', and 'João 65'. The name 'João' is underlined. At the bottom left of the dialog is a checkbox that is currently checked.

Figura 46 Caixa de diálogo usada para inserir dados de entrada do programa

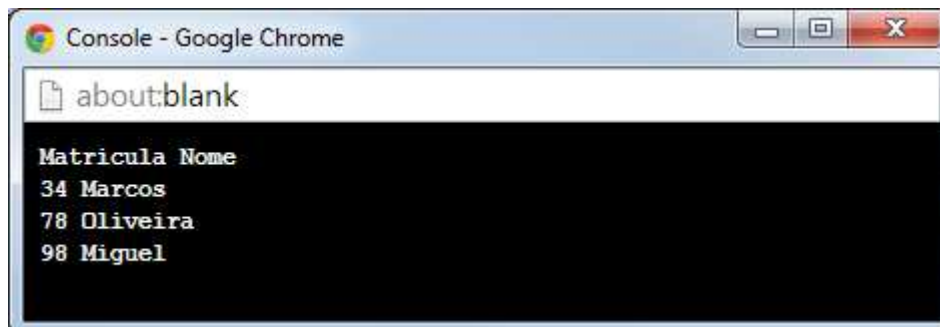


Figura 47 Exemplo de resultado da execução de um projeto

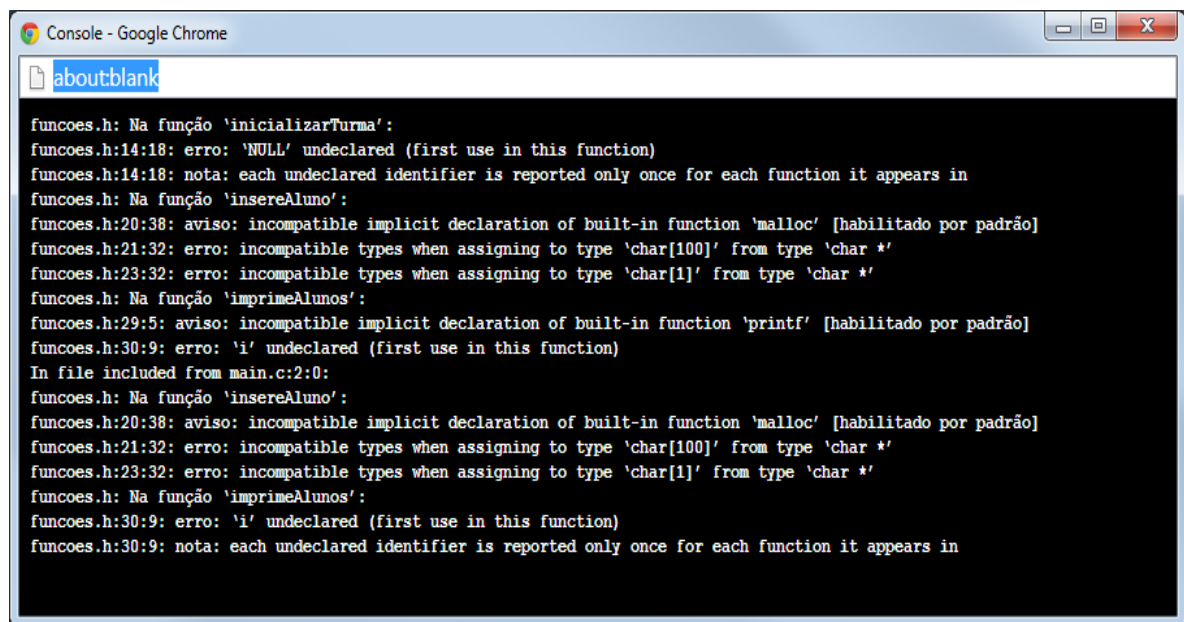


Figura 48 Erros de compilação mostrados na execução de um projeto

7 CONCLUSÃO

Com este trabalho de conclusão de curso foi possível obter um ambiente *online* que pode ser usado no suporte ao aprendizado de programação. Esse ambiente, chamado JOnline, auxilia tanto o professor no decorrer de suas disciplinas quanto o aluno em seu processo de aprendizagem de programação de computadores.

Consoante com o apresentado no capítulo 6 deste trabalho, o JOnline é composto de cinco módulos cada qual responsável por um conjunto de funcionalidades relacionadas. O módulo de disciplinas fornece um espaço para disponibilização de conteúdo sobre disciplinas e suas respectivas turmas. O módulo de problemas é usado pelos usuários para visualizar os problemas disponíveis no ambiente ou ainda gerenciá-los (caso o usuário tenha a permissão para isso). O módulo de programação colaborativa tem o principal objetivo de dar suporte ao desenvolvimento de programas em grupo. O módulo de submissão é responsável por avaliar os códigos-fonte enviados ao sistema e apresentar ao usuário o resultado dessa avaliação. O módulo de usuários é usado para gerenciamento da conta cadastrada no sistema.

Em seu estado atual, o JOnline suporta a avaliação de programas escritos em cinco linguagens de programação: C, C++, Pascal, Java e Python. Essas linguagens foram escolhidas por serem aquelas que mais são utilizadas atualmente nos cursos de computação da UFS. No entanto, como continuidade deste trabalho, pode-se incluir a avaliação de programas desenvolvidos em outras linguagens de programação, a exemplo de Lua, Tcl, entre outras.

Como trabalhos futuros é possível ainda adicionar ao sistema um módulo de processamento de erros de compilação, de tal forma a apresentar aos usuários dicas de como resolvê-los. Outra possibilidade é adicionar a detecção de plágio de modo que um professor possa verificar o grau de similaridade entre soluções para um mesmo problema.

No capítulo 4 mostrou-se que o JOnline disponibiliza a geração automática de resultados esperados dada a implementação da solução do problema e das suas entradas de teste. Sendo assim, é viável ainda adicionar a geração automática de entradas de teste. Para isso, seria apresentada uma interface em que o usuário especificaria o tipo de dados da entrada e seus respectivos limites e, em sequência, dados aleatórios seriam gerados baseados nessa especificação.

É possível ainda adicionar ao JOnline a opção de conversa com vídeo com codificação adaptativa entre dois usuários ao módulo de programação colaborativa. Agregar essa funcionalidade ao JOnline é desejável em virtude da crescente demanda por aplicações que permitam a interação entre usuários usando vídeo. Sendo assim, os usuários enviariam ao servidor os dados brutos de áudio e vídeo que seriam processados pelo servidor e codificados de acordo com o atraso e largura de banda do cliente que irá receber o vídeo. Uma maneira de incluir essa funcionalidade ao JOnline é descrita em detalhes no apêndice deste trabalho.

Durante o desenvolvimento do JOnline, diversas dificuldades foram encontradas. A primeira delas foi estabelecer um modo de executar os programas dos usuários em ambiente de execução seguro. Esta dificuldade foi superada através do uso do programa *safeexec* (discutido no capítulo 5) que se encarrega de fazer o monitoramento da execução dos programas.

Outra dificuldade encontrada, foi implementar WebSockets em Java. Por ser uma tecnologia recente, há poucos guias de implementação desta tecnologia em Java disponível na Internet. Além disso, os materiais encontrados sobre este assunto, por vezes, utilizavam uma versão obsoleta da API Grizzly Websockets.

Atualmente já é possível usar o JOnline como recurso adicional à disciplinas que envolvem práticas em programação de computadores, tais como, Algoritmos, Programação Imperativa/Orientada a Objetos, Estruturas de Dados e Introdução à Ciência da Computação. Desse modo, decorrido um espaço de tempo de uso do sistema é possível realizar uma análise do desempenho acadêmico dos discentes antes e depois desse uso. Consequentemente, é possível estabelecer um quantitativo dos reais impactos do uso de ambientes virtuais em processos de aprendizagem de programação de computadores.

Em síntese, espera-se que este trabalho contribua de forma significativa na formação acadêmica de profissionais na área de computação e estimule os alunos a participarem de maratonas de programação. Espera-se também criar uma cultura de aprendizado colaborativo através do uso da programação colaborativa disponibilizada pelo JOnline.

APÊNDICE

Usando o módulo de programação colaborativa do JOnline os usuários podem interagir usando um *chat* implementado usando WebSockets, uma tecnologia disponível em navegadores Web que suportam o HTML5. Em alguns navegadores Web mais recentes, o HTML5 também permite que os dados da Webcam (vídeo e áudio) possam ser capturados e enviados para um servidor. Dessa maneira, seria possível capturar dados brutos de áudio e vídeo e enviá-los para o servidor utilizando WebSockets. O servidor, ao receber esses dados, criaria vídeos a uma certa configuração de codificação a partir das imagens e do áudio recebidos. Após a codificação o servidor encaminharia o vídeo para o cliente interessado em recebê-lo. Esse funcionamento é mostrado na figura 49.

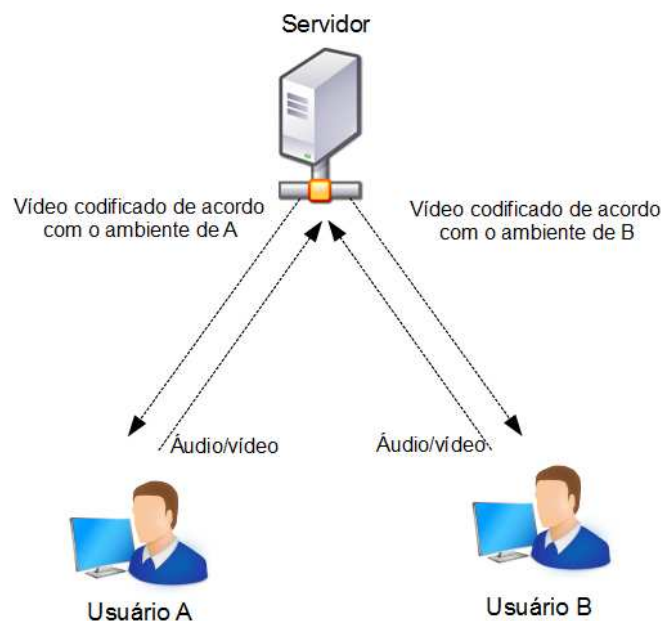


Figura 49 Possível implementação de *streaming* de vídeo adaptativo

A captura de áudio e vídeo pode ser feita usando a API *getUserMedia* (W3C, 2013a). Essa API permite que os dados de mídia do usuário sejam obtidos se o mesmo der permissão para isto. Sendo assim, uma vez dada a permissão para manipular o vídeo e áudio do usuário, é possível enviar esses dados periodicamente ao servidor a uma taxa constante. No quadro 10 é possível ver um *script* com os principais trechos para captura e envio das mídias do usuário.

```

var websocket, videoStream, audioStream, intervalID;
var FPS = 20; //frames por segundo
var VIDEO_LENGTH = 5; //segundos de duração de gravação de áudio
var framesNumber = 0; //número de frame enviado
//cria uma conexão Websocket
function joinChat(){
    websocket = new WebSocket('ws://localhost:8086/video_chat');
    websocket.onopen = function(e){
        //envia os dados a uma taxa
        intervalID = window.setInterval(sendDataToServer, 1000.0/FPS);
        //inicia a gravação do áudio
        startAudioRecording();
    }
}
//obtem uma imagem do vídeo em formato JPEG
var sendDataToServer = function(){
    var dataURL = getImgAsURL();
    //converte a URI para uma variável do tipo Blob
    blobImg = dataURIToBlob(dataURL);
    //envia ao servidor os dados
    websocket.send(blobImg);
    framesNumber = (framesNumber + 1) % (FPS*VIDEO_LENGTH);
    if(framesNumber == 0){
        stopAudioRecording(); //para a gravação do áudio
        saveAudio(); //envia o áudio
        startAudioRecording(); //reinicia gravação
    }
};
//pede permissão ao usuário para capturar os dados
function askPermission(){
    //verifica se o navegador tem suporte ao getUserMedia
    window.URL || (window.URL = window.webkitURL || window.msURL || window.oURL);
    navigator.getUserMedia || (navigator.getUserMedia =
    navigator.webkitGetUserMedia || navigator.mozGetUserMedia ||
    navigator.msGetUserMedia);
    var onFailSoHard = function(e) {
        alert('Você rejeitou o pedido de webcam ');
    };
    if(navigator.getUserMedia){
        //obtem somente vídeo
        navigator.getUserMedia({video: true}, function(stream) {
            videoStream = stream;
            var webcamSrc;
            if(window.URL && window.URL.createObjectURL)
                webcamSrc = window.URL.createObjectURL(stream);
            else
                webcamSrc = stream;
            $("#webcam").attr("src", webcamSrc);
            //invoca a função que inicializa as variáveis para manipular o áudio
            initAudioUserMedia();
            //obtem somente áudio
            navigator.getUserMedia({audio: true}, function(stream){
                audioStream = stream;
                gotStream(stream);
                joinChat();
            }, onFailSoHard);
        }, onFailSoHard);
    } else{ alert('Seu navegador não tem suporte ao HTML5'); }
}

```

Quadro 10 Script para capturar áudio e vídeo do usuário

Nesse código-fonte a função *askPermission* solicita ao usuário permissão de captura de áudio e vídeo de sua Webcam e, caso o usuário aceite essa solicitação, os dados são enviados periodicamente ao servidor através da função *sendDataToServer*. A taxa de envio desses dados é definida na variável *FPS* e, nesse exemplo, corresponde a uma taxa de 20 quadros por segundo. Após um dado intervalo de tempo a gravação de áudio também é enviada ao servidor. Esse intervalo de tempo é definido na variável *VIDEO_LENGTH* e, nesse caso, o intervalo de tempo é de cinco segundos.

O servidor ao receber esses dados brutos pode processá-los e criar vídeos de curta duração cada qual codificado de acordo com o estado atual da rede. Possíveis tecnologias que podem ser usadas na realização dessa atividade é a biblioteca FFmpeg (FFMPEG, 2013) ou a API JCodec (GARDINER, 2013).

Para que a adaptação seja alcançada, é necessário ainda monitorar o ambiente de execução do cliente. Dessa maneira, a codificação do vídeo é baseada no resultado desse monitoramento. No contexto de aplicações multimídia, é possível monitorar a latência e a largura de banda do usuário. Esse monitoramento pode ser feito usando a biblioteca Boomerang (YAHOO, 2013) que consiste de um *script* que monitora esses dois parâmetros ao longo do tempo.

Por fim, para reproduzir o vídeo enviado pelo servidor, é possível usar o elemento *video* disponível no HTML5. Esse elemento, atualmente, pode ser usado para reproduzir vídeos em formato MP4 (MP4REG, 2013), WebM (WEBM PROJECT, 2013) ou Ogg (XIPH, 2013).

É importante ressaltar que o uso do HTML5 para prover fluxo de vídeo adaptativo tem a vantagem deste não utilizar tecnologias proprietárias, sendo um padrão aberto. Entretanto, o HTML5 ainda não está completamente consolidado e não é suportado em todos os navegadores Web.

REFERÊNCIAS

1T3XT BVBA. **iText ® - Free / Open Source PDF Library for Java and C#**. Disponível em: <<http://itextpdf.com/>>. Acesso em: 15 jun. 2013.

ARRUDA, T. **Bootstrap Date/Time Picker**. Disponível em: <<http://tarruda.github.io/bootstrap-datetimepicker/>>. Acesso em: 7 ago. 2013.

APACHE SOFTWARE FOUNDATION. **Apache Commons**. Disponível em: <<http://commons.apache.org/>>. Acesso em 16 jul. 2013a.

_____. **Apache Taglibs**. Disponível em: <<http://tomcat.apache.org/taglibs/>>. Acesso em 16 jul. 2013b.

BARBOSA, R. M (Org.). **Ambientes Virtuais de Aprendizagem**. Porto Alegre: Editora Artmed, 2005.

BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. **UML - Guia do Usuário**. Tradução Fábio Freitas. Rio de Janeiro: Campus, 2000.

BOOTSTRAP. **Bootstrap**. Disponível em: <<http://getbootstrap.com/2.3.2/>>. Acesso em: 14 set. 2013.

CAMPOS, C. P.; FERREIRA, C. E. **BOCA: um sistema de apoio para competições de Programação**. Workshop de Educação em Computação, Anais do Congresso da SBC, Salvador-BA, 2004.

CAPTERRA. **The Top 20 Most Popular LMS Software Solutions**. Disponível em: <<http://www.capterra.com/top-20-lms-software-solutions>>. Acesso em: 20 abr. 2013.

CHEANG, B. et al. **On automated grading of programming assignments in an academic institution**. *Computers & Education*, Oxford: Elsevier, v. 41, n. 2, p. 121-131, set. 2003.

CHEN, R. Q. et al. **An effective E-Learning Platform for Computer Programming**. *Educational Technology Letters*. Delaware: Information Engineering Research Institute, v. 2, n. 2. p.57-63, Dezembro de 2012.

CHEN, Y. et al. **DRWSC – To Simplify Dynamic Invocation for RESTful Web Services**. In: 2010 IEEE International Conference on Software Engineering and Service Sciences (ICSESS), pp.226,229, 16-18 jul. 2010.

CHILDERS, J. **SimpleCaptcha - A CAPTCHA framework for Java**. Disponível em: <<http://simplecaptcha.sourceforge.net>>. Acesso em: 7 maio 2013.

CODEMIRROR. **CodeMirror**. Disponível em: <<http://codemirror.net>>. Acesso em: 14 set. 2013.

CODEPAD. **CodePad**. Disponível em:

<<https://github.com/sathvikp/CodePad/tree/master/applications/init/static/js>>. Acesso em: 14 set. 2013.

COURSERA. **About Us | Coursera**. Disponível em: <<https://www.coursera.org/>> Acesso em: 5 out. 2013.

DEITEL, H. M.; DEITEL, P. J. **Java TM: como programar**. 8. ed. São Paulo: Pearson, 2010.

EACEA. **Integrating Online Judge into effective e-learning**. Disponível em:

<http://eacea.ec.europa.eu/llp/projects/public_parts/documents/ict/2010/ict_mp_135221_edujudge.pdf>. Acesso em: 31 mar. 2013.

FERRANDIN, M.; STEPHANI, S. L. **Ferramenta para o ensino de programação via Internet**. I Congresso Sul Catarinense de Computação: UNESC – Criciúma, 2005.

FFMPEG. **FFmpeg**. Disponível em: <<http://www.ffmpeg.org>>. Acesso em: 10 set. 2013.

FUCHEN, Y.; PENGCHENG, X.; DI, X. **Welcome to PKU JudgeOnline**. Disponível em <<http://www.poj.org>>. Acesso em: 20 mar. 2013.

GARDINER, G. **JCodec**. Disponível em: <<http://jcodec.org/>>. Acesso em: 10 set. 2013.

GEARY, D. **JSP templates**. Disponível: <<http://www.javaworld.com/jw-09-2000/jw-0915-jspweb.html>>. Acesso em: 6 maio 2013.

GOMES, A. S et al. **Amadeus: novo modelo de sistema de gestão de aprendizagem**.

Revista Brasileira de Aprendizagem Aberta e a Distância, Associação Brasileira de Educação a Distância, v. 8, p 10-26, 2009.

HADLEY, M. **Web Application Description Language**. Disponível em:

<<http://www.w3.org/Submission/wadl/>>. Acesso em: 21 abr. 2013.

HAMAD, H.; SAAD, M.; ABED, R. **Performance Evaluation of RESTful Web Services for Mobile Devices**. *Internatinal Arab Journal of e-Techonology*, Dübendorf: Arab Open Publisher, v. 1, n. 3, p. 72-78, jan. 2010.

HANSEN, M. D. **SOA Using Java Web Services**. Upper Saddle River: Prentice Hall, 2007.

HICKSON, I. (2012). **The WebSocket API**. Disponível em :

<<http://www.w3.org/TR/websockets/>>. Acesso em: 10 jun. 2013.

HICKSON, I. et al . **HTML5**. Disponível em: <www.w3.org/TR/html5/>. Acesso em: 16 set. 2013

JQUERY FOUNDATION. **jQuery**. Disponível em: <<http://jquery.com/>>. Acesso em: 14 set. 2013.

KOSHY, T. **Fibonacci and Lucas Numbers with Applications**. Nova Iorque: Wiley-Interscience, 2001.

KURNIA, A.; LIM, A.; CHEANG, B. **Online Judge**. Computer & Education, vol. 36, No. 4, maio 2001, pp 299-315.

LI, S. et al. **The application of course-oriented online judge in “Data Structure”**. 2012 *IEEE Symposium on Robotics and Applications (ISRA)*, pp.687,690, 3 a 5 de Junho de 2012.

MOBWRITE. **Google-mobwrite - Real-time Synchronization and Collaboration Service - Google Project Hosting**. Disponível em: <<https://code.google.com/p/google-mobwrite/>>. Acesso em: 14 set. 2013.

MOREIRA, M.P. ; FAVERO, E. L. **Um Ambiente para Ensino de Programação com Feedback Automático de Exercícios**. In: Workshop sobre Educação em Computação (Congresso da Sociedade Brasileira de Computação), 2009, Bento Gonçalves. WEI - XVII Workshop sobre Educação em Computação, 2009. v. 1. p. 154-161.

MP4REG. **The 'MP4' Registration Authority**. Disponível em: <<http://www.mp4ra.org/specs.html>>. Acesso em: 14 set. 2013.

NOBRE, I. A. M.; MENEZES, C. S. **Suporte à Cooperação em um Ambiente de Aprendizagem para Programação (Samba)**. In: XIII Simpósio Brasileiro de Informática na Educação - SBIE, 2002, São Leopoldo - RS. XIII SBIE, 2002.

OASIS (2013). **UDDI | Online community for the Universal Description, Discovery, and Integration**. Disponível em: <<http://uddi.xml.org/>>. Acesso em: 12 dez. 2012.

ORACLE. **JavaServer Pages Standard Tag Library**. Disponível em: <<http://www.oracle.com/technetwork/java/index-jsp-135995.html>>. Acesso em: 5 maio 2013a.

_____. **Jersey**. Disponível em: <<http://jersey.java.net/>>. Acesso em: 2 abr. 2013b.

_____. **MySQL :: MySQL Connectors**. Disponível em: <<http://www.mysql.com/products/connector/>>. Acesso em: 5 maio 2013c.

PARK, T. **Bootswatch: Cerulean**. Disponível em: <<http://bootswatch.com/cerulean/>>. Acesso em: 24 maio 2013.

PAYNE, N. **Bootbox.js—alert, confirm and flexible dialogs for Twitter's Bootstrap framework**. <Disponível em: <http://bootboxjs.com/>>. Acesso em: 24 maio 2013.

PEREIRA JÚNIOR, J. C. R.; RAPKIEWICZ, C. E. **O processo de ensino-aprendizagem de fundamentos de programação: uma visão crítica da pesquisa no Brasil**. I Workshop sobre Educação em Computação Rio de Janeiro/Espírito Santo - 2004 - Vitória, ES - Rio das Ostras, RJ, Brasil.

PIMENTEL, E. P. et al. **Avaliação Contínua da Aprendizagem, das Competências e Habilidades em Programação de Computadores**. IX Workshop de Informática na Escola - WIE - 2003 - Campinas, SP, Brasil.

PKWARE (2013). **ZIP File Format Specification**. Disponível em: <<http://www.pkware.com/documents/casestudies/APPNOTE.TXT>>. Acesso em: 14 set. 2013.

PROJECT GRIZZLY (2013). **Project Grizzly – WebSocket Overview**. Disponível em: <<https://grizzly.java.net/websockets.html>>. Acesso em: 13 jun. 2013.

QUEIROZ, F. C. B. P. et al. **Contribuição dos Sistemas Integrados de Gestão para as Práticas de Ensino e Aprendizagem**. *Augusto Guzzo Revista Acadêmica*, São Paulo, n. 9, p.45-52, jul. 2002. Disponível em: <http://www.fics.edu.br/index.php/augusto_guzzo/article/view/24>. Acesso em: 6 out. 2013.

REVILLA, M. A. ; MANZOOR, S. ; LIU, R. **Competitive Learning in Informatics: The UVa Online Judge Experience**. *Olympiads in Informatics*, Vilnius: Vilnius University Institute of Mathematics and Informatics, v. 2, p. 131–148, 2008.

SAMPAIO, C. **SOA e Web Services em Java**. Rio de Janeiro: Brasport, 2006.

SINFO - A Superintendência de Informática da Universidade Federal do Rio Grande do Norte. **Projetos**. Disponível em: <<http://www.info.ufrn.br>>. Acesso em 6 out. 2013.

SOCIEDADE BRASILEIRA DE COMPUTAÇÃO - SBC. **MARATONA DE PROGRAMAÇÃO**. Disponível em: <http://www.sbc.org.br/index.php?option=com_content&view=category&layout=blog&id=303&Itemid=180>. Acesso em: 5 out. 2013.

SOMMERVILLE, I. **Engenharia de software**. 8. ed. São Paulo: Pearson, 2007.

SPHERE RESEARCH LABS. **Sphere Online Judge**. Disponível em: <<http://www.spoj.pl>>. Acesso em: 20 mar. 2013.

STAL, M. **Web services: beyond component-based computing**. *Communications of the ACM*, Nova Iorque: ACM, v. 45, n. 10, p.71-76, 2002.

TOPCODER. **TopCoder, Inc. | Home of the world's largest development community**. Disponível em : <<http://www.topcoder.com/>>. Acesso em: 20 mar. 2013.

UNIVERSIDAD DE VALLADOLID. **UVA Online Judge**. Disponível em : <<http://uva.onlinejudge.org/>>. Acesso em: 20 mar. 2013.

URI ONLINE JUDGE. **URI Online Judge**. Disponível em : <<http://www.urionlinejudge.com.br/>>. Acesso em: 20 mar. 2013.

XIPH. **Ogg Vorbis Documentation**. Disponível em: <<http://xiph.org/vorbis/doc/>>. Acesso em: 14 set. 2013.

YAHOO. **This, is boomerang**. Disponível em: <<http://yahoo.github.io/boomerang/doc/>>. Acesso em: 14 set. 2013.

YI, C. et al. **The Design of Course-Oriented Online Judge**. International Conference on Electrical Engineering and Automatic Control (ICEEAC). 2010

W3C. **Media Capture and Streams**. Disponível em: <<http://www.w3.org/TR/mediacapture-streams/>>. Acesso em: 14 set. 2012a.

_____. **SOAP Specifications**. Disponível em: <<http://www.w3.org/TR/soap/>>. Acesso em 12 dez. 2012b.

_____. **Web Services Description Language (WSDL) 1.1**. Disponível em: <http://www.w3.org/TR/wsdl>. Acesso em: 12 de dezembro de 2012c.

WEBM PROJECT. **The WebM Project | WebM Container Guidelines**. Disponível em: <<http://www.webmproject.org/docs/container/>>. Acesso em: 14 set. 2013.