Utilização de Problemas da Maratona de Programação e Juízes Eletrônicos como Estratégia de Ensino em um Curso de Graduação em Engenharia de Software

Tiago Gomes Pereira[1], **Edson Alves da Costa Júnior**[1], **Márcia Barros de Sales**[2]
André Barros de Sales[1]

[1]Faculdade UnB Gama
Universidade de Brasília (UnB) – Gama, DF – Brasil

[2]Departamento de Ciências da Administração
Universidade Federal de Santa Catarina (UFSC) – Florianópolis, SC – Brasil

tiagogomespereira@gmail.com, edsonalves@unb.br, marcia.barros@ufsc.br, andrebdes@unb.br

Abstract. *This study aimed to analyze the use of a teaching strategy based on programming contests and electronics judges and their influence on the performance of students in the subjects in an undergraduate course of Software Engineering at the University of Brasilia. The methodology in this study is exploratory in nature, it is a documentary qualitative research, involving a case study. Among the results it was found that these students performed better in other subjects after these disciplines, and some of these students achieved superior performance in programming disciplines compared to students who did not study with this teaching strategy.*

Resumo. *Este trabalho objetivou analisar a utilização da estratégia de ensino alicerçada nas maratonas de programação e no uso de juízes eletrônicos e a influência desta estratégia no desempenho dos alunos nas disciplinas do curso de Engenharia de Software da Universidade de Brasília. A metodologia neste estudo é de cunho exploratória, é uma pesquisa documental de caráter qualitativo, envolvendo um estudo de caso. Nos resultados encontrados foi constatado que esses alunos tiveram um melhor desempenho nas demais disciplinas após cursar as disciplinas que utilizaram a estratégia citada, e que alguns desses alunos obtiveram desempenho superior em disciplinas de programação em comparação a alunos que não tiveram contato com essa estratégia de ensino.*

1. Introdução

Os sistemas de informação (SI) fazem parte do cotidiano das pessoas, possibilitam ao indivíduo ter acesso a milhares de informações em diferentes contextos, complexidades e realidade. Sommerville afirma que o mundo moderno não poderia existir sem os SI [Sommerville 2011].

Estes SI são utilizados para tomadas de decisões, assim esperam-se desses SI qualidade da informação e rapidez no seu acesso. Para tanto, são necessários que as informações processadas e utilizadas para estas decisões estejam exatas, sejam confiáveis e atualizadas para dar segurança e garantia para o tomador de decisão agir com eficiência.

DOI: 10.5753/cbie.sbie.2016.210

O número de pessoas interessadas nos recursos e nas funções oferecidas por determinadas aplicações tem crescido significativamente [Pressman e Maxim 2016].

Os SI se tornaram indispensável para a infraestrutura, para a indústria, para a economia, para a ciência e para o entretenimento, de modo que o profissional de software capacitado se tornou essencial para o mercado [Pressman e Maxim 2016]. Neste contexto, o Engenheiro de Software é um dos profissionais responsáveis pelo desenvolvimento, manutenção destes SI garantindo essa qualidade. Para que isso seja possível é necessário que ele conheça bem os fundamentos da ciência da computação.

A necessidade de profissionais capacitados na área da computação pode ser notada pelo número de iniciativas e de pesquisas para popularização da programação. Um exemplo é o Code.org, uma organização sem fins lucrativos que possui como objetivo popularizar o ensino de Ciência da Computação nas escolas e fazer com que ela seja parte do currículo escolar. Um exemplo de pesquisa é a utilização de softwares educacionais para o ensino de programação [Marcolino e Barbosa 2015], [Sales e Dantas 2010].

Contudo, ensinar programação não é algo simples. O ensino de programação é uma grande dificuldade encontrada pelos professores de ensino superior de cursos de computação [Fassbinder et al. 2012, de Sales et al. 2013]. Esta dificuldade contribui para o alto índice de evasão dos alunos de cursos de computação [INEP 2013].

Existem diversos motivos para a alta evasão, entre eles a falta de conhecimento do aluno sobre os fundamentos básicos do curso, a dificuldade em perceber a lógica para resolver um problema, falta de dedicação do aluno, falta de interesse na linguagem de programação adotada ou a dificuldade de aprender através da estratégia de ensino adotada pelo docente [Fassbinder et al. 2012].

A falta de dedicação por parte do aluno e a dificuldade em perceber a lógica necessária para resolução de problemas é, em grande parte, devido aos exercícios abordados estarem fora da realidade do aluno e não possuírem um contexto ou utilidade prática na visão dele, levando a uma falta de motivação em aprender [Fassbinder et al. 2012].

Na Universidade de Toronto foram utilizadas competições de programação internas nos cursos introdutórios de Ciências da Computação para motivar os alunos a aprender programação [Rosembloom 2009]. Já na Faculdade de Computação da Universidade Nacional de Singapura foi utilizado um juiz eletrônico [Cheang et al. 2003] para automatizar o processo de correção e avaliação das tarefas de programação dos cursos introdutórios de programação.

Segundo [Cheang et al. 2003], juízes eletrônicos são programas que fornecem mecanismos de correção automática para os problemas propostos. A correção é feita através de testes unitários, e contempla desde a compilação e execução até a validação dos resultados de cada teste unitário. O problema a ser resolvido deve ser especificado através da descrição do conjunto completo de entradas e o retorno de cada instância do problema [Skiena e Revilla 2008].

A utilização de maratonas de programação[1] e juízes eletrônicos no ambiente

[1]O *ACM International Collegiate Programming Contest – ICPC*, conhecido no Brasil como Maratona de Programação, é uma competição de programação dividia em três etapas, na qual alunos de universidades do mundo inteiro competem com o objetivo de definir quais são os melhores programadores do mundo.

de ensino tem como objetivo principal proporcionar a oportunidade e o incentivo necessários para o desenvolvimento dos alunos quanto à capacidade de resolução de problemas [Fassbinder et al. 2012] e automatizar o processo de avaliação [Cheang et al. 2003].

Os problemas das maratonas de programação em sua maioria possuem o seguinte formato: descrição do problema, descrição das entradas e saídas, exemplos de entradas e saídas e notas de rodapé com dicas relacionadas ao problema ou descrições extras [Halim e Halim 2013]. A descrição do problema apresenta e explica o problema, e costuma vir acompanhada de uma história ou explicação de um conceito. O problema é uma abstração de um problema computacional em uma situação real, e é a partir da análise da descrição do problema que o competidor será capaz de determinar em qual categoria de problema aquele problema se encaixa [Skiena e Revilla 2008].

Diante do exposto emerge a seguinte pergunta de pesquisa: Quais são os benefícios que a utilização de juízes eletrônicos nos moldes utilizados na maratonas de programação pode proporcionar aos alunos de engenharia de softwares? Para responder a esta pergunta o presente trabalho analisou a estratégia de ensino alicerçada nas maratonas de programação e nos juízes eletrônicos e a influência desta estratégia no desempenho dos alunos nas disciplinas do curso de graduação em Engenharia de Software da Universidade de Brasília – UnB.

Este artigo está organizado da seguinte forma: a Seção 2 apresenta os materiais e método, enquanto a Seção 3 descreve sobre aquisição e tratamento dos dados. A Seção 4 mostra os resultados e discussões, e na Seção 5 apresenta as considerações finais. Por último, são apresentadas as referências bibliográficas.

2. Materiais e Métodos

Este estudo é de cunho exploratório, sendo uma pesquisa documental, visto que valer-se-á de materiais que já receberam tratamento analítico mais que foram reexaminados por meio de interpretações quantitativas para o atendimento do objetivo da pesquisa [Godoy 1995]. Foram utilizados o tratamento de dados quantitativos, elaboradas métricas para analisar o desempenho dos alunos e apresentados os resultados de forma descritiva, envolvendo um estudo de caso realizado na Faculdade do Gama da UnB.

A estratégia de usar problemas da maratona de programação e juízes eletrônicos consistiu em elaborar problemas que abordassem o conteúdo da disciplinas, mas que tivessem o mesmo formato [Halim e Halim 2013] dos problemas da maratona de programação. Estes problemas foram aplicados aos alunos em listas de exercícios, simulados e provas, sendo que as duas últimas atividades foram, em sua maioria, presenciais, e avaliadas em tempo real por um juiz eletrônico. Os juízes utilizados ou foram desenvolvidos pelo próprio docente da disciplina ou eram juízes utilizados em competições, como o BOCA [de Campos e Ferreira 2004].

O desempenho dos alunos foi calculado utilizando uma média ponderada das menções obtidas em cada disciplina cursada, levando em consideração apenas as disciplinas onde os alunos obtiveram como menção SR, II, MI, MM, MS e SS[2] (de modo que foram excluídos os casos de créditos concedidos e trancamentos) e o número de créditos

[2]Na Universidade de Brasília, as notas finais dos alunos nas disciplinas são convertidas em menções, segundo o seguinte critério: SR (0), II (0,1 a 2,9), MI (3 a 4,9), MM (5 a 6,9), MS (7 a 8,9) e SS (9 a 10).

da disciplina. Para cada uma dessas menções foi atribuído um peso inteiro sequencial, de 0 a 5, respectivamente, na ordem listada das menções.

O coeficiente de desempenho D do aluno, conforme a Equação 1, é dado pela média ponderada dos pesos P_i pelo número de créditos C_i das menções obtidas em cada disciplina i, onde N é o número total de disciplinas de programação consideradas.

$$D = \frac{\sum_i^N C_i \cdot P_i}{\sum_i^N C_i} \tag{1}$$

Primeiramente, foram utilizadas rotinas para computar o coeficiente de desempenho dos alunos que pudessem ser utilizadas para computar o coeficiente D para cada aluno, dadas as N disciplinas a serem consideradas no cálculo. Em seguida foi analisado o desempenho dos alunos em matérias de programação anteriores e posteriores às turmas que tiveram como estratégia de ensino o uso de juiz eletrônico.

Por fim, foi feita a análise de turmas de disciplinas do curso. Para isso foram selecionadas as disciplinas Estruturas de Dados e Algoritmos II e Paradigmas de Programação, que possuem em suas respectivas ementas o foco em programação. A capacidade do aluno em desenvolver algoritmos eficientes está intimamente ligada ao desempenho alcançado nestas disciplinas.

Em cada uma destas turmas foi feito um histograma das menções e identificado quais alunos fizeram as disciplinas que utilizaram juiz eletrônico e quais alunos não as fizeram antes das referidas disciplinas. Após essa separação foi obtida a média do desempenho dos alunos naquela disciplina que não tiveram contato com juiz eletrônico e dos alunos que tiveram contato.

Para analisar o desempenho dos alunos individualmente observou-se se houve ou não melhora em seu desempenho acadêmico nas matérias onde a ementa fosse composta, essencialmente, de assuntos relacionados a programação. Para isso foram calculados os índices de desempenho anterior (IDA) e posterior (IDP), os quais contabilizam o desempenho do aluno levando em consideração as menções alcançadas por ele nas disciplinas cursadas no períodos anteriores (ou posteriores, respectivamente) ao semestre, e o próprio semestre, no qual cursou uma disciplina com juiz eletrônico. Também foram computados as diferenças absolutas δ (Equação 2) e percentuais Δ (Equação 3) entre estes dois índices.

$$\delta = IDP - IDA \tag{2}$$

$$\Delta = 100 \cdot \frac{\delta}{IDA} \tag{3}$$

Para analisar o desempenho individual dos alunos em disciplinas de programação antes e depois da utilização de estratégia de ensino foram considerados o índice de desempenho anterior em disciplinas de programação ($IDAP$), o índice de desempenho posterior em disciplina de programação ($IDPP$) e suas respectivas diferenças.

Todas as rotinas citadas e resultados do estudo estão disponíveis em um repositório Git púbico, hospedado na plataforma GitHub[3], e podem ser encontrados na integra em [Pereira 2015].

[3]`https://github.com/runys/students_history_parsing`

3. Aquisição e Tratamento dos Dados

O recurso necessário para a investigação do impacto da estratégia de uso de juízes eletrônicos é o desenvolvimento de rotinas e procedimentos que permitam auferir o desempenho dos alunos antes e após a exposição dos mesmos à esta estratégia.

A criação do curso de Engenharia de Software da UnB aconteceu no 2o semestre de 2008. Foram analisados os dados de todos os alunos de todos os semestres do curso até a obtenção dos dados, totalizando 13 semestres, ou seja do segundo semestre de 2008 até o 2o semestre de 2014. Foram identificadas e selecionadas as disciplinas que utilizaram juiz eletrônico, as disciplinas que não utilizam juiz eletrônico e as disciplinas programação antes e depois das disciplinas com juiz eletrônico.

As disciplinas analisadas foram divididas em dois grupos. O primeiro grupo são as das disciplinas que utilizaram os problemas da maratona de programação e juízes eletrônicos como estratégia de ensino e avaliação. Foram analisadas nove (9) turmas que estão listadas a seguir na Tabela 1.

Tabela 1. Disciplinam que utilizaram juízes eletrônicos

Número	Nome	Ano	Semestre
1	Estruturas de Dados e Algoritmos	2012	2
2	Introdução aos Jogos Eletrônicos	2012	2
3	Estruturas de Dados e Algoritmos	2013	1
4	Introdução aos Jogos Eletrônicos	2013	1
5	Programação Para Competições	2013	1
6	Introdução aos Jogos Eletrônicos	2013	2
7	Programação Para Competições	2013	2
8	Programação Para Competições	2014	1
9	Programação Para Competições	2014	2

O segundo grupo consiste nas disciplinas do curso de Engenharia de Software da Faculdade UnB Gama ou do curso de Ciência da Computação da Universidade de Brasília (neste caso, apenas as que constam em algum dos históricos analisados) que possuem o ensino de programação como foco de sua ementa ou de seu plano de ensino, num total de 24 disciplinas. São elas: Computação Básica, Desenho de Software, Desenho Avançado de Software, Estruturas de Dados e Algoritmos, Estruturas de Dados e Algoritmos 2, Estrutura Matemáticas para Computação, Fundamentos de Compiladores, Fundamentos de Sistemas Operacionais, Inteligência Artificial, Introdução a Ciência da Computação, Introdução a Computação Gráfica, Introdução ao Desenvolvimento de Jogos, Introdução aos Jogos Eletrônicos, Manutenção e Evolução de Software, Métodos de Desenvolvimento de Software, Orientação a Objetos, Paradigmas de Programação, Programação para Competições, Programação Web, Sistemas Embarcados, Técnicas de Programação, Tópicos Especiais em Jogos Digitais, Tópicos Especiais em Programação e Tópicos Especiais em Sistemas Críticos.

O intuito desse segundo grupo é de avaliar separadamente o desempenho dos alunos nessas disciplinas. Observe que, embora algumas destas disciplinas citadas tenham utilizado os juízes eletrônicos, a maioria não o fez.

Para preservar a identidade dos alunos, os históricos foram tratados de forma a

excluir todos os dados pessoais de cada aluno (nome, idade, sexo, pai, mãe, matrícula). Para fins de análise, cada aluno recebeu um ID numérico inteiro e sequencial.

4. Resultados e Discussão

A análise dos dados obtidos através das rotinas citadas na Seção 2 foi realizada a partir de dois pontos de vista: do desempenho individual dos alunos e do desempenho dos alunos dentro de disciplinas específicas.

Os 388 alunos de Engenharia de Software incluídos no estudo consistem em todos aqueles que foram alunos regulares do curso de Engenharia de Software, considerado o intervalo de tempo desde o início do curso (segundo semestre de 2008) até o segundo semestre do ano de 2014, num total de 13 semestres letivos. Destes alunos, 133 (34%) cursaram uma ou mais dentre as disciplinas citadas na Tabela 1.

4.1. Análise do ponto de vista do desempenho individual dos alunos

O objetivo foi de observar se a dinâmica de estudo necessária para obter aprovação em uma disciplina que utilize um juiz eletrônico como método de avaliação impacta de alguma forma o desempenho geral do estudante. A primeira abordagem foi analisar o desempenho geral do estudante, através dos coeficientes IDA e IDP e suas respectivas diferenças.

A Tabela 2 apresenta uma amostra destes coeficientes para alunos que cursaram a disciplina com juiz eletrônico, ordenados por pela diferença percentual Δ.

Tabela 2. Amostra do desempenho geral dos alunos

Aluno (ID)	D	IDA	IDP	δ	Δ
221	3,34	3,05	4,34	1,30	42,56%
005	2,61	2,37	3,23	0,87	36,55%
251	3,11	2,72	3,67	0,95	35,10%
020	2,95	2,81	3,79	0,98	34,78%
002	2,32	1,82	2,45	0,63	34,42%
041	3,28	2,70	3,57	0,87	32,03%
132	3,12	2,64	3,46	0,82	31,13%
316	3,34	3,09	4,04	0,94	30,47%
312	3,44	3,12	4,04	0,92	29,52%
227	3,24	2,77	3,58	0,80	28,96%
...	...	...	...	...	...
139	2,84	3,68	2,61	-1,06	-28,93%
044	3,23	3,24	2,21	-1,03	-31,73%
054	2,09	2,88	1,89	-0,99	-34,26%
362	2,63	2,89	1,88	-1,01	-34,88%
343	1,96	2,37	1,40	-0,97	-40,89%
187	2,18	2,37	1,13	-1,23	-52,20%
299	2,50	2,10	1,00	-1,10	-52,38%
361	2,11	2,56	1,18	-1,39	-54,09%
379	1,81	2,00	0,89	-1,11	-55,56%
283	2,04	2,04	0,00	-2,04	-100,0%

Dentre os 133 alunos que tiveram contato com a estratégia de uso de juízes eletrônicos, 93 alunos (69,9%) obtiveram um aumento no coeficiente de desempenho D e 40 alunos (30,1%) obtiveram uma queda no desempenho.

Em seguida, foi analisado o desempenho dos alunos restrito apenas às 24 disciplinas de programação listadas anteriormente, utilizando-se os coeficiente $IDAP$ e $IDPP$. A Tabela 3 apresenta uma amostra destes coeficientes para alunos que cursaram a disciplina com juiz eletrônico, ordenados por pela diferença percentual Δ.

Tabela 3. Amostra do desempenho em matérias de programação

Aluno (ID)	D	$IDAP$	$IDPP$	δ	Δ
078	2,63	1,73	3,00	1,27	73,68%
029	2,33	1,75	2,64	0,89	50,65%
194	2,86	2,33	3,30	0,97	41,43%
030	3,62	3,62	5,00	1,38	38,16%
110	3,20	2,75	3,60	0,85	30,91%
164	3,84	3,63	4,71	1,08	29,88%
220	1,87	1,71	2,14	0,43	25,00%
067	3,22	2,70	3,33	0,63	23,46%
150	3,73	3,62	4,45	0,83	22,88%
245	3,31	3,00	3,67	0,67	22,22%
359	3,57	3,41	4,14	0,74	21,58%
312	3,00	2,77	3,33	0,56	20,37%
...	...	...	...	...	...
146	2,71	3,43	2,43	-1,00	-29,17%
155	3,22	3,42	2,33	-1,09	-31,79%
279	2,41	2,22	1,50	-0,72	-32,50%
032	1,78	2,67	1,73	-0,94	-35,23%
244	3,00	3,80	2,33	-1,47	-38,60%
113	2,04	3,00	1,84	-1,16	-38,60%
139	2,61	4,00	2,22	-1,78	-44,57%
280	2,00	2,12	1,00	-1,12	-52,94%
044	2,33	3,00	1,33	-1,67	-55,56%
361	1,75	2,00	0,50	-1,50	-75,00%
283	2,00	2,00	0,00	-2,00	-100,00%
362	2,36	2,59	0,00	-2,59	-100,00%
339	2,33	2,62	0,00	-2,62	-100,00%

Do total de 133 alunos, 67 (50,3%) obtiveram aumento no desempenho em programação, 60 (45,1%) apresentaram uma queda no desempenho e 6 (4,5%) mantiveram seus desempenhos. Um fato notável é que para 49 deste 67 alunos (73,1%) o aumento foi superior a 5 pontos percentuais.

Estes dados indicam que o contato com a estratégia de ensino pode ter promovido uma melhor fixação os conceitos de programação no aprendizado do aluno e ampliado a motivação de uma parte significativa do grupo de alunos em aprender programação. Porém, como o número de alunos com aumento é percentualmente equivalente ao número de alunos com queda no desempenho, podemos avaliar que a inserção dessa estratégia de ensino em disciplinas pontuais não foi suficiente para promover uma mudança significativa na atitude do grupo de alunos como um todo.

Vale notar também que o único critério para determinar se um aluno teve ou não contato com problemas de maratona e juízes eletrônicos foi o registro, em histórico, de uma das disciplinas listadas na Tabela 1 com menção superior a SR. Deste modo, não foram levados em conta, nesta análise, as reprovações, evasões (nas matérias em questão

e no próprio curso de Engenharia de Software) e desistências, o que eleva o número de alunos com queda no desempenho.

4.2. Análise do ponto de vista do desempenho dos alunos dentro de disciplinas específicas

As turmas de programação escolhidas para serem analisadas foram oriundas das disciplinas Estruturas de Dados e Algoritmos 2 (EDA 2) e Paradigmas de Programação (Paradigmas) do segundo período de 2014. Foram escolhidas essas turmas porque haviam entre os seus alunos alguns daqueles que cursaram disciplinas com juiz eletrônico anteriormente.

Estas disciplinas foram selecionadas também por possuírem em sua ementa tópicos relacionados a programação que necessitam de conhecimentos prévios providos pelas disciplinas que utilizaram juiz eletrônico como apoio na estratégia de ensino e na avaliação.

Foram excluídos desta análise os alunos que trancaram a disciplina, os que solicitaram o aproveitamento da disciplina (crédito concedido ou aproveitamento de estudos) e os alunos que obtiveram SR como menção. A Tabela 4 apresenta as menções finais dos alunos nestas disciplinas e outros dados correlacionados.

Tabela 4. Turmas do segundo semestre de 2014 avaliadas

Disciplina	Menção	Total de alunos	Alunos com juiz	Alunos sem juiz
EDA 2	SS	3	3	0
	MS	4	3	1
	MM	9	4	5
	MI	3	1	2
	II	4	2	2
Paradigmas	SS	3	3	0
	MS	5	2	3
	MM	19	7	12
	MI	5	2	3
	II	6	3	3

Com base nas menções dos alunos nas disciplinas é possível observar que os alunos com as maiores menções são, em sua maioria, alunos que tiveram contato anteriormente com juiz eletrônico.

Para a analise do desempenho médio desses alunos nessas turmas foi somado o peso de cada menção dos alunos e depois dividido pelo número de alunos, conforme apresentado na Tabela 5.

Tabela 5. Comparação do desempenho médio

Disciplina	Período	Média com juiz	Média sem juiz
EDA 2	02/2014	3,30	2,50
Paradigmas	02/2014	3,00	2,71

A Tabela 5 apresenta que a média de desempenho dos alunos que tiverem contato com juiz eletrônico é de 32% superior a média dos outros alunos na disciplina de

Estrutura de Dados e Algoritmos 2, e 10.7% superior a dos outros alunos na disciplina de Paradigmas de Programação.

Como citado anteriormente, o rendimento superior por pare destes alunos pode ser proveniente da melhor assimilação dos fundamentos necessários para aprendizado de conceitos mais avançados. Outro fator pode ser a motivação desses alunos: ao serem apresentados a problemas próximos à realidade, a serem resolvidos precisamente e em um curto espaço de tempo, estes alunos passam a ver a programação como uma ferramenta de trabalho e podem vir a desenvolver uma disciplina de estudo que envolva também a velocidade na elaboração das soluções e a validação e teste contínuos de suas rotinas, habilidades necessárias para se obter uma avaliação positiva de um juiz eletrônico.

5. Considerações Finais

Esse trabalho analisou a utilização da estratégia de ensino alicerçada nas maratonas de programação e juízes eletrônicos e a sua influência no desempenho dos estudantes do curso de Engenharia de Software da Universidade de Brasília de modo global e em algumas disciplinas do curso.

Entre as influências encontradas, quanto ao desempenho individual, com base nos resultados da análise de desempenho geral do aluno, observou que 69,9% dos alunos tiveram um desempenho maior do que o desempenho anterior a esse contato. Ainda sobre a análise desempenho individual destes alunos em disciplinas de programação antes e depois da utilização desta estratégia de ensino, observou-se que 50,3% dos alunos apresentaram um aumento no desempenho em disciplinas de programação.

Do ponto de vista do desempenho dos alunos nas disciplinas Estruturas de Dados e Algoritmos 2 e Paradigmas de Programação, foi observada que a média de desempenho dos alunos que tiverem contato com os problemas da maratona de programação e juízes eletrônicos é de 32% superior a média dos outros alunos na disciplina de Estrutura de Dados e Algoritmos 2 e 10.7% superior a média dos demais alunos na disciplina de Paradigmas de Programação.

Pode-se supor que o rendimento superior por parte destes alunos pode ser proveniente de uma melhor assimilação dos fundamentos necessários para aprendizado de conceitos mais avançados, que foram instigados ou induzidos por meio dos desafios em resolver os problemas de maratona de programação, ora de forma individual ora em pequenos times/grupos, e no feedback imediato do juiz eletrônico.

Outra influência pode ser a motivação desses alunos, pois ao serem apresentados a problemas próximos de sua realidade, eles passaram a ver a programação como uma ferramenta de trabalho com aplicações reais e se tornaram mais ávidos em aprender novos conceitos e técnicas.

Um engenheiro de software deve conhecer os fundamentos básicos da computação para que possa propor e projetar a melhor solução para atender as necessidade dos clientes, habilidade provida pela gama de conhecimentos adquiridos para ser competitivo em uma maratona de programação. A utilização de juízes eletrônicos e problemas oriundos de maratonas de programação, com base nas analise desse trabalho do desempenho dos alunos, melhora a formação dos alunos de Engenharia de Software.

Como continuação deste trabalho está o estudo da possibilidade de unificar a base

de dados da universidade acerca do histórico dos alunos, sem o acesso a informações que os identifique. Assim, será possível a implementação de uma aplicação que consulte a base de dados de alunos e, a partir de um filtro aplicado pelo usuário, mostrar gráficos, tabelas e índices para que seja possível acompanhar os desempenhos dos alunos sob várias óticas, em buscas de metodologias e estratégias que promovam uma melhor formação dos alunos.

Referências

Cheang, B., Kurmia, A., and Oon, W. (2003). On automated grading of programming assignments in an academic institution. In *Computers & Education*, volume 41, pages 121–131.

de Campos, C. P. and Ferreira, C. E. (2004). Boca: um sistema de apoio a competições de programação. In *Workshop de Educação em Computação*. Salvador.

de Sales, A. B., de Moura del Esposte, A., da Silva, J. P. L., and de Sales, M. B. (2013). Recursos didáticos digitais para auxiliar na aprendizagem da linguagem c. In *XV Simpósio Internacional de Informática Educativa*, volume 1, pages 51 – 55. Viseu.

Fassbinder, A. G. O., Paula, L. C., and Araujo, J. C. D. (2012). Experiências no estímulo à prática de programação através do desenvolvimento de atividades extracurriculares relacionadas com as competições de conhecimentos. In *Congresso da Sociedade Brasileira de Computação (CSBC)*.

Godoy, A. S. (1995). Introdução à pesquisa qualitativa e suas possibilidades. In *RAE – Revista de Administração de Empresas*, volume 35, pages 57–63. São Paulo.

Halim, S. and Halim, F. (2013). *Competitive Programming 3*. Lulu.

INEP (2013). Censo da educação do ensino superior 2013.

Marcolino, A. S. and Barbosa, E. F. (2015). Softwares educacionais para o ensino de programação: Um mapeamento sistemático. In *Simpósio Brasileiro de Informática na Educação*.

Pereira, T. G. (2015). Utilização de juízes eletrônicos e problemas oriundos da maratona de programação no ensino de programação da faculdade unb gama: Um estudo de caso. Trabalho de conclusão de curso, Universidade de Brasília – Faculdade UnB Gama. Curso de Gradução em Engenharia de Software.

Pressman, R. S. and Maxim, B. R. (2016). *Engenharia de Software: Uma Abordagem Profissional*. AMGH Editora, 8 edition.

Rosembloom, A. (2009). Running a programming contest in an introductory computer science course. In *ACM SIGCSE Bulletin*, volume 4, page 347. ACM.

Sales, C. G. and Dantas, V. F. (2010). Progame: Um jogo para o ensino de algoritmos e programação. In *Simpósio Brasileiro de Informática na Educação*.

Skiena, S. S. and Revilla, M. A. (2008). *Programming Challenges: The Programming Contest Training Manual*. Springer.

Sommerville, I. (2011). *Engenharia de Software*. Pearson Prentice Hall, 9 edition.