AMBIENTE VIRTUAL DE APRENDIZAGEM DE ALGORITMOS (AVAA): A CONSTRUÇÃO DE UMA FERRAMENTA NO PROCESSO ENSINO-APRENDIZAGEM

Jefté Goes Salvador Silva[1]

Cláudia Pinto[2]

RESUMO

A temática do presente estudo surgiu da observação e relatos dos estudantes, durante o programa de monitoria de algoritmos e linguagens de programação do curso de Sistemas de Informação, no qual se constatou a dificuldade dos alunos, com relação ao processo educacional. Considerando-se o alto índice de evasão e reprovação dos estudantes nessa disciplina, devido às dificuldades encontradas pelo estudante no momento de usar o raciocínio lógico, surgiu o interesse em criar uma ferramenta dentro do ambiente virtual de aprendizagem voltado para construção de algoritmos, com o objetivo de contribuir com o aprendizado do estudante na construção de algoritmos, a partir de exercícios e jogos educacionais propostos pelo professor, através de uma interface web interativa e intuitiva, e auxiliar o professor na avaliação desses estudantes, minimizando, desta forma, alguns dos impactos gerados no processo de ensino-aprendizagem.

1 INTRODUÇÃO

Com a crescente evolução dos sistemas computacionais se tornou necessária à presença de profissionais cada vez mais capacitados, que sejam capazes de inovar buscando as melhores soluções para os diversos tipos de problema. Para que isso seja possível é necessário que sua base seja consolidada no período acadêmico, principalmente nas áreas de Computação e Informática, que envolvem tecnologias em constante desenvolvimento. Nos cursos da área, em especial os de Sistemas de Informação e Ciência da Computação, observa-se uma considerável evasão e/ou reprovação na disciplina de Algoritmos e Programação, seja pela pouca experiência dos estudantes, que geralmente estão ingressando no curso, pela deficiência em raciocínio lógico ou por outros motivos relacionados.

> As disciplinas que envolvem o ensino de Algoritmos e Programação são normalmente lecionadas nas primeiras fases dos cursos das áreas tecnológicas. Tais disciplinas são consideradas desafiadoras pelos alunos, pois exigem o desenvolvimento de estratégias de solução de problemas com base lógico-matemática. A consequência disso é o elevado número de problemas de aprendizagem, favorecendo a ocorrência de reprovações (DETERS, 2008, p. 1).

[1] Graduando em Sistemas de Informação (UNIFACS). jefte.goes@hotmail.com

[2] Professora e Coordenadora do Curso de Sistemas de Informação (UNIFACS). caupinto.sena@gmail.com

A primeira disciplina de programação em um curso de sistemas de informação é geralmente ministrada nos períodos iniciais do curso, sendo bastante comum ser oferecida já nos primeiros semestre.

De acordo com FALKEMBACH (2003, p. 2):

> É muito importante que a primeira disciplina que trabalha com os conceitos básicos de resolução de problemas via computador forneça ao aluno subsídios que sustentem abstrações e o formalismo exigido na área computacional. É necessário estabelecer, desde o início do curso, estratégias para organizar as ideias, o raciocínio e a representação simbólica, a afim de que o aluno incorpore esses hábitos e os utilize sempre na resolução de problemas durante o próprio curso e, posteriormente, como profissional.

As dificuldades encontradas pelo professor e pelo estudante para estabelecer a comunicação são muitas, talvez pela falta de um método de ensino adequado para aquele estudante ou um material didático especial.

Aliada a esta realidade, observou-se durante o período de um semestre, enquanto monitor voluntário da disciplina de Algoritmos e Linguagens de Programação, que os estudantes tinham noção das estruturas lógicas, de repetição e outras, pertinentes aos algoritmos ou alguma linguagem de programação, mais tinha problema em montar a lógica, escolher a estrutura mais adequada para determinado problema, dividir o problema em partes e entre outros, por mais claro que o exercício fosse.

Para tentar minimizar essa dificuldade, é proposto o desenvolvimento de jogos educacionais dentro do próprio Ambiente Virtual, proporcionando ao estudante desenvolver seus processos cognitivos (atenção, concentração, criatividade), tornando o ambiente virtual um ambiente de interação que vai além das salas de aula. A partir daí o professor estará avaliando tanto a turma da disciplina corrente quanto o estudante individualmente, identificando os estudantes que precisam de mais atenção, quais exercícios tem mais aproveitamento e estabelecendo novas estratégias. Dessa maneira, é possível também acompanhar e avaliar o estudante no estágio inicial do curso relacionado ao estudo de algoritmos e programação, minimizando problemas relatados anteriormente, como evasão, desinteresse, reprovação, dificuldade de aprendizagem. Este artigo se organiza em cinco seções, incluindo a introdução e as referências. A Seção 2 traz aspectos teóricos relacionados a Ambientes Virtuais de Aprendizagem (AVAs); a Seção 3 aborda o Sistema AVAA, desde sua concepção, levantamento de requisitos até sua prototipação e a Seção 4 traz as conclusões.

2 AMBIENTES VIRTUAIS DE APRENDIZAGEM

Atualmente, o avanço e o desenvolvimento tecnológico vêm impulsionando e transformando a maneira de ensinar e de aprender. Nos últimos anos, os Ambientes Virtuais de Aprendizagem estão sendo cada vez mais utilizados no âmbito acadêmico para atender as necessidades estudantes e professores.

Segundo Theresinha e outros (2007, p. 5), “Ambiente Virtual de Aprendizagem (AVA) consiste em uma opção de mídia que está sendo utilizada para mediar o processo ensino-aprendizagem à distância”. Os AVAs não substituem o processo pedagógico de ensino-aprendizagem, eles devem ser usados como um complemento do processo, de forma que seja possível acompanhar o estudante e dar suporte as mais variadas necessidades, tanto do professor quanto do estudante, seja dentro ou fora de aula ou até mesmo separados geograficamente.

Theresinha e outros (2007, p. 6) resumem que para a gestão do aprendizado e a disponibilização de matérias, um AVA deve apresentar algumas ferramentas como:

a) Controle de acesso: geralmente realizado através de senha;
b) Administração: refere-se ao acompanhamento dos passos do estudando dentro do ambiente, registrando seu progresso por meio das atividades e das páginas consultadas;
c) Controle de tempo: feito através de algum meio explícito de disponibilizar matérias e atividades em determinados momentos do curso, por exemplo, o recurso calendário;
d) Avaliação: usualmente formativa (a auto avaliação);
e) Comunicação: promovida de forma síncrona e assíncrona;
f) Espaço privativo disponibilizado para os participantes trocarem e armazenarem arquivos;
g) Gerenciamento de uma base de recursos: Como forma de administrar recursos menos formais que os materiais didáticos como FAQ (perguntas frequentes) e sistemas de busca.
h) Apoio: como por exemplo, a ajuda on-line sobre o ambiente;
i) Manutenção: Relativo à criação e atualização de matérias de aprendizagem.

Atualmente, há muitos sistemas de Ambiente Virtual de Aprendizagem. Foram tomados como bases referenciais e de comparação em relação ao ambiente proposto os ambientes Moodle (MOODLE, 2013) e TelEduc (TELEDUC, 2013).

MOODLE é o acrónimo de *Modular Object-Oriented Dynamic Learning Environment*, criado pelo educador e cientista computacional Martin Dougiamas. O TelEduc é um ambiente de educação a distância pelo qual se pode realizar cursos através da Internet, desenvolvido conjuntamente pelo Núcleo de Informática Aplicada à Educação (NIED) e pelo Instituto de Computação (IC) da Universidade Estadual de Campinas (UNICAMP).

Ambos os ambientes são disponibilizados livremente na forma de software livre e podem ser instalados em diversos sistemas operacionais, desde que os mesmos consigam executar a linguagem PHP. Com o Moodle, é possível utilizar os seguintes bancos de dados: MySql, PostgreSQL, Oracle, Access, Interbase ou qualquer outra acessível via ODBC, já o TelEduc só MySql. Tanto o Moodle quanto o TelEduc são AVAs genéricos, que podem se adaptar a qualquer tipo de instituição, se moldando a necessidade dos estudantes e professores. O Quadro 1 mostra a funcionalidade de ambos os ambientes.

Quadro 1 – Funções dos ambientes TelEduc e Moodle

Funções Comuns	TelEduc	Moodle
Agenda	X	X
Avaliações	X	X
Atividades	X	X
Material de Apoio	X	X
Enquetes	X	X
Mural	X	X
Correio	X	X
Grupos	X	X
Fórum	X	X
Chat	X	X
Perfil	X	X
Diário de Bordo	X	
Portfólio	X	
Relatórios	X	X
Notas	X	X
Livro		X
Glossário		X
Blog		X

Comunicador instantâneo		
Usuários	X	X
Wiki		X
Ajuda	X	X

Fonte: TelEduc, 2013 e MOODLE, 2013

O Ambiente Virtual de Aprendizagem de Algoritmos (AVAA), apresentado na Seção seguinte, diferente dos ambientes Moodle e TelEduc, é destinado somente às disciplinas relacionadas ao estudo de algoritmos e programação e não genérico como os anteriores.

3 JOGOS INTERATIVOS NO PROCESSO DE ENSINO-APRENDIZAGEM

A partir do que foi apresentado na Introdução, à evasão e reprovação podem levar a falta de motivação e gerar outros fatores encadeados a ele, tais como: falta de atenção, não realização de exercícios de fixação, dentre outros. Dessa forma, é preciso proporcionar estímulos aos estudantes ao estudo da disciplina, usando ferramentas que proporcionem interatividade e retorno imediato, tornando o aprendizado dinâmico.

A inclusão de metodologias de ensino-aprendizado no formato de jogos proporciona ao estudante a capacidade de interagir com o mundo em sua volta. É preciso criar condições para a formação de sujeitos capazes de agir sobre o mundo e transformá-lo em uma sociedade na qual é crescente o uso de tecnologias da informação e comunicação (TIC).

Os jogos educacionais são ferramentas que apoiam o processo de ensino-aprendizagem. Segundo Tarouco (2013), os jogos podem ser ferramentas eficientes, pois eles divertem enquanto estimulam, facilitam o aprendizado e aumentam a capacidade de retenção do que é ensinado, exercitando as funções mentais e intelectuais do jogador. A autora destaca algumas características que fazem com que os jogos sejam atrativos como: possuem regras, metas, privilegiam interação homem-máquina, apresentam resultados e feedback e são, acima de tudo, divertidos.

Através dos jogos, os estudantes se sentem estimulados no desenvolvimento das tarefas e, consequentemente, desenvolvem o raciocínio. Isso porque o jogo agrega um conjunto de elementos visuais que prendem mais a atenção do que questões em papel. O fato de o aluno interagir é um benefício que o jogo oferece, uma vez que o estudante consegue visualizar sua ação sobre o jogo. Através da interação, o estudante consegue visualizar os passos necessários para alcançar um resultado quando, por exemplo, visualiza, seleciona e transporta os elementos no jogo. O ensino de algoritmos por meio dos jogos digitais pode diminuir as dificuldades encontradas pelos estudantes e facilitar o processo de ensino-aprendizagem.

Com o intuito de utilizar os jogos para auxiliar o desenvolvimento do raciocínio lógico dos estudantes, tanto para leitura de algoritmos quanto para posterior elaboração de algoritmos simples, foi desenvolvido a metodologia quebra-cabeça, cuja concepção gira em torno dos problemas de ensino/aprendizagem de Algoritmos e Programação, mas sem deixar de lado os aspectos lúdicos do ato de jogar, para garantir a motivação dos estudantes.

O professor pode propor a construção de diversos jogos que podem funcionar dentro do AVAA através de *plug-in* anexados e implementados no projeto principal do AVAA utilizando a linguagem de programação Adobe Flex.

Inicialmente é proposto um jogo de montagem de peças chamado de quebra cabeça, em que, dependendo do nível do exercício informado pelo professor, o algoritmo será “quebrado” e o aluno terá que arrumar o algoritmo de forma que quando for submetido o sistema irá verificar se está correto ou não apontando os possíveis erros (cadastrado previamente pelo professor).

O algoritmo cadastrado pelo professor será apresentado bagunçado e separado por partes, baseado na dificuldade que o professor escolheu para aquele algoritmo. Quando o estudante seleciona o bloco, linha ou palavra de código, o enunciado se destaca por uma cor previamente definida, para mostrar que aquele código é referente à entrada, processamento e saída, azul, vermelho e verde respectivamente. Através de um recurso computacional *drag-and-drop* (arraste e solte), o estudante poderá interagir com o algoritmo associando as cores, elementos e enunciado montando o algoritmo.

Para resolução de um algoritmo qualquer, são necessários os passos descritos a seguir:

1. Ler atentamente o enunciado;
2. Definir os dados de entrada, arrastando os respectivos elementos de entrada no início, associando os elementos pela cor AZUL do enunciado;
3. O processamento é responsável pela transformação dos dados de entrada em dados de saída. Arrastar os respectivos elementos de processamento no meio da área, associando os elementos de processamento pela cor VERMELHO do enunciado;

4. Definir os dados de saída, ou seja, quais dados serão gerados depois do processamento, arrastando os respectivos elementos de saída no final da área, associando os elementos de saída pela cor verde do enunciado;
5. Finalizar o algoritmo.

4 SISTEMA AVAA

Algumas instituições de ensino que oferecem cursos voltados à área da computação como Engenharia da Computação, Ciência da Computação e Sistemas de Informação e áreas afins, mostram um grande déficit de aprovação nas disciplinas que envolvem algum tipo de algoritmo, seja ela em qualquer linguagem de programação.

Antes do advento e amadurecimento da Internet, as pessoas estavam, de certa forma, limitadas geograficamente para interagir e aprender através de ambientes virtuais. Com o surgimento da Internet e a melhoria das diversas tecnologias que a compõem, os ambientes virtuais passaram a ser vistos como um importante meio de aprendizagem. O surgimento dos ambientes virtuais deu às pessoas a possibilidade de adquirir conhecimento e aprender sem a necessidade de saírem das suas casas.

Visando criar uma nova forma de Ambiente Virtual direcionado para os estudantes da área de informática, este documento descreve todas as necessidades e restrições para a criação de um Ambiente Virtual de Algoritmos no qual estudantes e professores poderão interagir, respondendo exercícios, avaliando resultados e aprender jogos aplicados pelos professores.

O objetivo desse artigo é apresentar um Ambiente Virtual de Aprendizagem de Algoritmos - (AVAA) que tem como objetivos:

a) Avaliar o estudante através de exercícios.
b) Permitir aos professores avaliar o desempenho dos estudantes através do tempo de resolução e dificuldade dos exercícios.
c) Aplicar as metodologias de aprendizagem no formato de jogos (quebra-cabeça).
d) Aplicar atividades avaliativas.
e) Submeter materiais para auxiliar na resolução de exercícios.

4.1 Metodologia de Pesquisa

Para a escolha dos métodos que irão atender os objetivos deste projeto, a metodologia adotada foi à pesquisa de campo exploratória.

> Utilizada com o objetivo de conseguir informações e/ou conhecimentos acerca de um problema, para o qual se procura uma resposta, ou de uma hipótese, que se queira comprovar, ou, ainda, de descobrir novos fenômenos ou as relações entre eles. (MARCONI; LAKATOS 2010, p. 169).

Em uma pesquisa exploratória, podem ser utilizados questionários, entrevistas, observação participante e análise de conteúdos, dentre outros. A escolha do método e instrumento utilizado depende do objetivo da pesquisa, dos recursos financeiros disponíveis, da equipe e elementos no campo da investigação (MARCONI; LAKATOS, 2010).

A partir de uma coleta de dados feita inicialmente de forma empírica, através de observação e relatos dos estudantes, durante o programa de monitoria de algoritmos e linguagens de programação com a duração de um semestre, foi constatada a necessidade de trabalhar com ferramentas de pesquisa mais consistentes.

Para os testes e análise da ferramenta e resultados, a pesquisa propõe que, depois do desenvolvimento da ferramenta (AVAA), sejam utilizados três instrumentos, que poderão ser aplicados separadamente ou em conjunto para atender os objetivos do estudo: entrevista, questionário e formulário, detalhados a seguir:

a) Os **formulários** consistem em um conjunto de questões que são perguntadas e anotadas por um entrevistador durante uma entrevista face a face (MARCONI; LAKATOS, 2010);

b) O método da **entrevista** se caracteriza pela existência de um entrevistador, que faz perguntas ao entrevistado anotando as suas respostas. O entrevistador pode tirar dúvidas, explicar as questões permite também identificar as falhas. Além disso, a entrevista permite obter muitas respostas gerando uma grande quantidade de dados (MARCONI; LAKATOS, 2010);

c) O **questionário** é um conjunto de perguntas, que a pessoa lê e responde sem a presença de um entrevistador. As vantagens do uso do método do questionário em relação às entrevistas são: utilizam-se menos pessoas para ser executado e proporciona economia de custo, tempo, viagens, com obtenção de uma amostra maior e não sofre influência do entrevistador (MARCONI; LAKATOS, 2010)**.**

Assim, registrando a evolução do estudante no que diz respeito à interação com o AVAA, avaliando às habilidades usadas para resolução de problemas e especificação de soluções, o projeto optou pelo uso do questionário na primeira fase, que identificará os estudantes que já tem experiência com o uso de algoritmos e programação e os que têm algum tipo de dificuldade e qual a sua expectativa com o uso do AVAA. Entre os principais fatores que explicam esta escolha está a pouca disponibilidade de recursos e de pessoas para realizar a pesquisa com amostra grande por entrevista (aproximadamente 100 estudantes). Neste projeto, serão utilizados formulários *online* pela facilidade e praticidade, visto que praticamente todos os estudantes de computação dispõem de um computador e a internet, e a própria instituição oferece laboratórios com acesso ao mesmo.

Na segunda fase da pesquisa, pretende-se aprofundar os dados coletados, realizando uma nova coleta de dados, apenas com os estudantes que interagiram com o AVAA a fim de identificar os problemas de usabilidade, erros, e benefícios alcançados com a sua implantação. Como nesta etapa será usada uma pequena amostra e seu objetivo é conseguir um aprofundamento dos dados desse tema, resolveu-se usar o método da entrevista.

4.2 Metodologia de Desenvolvimento

Como metodologia de desenvolvimento da ferramenta AVAA, propõe-se o modelo evolucionário, em função de ser considerado por Sommerville (2007, p.45) mais eficiente que o modelo em cascata, levando em consideração que no modelo evolucionário a especificação pode ser desenvolvida à medida que vão aparecendo às necessidades. Um ponto negativo para esse modelo é que no final do projeto podem surgir falhas na arquitetura e desenvolvimento da ferramenta. Nesse sentido, as seguintes etapas foram seguidas:

a) **Estudo dos Ambientes Virtuais existentes para as disciplinas que envolvem algoritmos** em livros, artigos, revistas, monografias, dissertações e páginas da *Web* para o desenvolvimento da fundamentação teórica e prática para o trabalho;
b) **Estudo das metodologias de ensino de algoritmos**, como são construídos e suas principais características;
c) **Análise dos requisitos necessários** para integrar o ensino de algoritmos a ferramenta virtual;
d) **Implementação de um protótipo** para avaliar a usabilidade da ferramenta.

e) **Criação de um conjunto de critérios para avaliar o AVAA** frente aos outros ambientes virtuais;
f) **Pesquisa de Campo, avaliar os resultados** obtidos com o uso da ferramenta no curso de Sistemas de Informação;
g) **Conclusão**. Questões pendentes de implementação frente ao objetivo deste trabalho.

4.2.1 Levantamento de Requisitos

A tarefa de desenvolvimento de software engloba uma série de fases e atividades que, independentemente da metodologia escolhida, ocorrem para a realização do seu objetivo: entregar um software funcionando corretamente dentro do orçamento e prazos previstos para o seu desenvolvimento. Diante da ideia de se desenvolver um ambiente virtual voltado para os cursos que envolvem matérias de algoritmos com âmbito educacional, todas as atividades de desenvolvimento têm que ser criteriosamente elaboradas e desenvolvidas, formalizando: atividades de análise de requisitos, design, definição de arquitetura, codificação e outras. Um trabalho consistente de análise dos requisitos, ou seja, identificar, quantificar, definir, priorizar e classificar os principais problemas que o futuro software deve resolver é à base de um projeto de software de sucesso.

A seguir são apresentados todos os requisitos funcionais, não funcionais e inversos do AVAA.

4.2.1.1 Requisitos Funcionais

O termo “gerenciar” nesse documento engloba as funções de: organizar, inserir, desativar, ativar, editar e ler uma informação de um determinado cadastro no sistema. O sistema AVAA deve:

1. Permitir ao administrador <u>parametrizar a instituição de ensino</u> no AVAA como colocar o nome da instituição e logotipo.
2. Permitir ao administrador <u>Gerenciar os Cursos</u> da instituição como: Ciência da Computação, Engenharia de Software, Sistemas de Informação, etc.
3. Permitir ao administrador <u>gerenciar as disciplinas dos cursos</u>.

3.1. Permitir <u>atribuir as disciplinas</u> aos seus respectivos cursos.

4. Permitir ao administrador <u>gerenciar o cadastro dos professores</u>.

4.1. Permitir alocar os professores as disciplinas que leciona.

5. Permitir ao administrador <u>gerenciar o cadastro dos estudantes</u>

5.1. Alocar os estudantes as disciplinas de acordo com o curso informado.

6. Permitir ao professor gerenciar os avisos para determinado disciplina, exemplo de aviso: “Prova no dia 18/10/2012” ou “Atividade Avaliativa”.

7. Permitir ao professor gerenciar as disciplinas que leciona.

8. Permitir ao professor gerenciar os arquivos de ajuda para determinada disciplina, os formatos disponíveis são arquivos em *.txt, *.rar, *.pdf e *.doc.

9. Permitir ao professor associar ou desassociar o estudante na disciplina.

9.1. Permitir associar ou desassociar todos os estudantes de vez.

10. Permitir ao professor gerenciar os exercícios das disciplinas.

10.1. Permitir ao professor duplicar o exercício selecionado para outra disciplina.

11. Permitir ao professor gerenciar os exercícios da metodologia (inicialmente chamada de “quebra-cabeça”).

11.1. Permitir ao professor replicar o exercício da metodologia selecionada para outra disciplina.

12. Permitir ao professor analisar o desempenho através do tempo de resposta do exercício + avaliação do professor + dificuldade do exercício que varia de um a cinco, e a partir dessa avaliação, o programa irá somar uma quantidade de pontos e incrementar com a quantidade de pontos existentes no estudante.

13. Permitir ao professor gerenciar as avaliações da disciplina.

13.1. Permitir ao professor duplicar a avaliação selecionada para outra disciplina.

13.2. Permitir ao professor comparar as provas dos estudantes da disciplina.

13.3. Permitir ao professor lançar os resultados das provas e fechar as provas da disciplina.

14. Permitir ao professor alterar seu próprio cadastro, a fim de atualizações.

15. Permitir ao estudante se cadastrar.

15.1. O estudante deve solicitar entrada na disciplina para o professor pelo ambiente virtual.

16. Permitir ao estudante responder os exercícios disponíveis na disciplina.

17. Permitir ao estudante responder os exercícios da metodologia “quebra-cabeça” disponíveis na disciplina.

18. Permitir ao estudante responder as avaliações submetidas pelo professor.

4.2.1.2 Requisitos Não Funcionais

1. O sistema AVAA deve ter um nível de desempenho apropriado.
   1.2. O tempo decorrido para visualização dos exercícios deve ser o menor possível, depois que o *plug-in* do *Flash Player* estiver carregado, estima-se que os exercícios estejam disponível entre 2 e 5 segundos.
   1.3. O tempo decorrido entre a resposta de um exercício e a atualização do ambiente deve ser o menor possível, estima-se que essa atualização deve ocorrer entre 1 e 3 segundos.
   Observação: O tempo estimado dos tópicos 1.2 e 1.3 variam de acordo com a conexão do usuário, a estimativa informada é baseada em testes com conexão local.
2. Garantir a segurança dos dados dos professores e estudantes.
3. O sistema deve rodar em qualquer tipo de browser (navegador) que possua o *plug-in* do Flash Player.
   3.2. Deve ser possível interagir com o sistema através de dispositivos móveis.
4. O sistema deve desassociar o estudante automaticamente que já tenha completado 1 (um) semestre (6 meses), pois subtende-se que o mesmo passou na disciplina.
5. Possuir fácil navegabilidade entre os exercícios e o sistema como um todo.
6. Possibilitar a utilização por usuários não avançados.

4.2.1.3 Requisitos Inversos

1. Caso o estudante já tenha concluído algum exercício da metodologia proposta, o exercício do mesmo não poderá ser mais alterado.

4.2.2 Modelagem do AVAA

Diante dos requisitos descritos na seção anterior, foram identificados os seguintes atores e funcionalidades.

Professor

Descrição:	Qualquer professor que lecione na instituição de ensino matérias relacionadas à área de desenvolvimento.
Responsabilidades:	É de responsabilidade do professor, avaliar os arquivos e exercícios postados no ambiente virtual.

Estudante

Descrição:	Qualquer estudante que estude na instituição de ensino que

	o ambiente foi implantado, estude matérias relacionadas à área de desenvolvimento, com um razoável conhecimento técnico em browser e internet.
Responsabilidades:	É de responsabilidade do estudante, conferir as respostas enviadas ao professor e submetidas no ambiente virtual.

Administrador

Descrição:	Usuário responsável pela configuração do ambiente virtual.
Responsabilidades:	Organizar e configurar o ambiente de forma que atenda aos requisitos estruturais da instituição (cursos, turmas, salas).

Fonte: Própria, 2013

4.2.2.1 Restrições

A seguir são apresentadas as restrições para que ocorra um bom funcionamento do sistema AVAA.

1. Apenas um exercício pode ser respondido por vez.
2. Todos os exercícios iniciados devem ser terminados. Se o exercício for abandonado, fica em aberto, para que dá próxima vez possa ser continuado.

4.2.2.2 Suposições e Dependências

1. Para que o estudante, professor ou administrador possam interagir, é necessário que ele possua um computador com conexão à Internet, qualquer tipo de browser que suporte o *plug-in* do Flash Player.
2. Para dispositivos móveis, dependendo da plataforma, talvez seja necessário adicionar algum tipo de *framework.*

4.2.2.3 Visão Geral do Sistema

Esse sistema tem como usuário MASTER o administrador, representado pelo ator Administrador, que é o usuário que administra e configura o ambiente virtual para atender a instituição corrente. Ele é responsável pela alocação dos professores aos cursos e disciplinas. O ideal é que o Administrador ocupe um cargo de coordenadoria do curso de computação da instituição, pois se subtende que ele tem uma visão macro do curso em questão.

Os discentes e docentes, que estão representados pelos atores estudante e professor, têm acessos idênticos, embora com visões diferentes. Eles estão vinculados a disciplinas e cursos, através de exercícios, arquivos, avisos, arquivos e jogos.

O professor pode estar alocado em mais de um curso e em várias disciplinas dentro de um curso. Uma das funções do professor é replicar um exercício feito para disciplina X de um curso A e replicar para o curso B de uma disciplina Y. O AVAA não trabalha com turmas, o professor pode alocar estudantes às disciplinas que leciona ou autorizar a entrada deles na disciplina. Essa estratégia foi pensada para os estudantes dessemestralizados.

Para a modelagem dos dados a ser utilizada e implementada no sistema, utilizou-se o diagrama de entidade relacionamento (DER). Neste diagrama, identificaram-se as entidades (elementos do problema), seus atributos (características pertinentes) e relacionamentos existentes entre objetos (entidades).

4.2.3 Tecnologias

No projeto foram usados as seguintes ferramentas:

a) Linguagem de programação: Java
b) Banco de Dados: MySql
c) Ferramentas de Interface: Eclipse (Java / Adobe Flex SDK)

Todas as ferramentas podem ser obtidas de graça nos respectivos sites dos fabricantes, o que possibilita um baixo custo de instalação e manutenção das ferramentas.

O projeto utilizou o Java como *back-end* pela facilidade de criar um serviço de comunicação único (*WebService*) que contém toda regra de negócio do sistema como um todo, com isso é possível criar várias interfaces, para diversos tipos de dispositivos sem precisar alterar o modelo de classes e persistência com o banco de dados.

A tecnologia de *front-end* usada no processo de desenvolvimento é o Adobe Flex, que tem como objetivo trazer o dinamismo de uma aplicação desktop para web (RIA - *Rich Internet Application*), facilitando a interação do usuário com o sistema. O SGBD utilizando foi o MySql pelo seu desempenho para ambientes WEB, multiplataforma e pouco exigente se tratando de hardware.

O AVAA pode ser implantado em qualquer sistema operacional que suporte Java e o banco de dados MySql. Espera-se também que os professores e estudantes sintam-se confortáveis ao interagir com a interface, e seguros para usá-lo e aprender através do ambiente virtual através de qualquer dispositivo que disponha de um browser e internet.

4.2.4 Arquitetura Utilizada

O MVC (*Model View Controller*) é um padrão de arquitetura usado em diversos tipos de aplicações. Seu principal objetivo é separar a lógica de programação e a interface gráfica do próprio software, resultando em uma enorme versatilidade da aplicação, além de tornar muito mais fácil modificar o aspecto visual da aplicação ou o código sem criar dependências/afetação entre eles. Facilita, dessa maneira, a comunicação entre usuário, interface gráfica e código da aplicação. O MVC se divide da seguinte forma:

a) *Model* – Representa a informação/dados da aplicação e as “regras/definições” para manipular/trabalhar com esses mesmos dados da aplicação. No Flex, geralmente *class* pessoais ou classes de serviços para lidar com um back-end;
b) *View* – Representa os elementos gráficos da aplicação, como *inputText*, *datagrid*, *textArea*. O Flex engloba *states*, *viewstacks*, em resumo, todos os componentes gráficos;
c) *Controller* – Representa o tipo de controle/detalhes que envolvem a comunicação com o *Model* (Dados e definições) e o *View* (Interfaces gráficas). Estes detalhes resultam da lógica de comunicação entre o *Model* e o *View* e normalmente representam também a interação com o usuário. No AVAA, o responsável é Java mais poderia ser utilizado o PHP.

Este padrão MVC foi adaptado por centenas de frameworks, inclusive o Flex. O principal objetivo de uso do MVC no Flex é a simplicidade, reutilização do código, e fácil comunicação entre os componentes, sem falar em uma otimização do desempenho da aplicação. O próprio Flex/Framework já está baseado nesta arquitetura e possui o seu próprio padrão MVC definido, como os componentes que definem a interface do usuário, os modelos de apresentação de dados e os componentes responsáveis pelo controle de dados como interações com linguagens *back-end*.

A justificativa para adotar esse padrão na construção do AVAA é a simplicidade e a organização de código, permitindo criar uma *class* controlador que recebe/lida com dados de um conjunto de classes (*model*) e que faz a devida atualização na interface gráfica/componentes (*view*).

4.2.5 *Prototipação do Sistema*

Figura 1 - Tela de Login

Fonte: Própria, 2013

A Figura 1 exibe a tela de *login* do sistema, utilizada por todos os usuários para a verificação e validação de acesso ao mesmo. Só o administrador pode criar o usuário professor. O usuário estudante pode se cadastrar e pedir permissão para entrar em uma determinada disciplina. O professor, então, permite ou não a participação do estudante na disciplina. Inicialmente, esse processo é manual, mas a intenção é que futuramente o AVAA interaja com o sistema da instituição de ensino, para buscar os estudantes que estão cadastrados no portal do estudante.

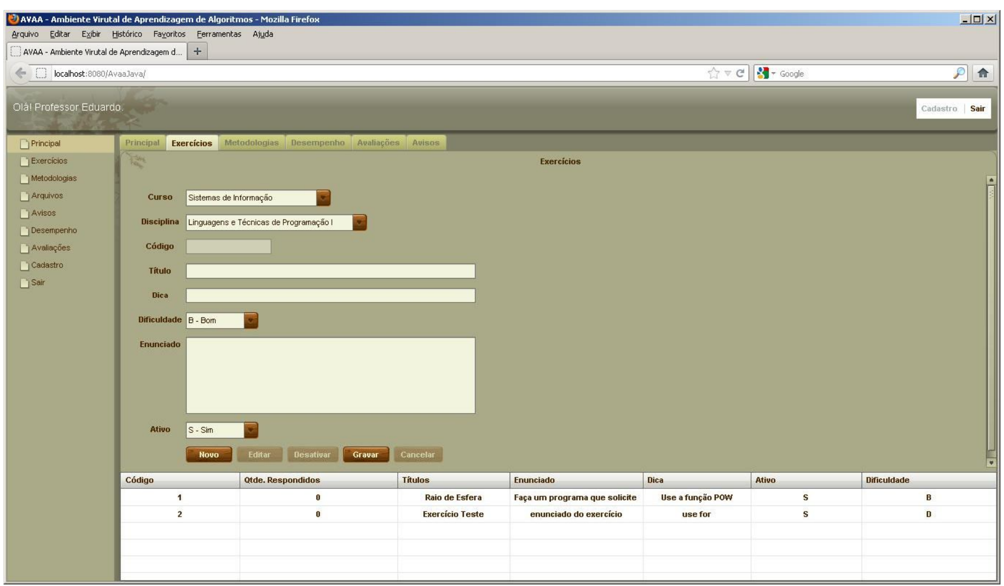

Figura 2 - Tela de Gerenciamento de Exercícios do Professor

Fonte: Própria, 2013

A Figura 2 mostra a tela de gerenciamento dos exercícios. O AVAA é um ambiente virtual multicursos, ou seja, o professor pode lecionar a mesma disciplina em diferentes cursos. Embora as disciplinas possam ser as mesmas (e.g. Linguagem de Programação I), subtende-se que o contexto dos exercícios pode diferir. O professor escolhe previamente o curso e a disciplina, informa o título, dica e enunciado do exercício.

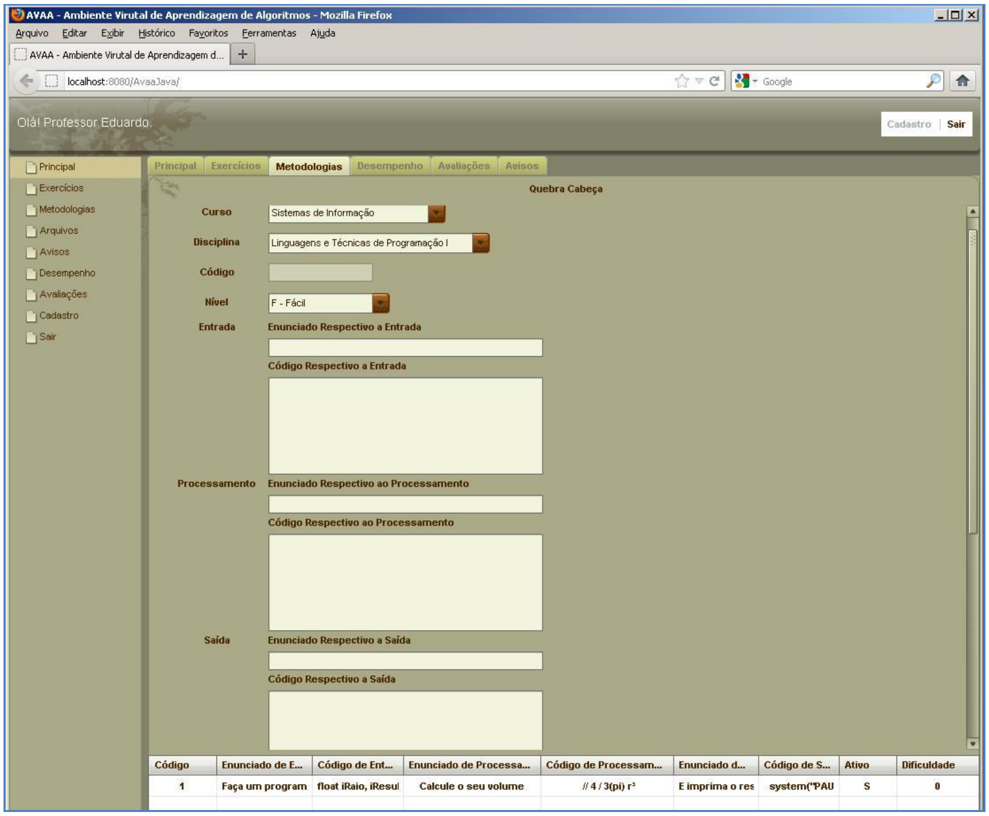

Figura 3 - Tela de Gerenciamento do Jogo Quebra Cabeça

Fonte: Própria, 2013

A Figura 3 mostra o jogo padrão do AVAA, o “quebra-cabeça”. Esse jogo utiliza a estratégia de dividir um problema complexo em vários problemas de resolução simples. O campo nível interfere na forma como o quebra cabeça quebrará o exercício:

F – Fácil. A entrada, processamento e saída são quebrados em blocos de código no formato em que são cadastrados;

B – Bom. A entrada, processamento e saída são quebrados em linhas de código;

F – Difícil. A entrada, processamento e saída são quebrados em palavras;

A Figura 4 mostra os elementos que são considerados para que o estudante seja avaliado. Essa avaliação é incremental, ou seja, a cada exercício respondido, o nível do estudante será incrementado. Atualmente, o AVAA não valida o código que foi enviado pelo aluno, através de um analisador léxico, sintático ou o próprio compilador da linguagem de programação informada (essa opção teria que ser implementada antes do aluno submeter o exercício), e não permite algoritmos modularizados em arquivos; toda essa validação é feita pelo professor. O critério utilizado para avaliação do aluno é a soma dos seguintes resultados: tempo de resposta, avaliação do professor de 1 a 10 e dificuldade do exercício. A Figura 5 ilustra a tela principal do estudante, com avisos e exercícios disponíveis.

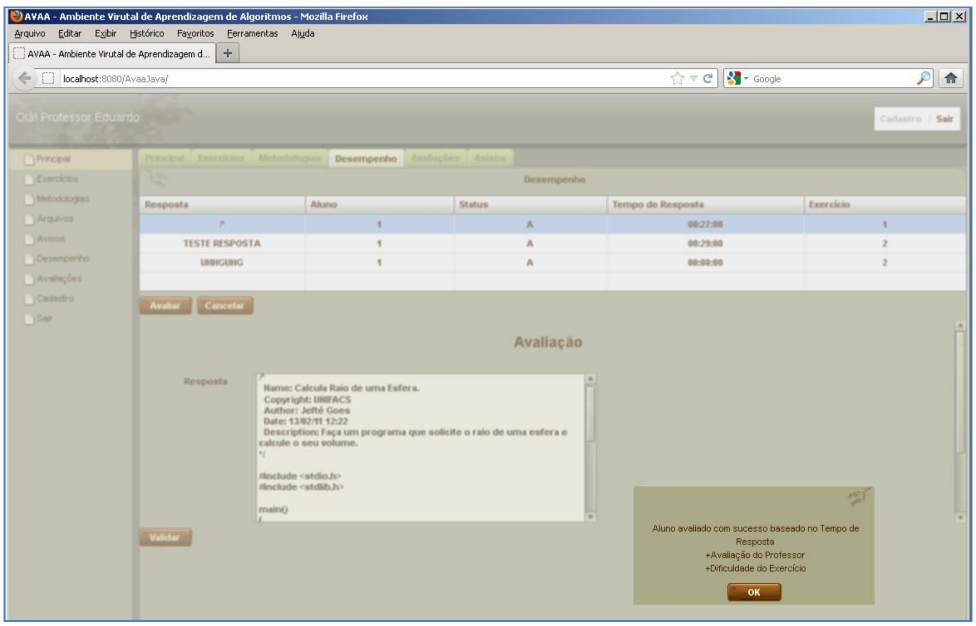

Figura 4 - Tela de Avaliação Exercício x Estudante

Fonte: Própria, 2013

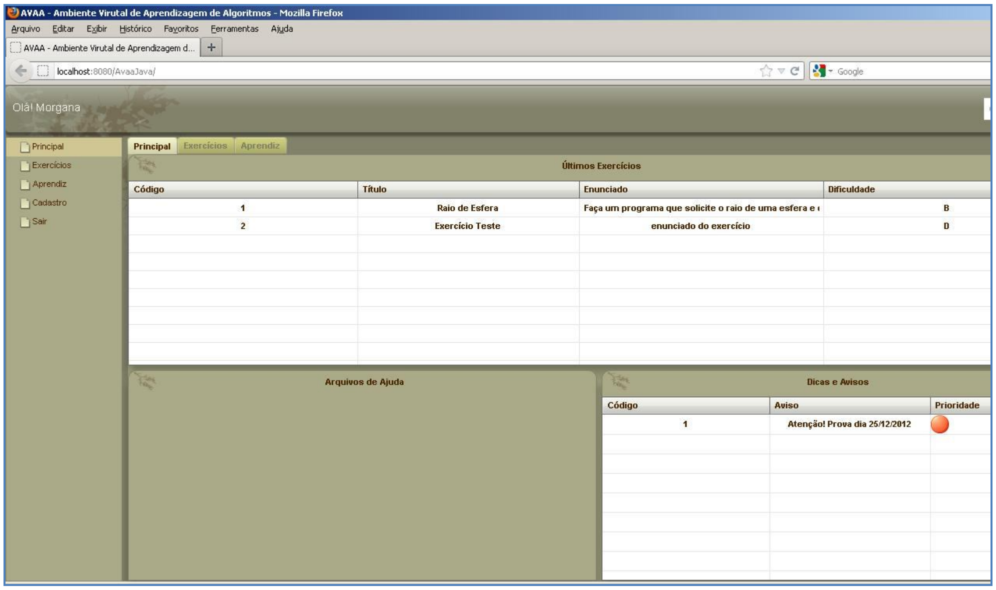

Figura 5 - Tela Principal do Estudante

Fonte: Própria, 2013

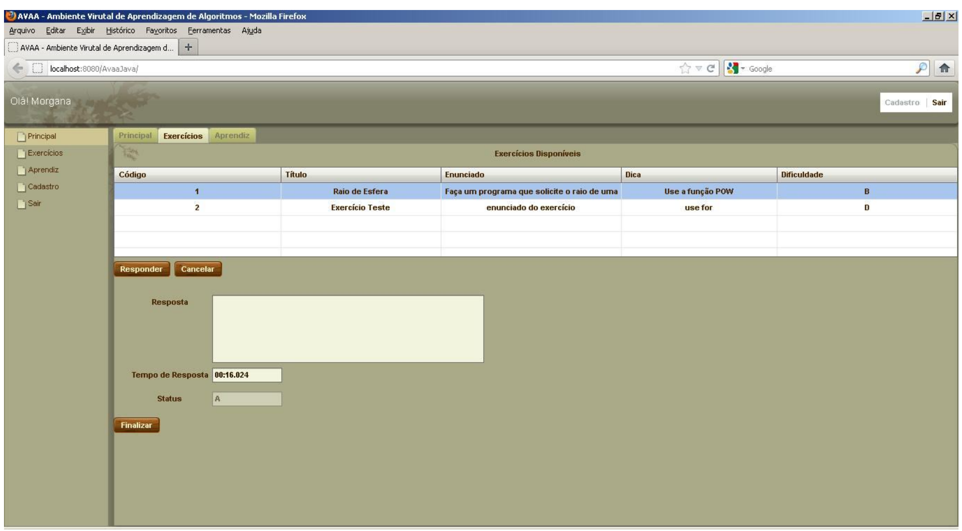

Figura 6 - Tela de Resposta do Exercício do Estudante

Fonte: Própria, 2013

A Figura 6 mostra o estudante solicitando responder o exercício. No momento que ele solicita, um contador de tempo é ativado. Esse tempo irá influenciar na avaliação; se o estudante tirar o foco da tela, o contador é pausado, ou seja, o contador só continua se o estudante estiver interagindo com o ambiente virtual; se o estudante abandonar o exercício, o mesmo consta como aberto para que da próxima vez possa continuar do ponto onde parou. A Figura 7 ilustra a metodologia em forma de jogo, de resposta aos exercícios propostos, explicada anteriormente na seção 3.

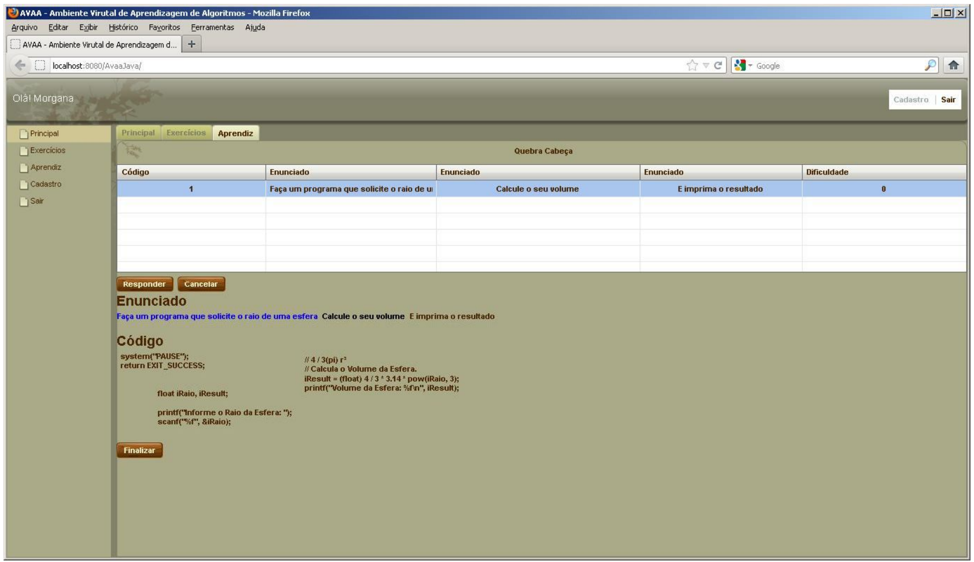

Figura 7 - Tela de resposta do estudante ao Jogo proposto pelo professor

Fonte: Própria, 2013

5 CONCLUSÃO

O uso de ambientes virtuais de aprendizagem tem demonstrado resultados relevantes nos processos de ensino e aprendizagem. Este artigo apresentou a proposta de uma ferramenta de apoio ao processo de ensino-aprendizagem de algoritmos. A ferramenta permite a construção de um conjunto de exercícios em forma de quebra-cabeça que, uma vez propostos pelos professores e resolvidos pelo estudante, podem ser utilizados de forma mais efetiva para o aprendizado.

Um dos aspectos na utilização do jogo de quebra-cabeça é o fator motivador, podendo-se destacar sua importância no processo de ensino-aprendizagem. Convém lembrar que os jogos propostos têm como objetivo desenvolver o raciocínio lógico, indispensável à construção de algoritmos e programação dentro de um ambiente virtual de aprendizagem.

O projeto surgiu em função das fragilidades encontradas nas disciplinas que envolvem algoritmos e linguagens de programação e na vivência com a monitoria, na qual foi identificada a necessidade de uma ferramenta para ajudar o professor com alternativas e estratégias que estimulassem e facilitassem a aprendizagem dos alunos.

Espera-se que o AVAA se torne um ambiente virtual de aprendizagem para os discentes do curso de sistemas de informação e áreas afins, e mais uma alternativa para os professores sugerirem atividades e exercícios de aprendizagem de algoritmos, que podem ser resolvidos presencialmente ou virtualmente através da internet. Durante o desenvolvimento deste projeto, embora algumas dificuldades tenham se apresentado, não foram impeditivos para a sua realização, mesmo que de forma parcial.

Com algumas funcionalidades implementadas, tais como o cadastro de exercícios do jogo quebra cabeça, avisos, tela de resolução de exercícios, o ambiente ainda não está completo. Outras funcionalidades, como comparador de exercícios avaliativos (provas), compilador *online* que contemple algumas linguagens de programação (Java, C, pascal, PHP) e exercícios modularizados, ainda não foram implementados. Ainda é necessário, além da conclusão do desenvolvimento da ferramenta, que os testes e validações sejam feitos. Com a implementação de novas funcionalidades e melhorias necessárias, antes não previstas no modelo, revisões e redesenho do modelo proposto deverão ser feitos, assim como redefinições de requisito funcionais.

Após o desenvolvimento da ferramenta, serão construídos formulários, questionários e a elaboração das entrevistas, como parte do estudo de campo exploratório com os alunos que interagirão com o AVAA. Com os resultados da aplicação dos instrumentos e a avaliação da ferramenta, será possível melhorar as funcionalidades e usabilidade do sistema.

A tecnologia Adobe Flex, usada atualmente, passa por um processo de migração da Adobe para Apache Foundation, já que há uma tendência que os projetos em Flash/Flex migrem para HTML5. Futuramente, o projeto será migrado para ASP. NET MVC 4, sendo reaproveitada a maior parte do projeto, principalmente a parte de persistência e classes. Acredita-se que o que foi apresentado pode ser aplicado em outras disciplinas, tanto da área de informática, quanto de outras áreas, contribuindo para o processo de ensino-aprendizagem dos estudantes de cursos de graduação de áreas afins.

REFERÊNCIAS

SOMMERVILLE, I. **Engenharia de Software**. 8th ed., Addison - Wesley, 2007.

LAKATOS, Eva Maria; MARCONI, Marina de Andrade. **Fundamentos da metodologia científica**. 7. ed. São Paulo: Atlas, 2010.

PEREIRA, Alice T. Cybis. (Org.). AVA - **Ambientes Virtuais de Aprendizagem em Diferentes Contextos**. Rio de Janeiro: Editora Ciência Moderna, 2007.

Carlos, José Rocha Pereira Júnior; Rapkiewicz , Clevi Elena. **Um Ambiente Virtual para apoio a uma Metodologia para Ensino de Algoritmos e Programação**, Dissertação de Mestrado. Universidade Estadual do Norte. Fluminense – UENF, 2005.
Disponível em: <http://seer.ufrgs.br/renote/article/viewFile/14021/7911>. Acesso em: 13 mar. 2013.

Pereira, Rodrigo dos Santos; Augustus, Heitor Xavier Costa. **Análise de Metodologias e Ambientes de Ensino para Algoritmos, Estruturas de Dados e Programação aos iniciantes em Computação e Informática**, UFLA – Universidade Federal de Lavras, 2005. Disponível em: <http://www.dcc.ufla.br/infocomp/artigos/v5.1/art06.pdf>. Acesso em: 13 mar. 2013.

FALKEMBACH, G. A. M., AMORETTI, M. S. M., TAROUCO, L. R., VIERO, F. Aprendizagem de Algoritmos: Uso da Estratégia Ascendente de Resolução de Problemas. In: TALLER INTERNACIONAL DE SOFTWARE EDUCATIVO, 8., 2003. Santiago. **Anais...**. Santiago, Chile, 2003.
Disponível em: <http://200.17.137.110:8080/licomp/Members/rfidalgo/pea/pratica-de-ensino-de-algoritmo/didatico/05_aprendizagem_de_algoritmos-estrategia-ascendente.pdf>. Acesso em: 13 mar. 2013.

ENDY, Uirá Ribeiro. **Um ambiente virtual de aprendizagem de colaborativo para o ensino de algoritmos**. 2008. 101 f. Dissertação de Mestrado. Pontifica Universidade Católica de Minas Gerais, 2004. Disponível em:
<http://www.biblioteca.pucminas.br/teses/Informatica_RibeiroUE_1.pdf>
Acesso em: 13 mar. 2013.

DETERS, J. I., MIRANDA, E. M., FERNANDES, A. M. R. O Desafio de Trabalhar com Estudantes Repetentes na Disciplina de Algoritmos e Programação. In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO, 19., 2008. Ceará. **Anais...** Novembro de 2008. UFCE – Ceará.
Disponível em:
<http://www.proativa.virtual.ufc.br/sbie/CD_ROM_COMPLETO/workshops/workshop%202/O%20Desafio%20de%20Trabalhar%20com%20Estudantes%20Repetentes%20na.pdf.>
Acesso em: 13 mar. 2013.

Menezes, C. S.; Nobre, I. A. M. Um ambiente cooperativo para apoio a cursos de introdução a programação. In: CONGRESSO DA SOCIEDADE BRASILEIRA DE COMPUTAÇÃO, 22., 2002. Florianópolis. **Anais...** Florianópolis: UFSC, 2002. 1 CDROM.
Disponível em: <www.virtual.ufc.br/aires/UNOPARVIRTUAL/textos/aprende.pdf>. Acesso em: 13 mar. 2013.

COLAÇO, Methanias Rodrigues Júnior. **Experiências Positivas para o Ensino de Algoritmos**. 2004. 9 f. Artigo ERBASE. Centro de Ciências Formais e Tecnologia –

Universidade Tiradentes (UNIT), 2004. Disponível em:
<http://www.uefs.br/erbase2004/documentos/weibase/Weibase2004Artigo001.pdf> Acesso em: 13 mar. 2013.

Teleduc. **Teleduc Ambiente Virtual de Ensino a Distância.** Disponível em:
<http://www.teleduc.org.br/pagina/apresentacao/>.
Acesso em: 08 maio. 2013.

Moodle, **Moodle Ambiente Virtual de Ensino a Distância.** Disponível em:
<http://docs.moodle.org/24/en/About_Moodle>.
Acesso em: 08 maio. 2013.

TAROUCO, L**. Jogos educativos**, 2005Disponível em:
<http://penta3.ufrgs.br/animacoes/JogosEducacionais> Acesso em: 09 maio. 2013