Comunidades de Investigação em Programação: Uma Estratégia de Apoio ao Aprendizado Inicial de Programação

Scheila W. Martins, António J. Mendes e António D. Figueiredo

***Title*— Research Communities in Programming: A Strategy to Support the Initial Programming Learning.**

***Abstract*— This paper describes a research process that intends to develop a strategy to support initial programming learning. It is inspired in the theory of Matthew Lipman, as it was conceived through the redefinition of the concept of Learning Communities, to assist the students to maximize their learning through the conscientious evaluation of their level of self-efficacy, while they develop a better behavior to study programming. We present the results of the preliminary tests made to the proposed strategy, and the changes in the strategy that result from those results.**

***Index Terms*— Computer Science Education, Programming Education, Self-efficacy**

I. Introdução

As propostas e pesquisas relacionadas com a área de informática e educação têm como objetivo disponibilizar recursos e contextos que auxiliem professores e estudantes a aumentar a efetividade do aprendizado. Quanto ao aprendizado de programação, os avanços neste contexto de pesquisa são inegáveis, com inúmeras ferramentas e ambientes já disponibilizadas.

Há um intenso esforço de pesquisadores e professores em tentar compreender porquê o aprendizado de programação, ainda é um obstáculo para um número crescente de estudantes [1, 2]. Além disso, há todo o empenho dos governos em ações para trazer melhores condições para o aprendizado nas diversas etapas do sistema educacional: desenvolvendo políticas de inclusão social e digital, investimentos em reforma de infra-estrutura, informatização e aparelhagem de salas de aulas e laboratórios, e dos programas de incentivo à qualificação docente.

Apesar de tudo isso, essas reformas não conseguiram ainda produzir um sistema que se desenvolva na direção do "ensinar a pensar". Atualmente um elevado número de estudantes do nível elementar ao secundário não desenvolvem diversas das habilidades e competências acadêmicas necessárias para evoluir de maneira mais produtiva na vida universitária, conforme análise dos resultados dos Programas Internacionais de Avaliação do Ensino [3].

O grande problema com o ensino de programação não se resume só na dificuldade dos estudantes com abstração para resolver problemas. Há ainda a dificuldade em encontrar uma forma que motive os estudantes a se envolverem com a disciplina, apesar da dificuldade, e a não desistirem de tentar ultrapassar as dificuldades naturais e inerentes deste aprendizado [4]. Fazê-los compreender que os obstáculos são superáveis, que as habilidades necessárias podem ser melhoradas, incentivando-os a desenvolver e a consolidar um conjunto de competências acadêmicas essenciais para que eles evoluam com melhor qualidade nos estudos e na futura vida profissional.

O número crescente de estudantes no sistema universitário tem sobrecarregado o modelo acadêmico tradicional. Num curto espaço de tempo a academia se vê com um elevado número de estudantes, com pouco tempo para modificar o seu modelo de trabalho e despreparada para atender adequadamente essa demanda. Na urgência de encontrar modos de gerenciar esta situação, muitas vezes, a academia opta por processos que privilegiam as questões administrativas (formato de aulas, composição de turmas, alocação de recursos físicos e humanos), o que do ponto de vista didático, nem sempre é a melhor solução. Uma transformação do modelo acadêmico para se adequar à nova realidade é uma necessidade, mas é um processo que está em desenvolvimento, onde as reformas de natureza administrativas acabam por influenciar a evolução dos processos didácticos.

Para passar da reflexão para ações de fato, é importante que os resultados das mudanças desejadas sejam documentados formalmente, num conjunto de medidas que

S. W. Martins é aluna de doutorado do Centro de Informatica e Sistemas da Universidade de Coimbra - CISUC, Pólo II, Pinhal de Marrocos, 3030–290, Coimbra, Portugal (tel: +351 239790078; fax: +351-239701266; scheila@ dei.uc.pt).

A. J. Mendes, é docente do Departamento de Engenharia Informática da Universidade de Coimbra - DEI e investigador do CISUC (tel: +351 239790036; fax: +351-239701266; toze@ dei.uc.pt)

A D. Figueiredo, é investigador do CISUC (tel: +351 239790021; fax: +351-239701266; adf@ dei.uc.pt)

DOI (Digital Object Identifier) Pendiente

tenham sido previamente testadas em contextos tão próximos quanto possível da realidade atual.

II. Aprendizado de Programação

Parece existir um consenso entre professores e pesquisadores de que o aprendizado de programação não é uma atividade trivial, já que introduz no cotidiano do estudante uma série de requisitos cognitivos, para além dos requisitos técnicos. Isso cria a necessidade de modificar o seu modo de pensar e de agir na vida universitária, para uma realidade diferente daquela a que ele se habituou durante anos no ensino primário e secundário. Tudo isso em um espaço muito curto de tempo. Entre esses requisitos podemos destacar:

1) A resolução de problemas é antes de tudo uma competência que envolve processos cognitivos como criatividade e racionalidade, a partir de um conjunto de meta-habilidades mentais que muitas vezes passam desapercebidas (abstração, inferência, dedução, etc), e que se apóiam em habilidades literárias básicas como ler e interpretar a descrição de um problema. Além disso, é necessário que o estudante aprenda a contextualizar o conhecimento adquirido para que ele consiga iniciar esses processos cognitivos, exercendo suas habilidades de raciocínio para construir uma solução. Tornar esses processos conscientes para apoiar a tomada de decisão é algo que demanda tempo e ritmo individualizados, que precisam ser respeitados e, sobretudo, incentivados;
2) A compreensão integral dos requisitos de um paradigma de programação não é uma atividade trivial, e implica em um grau de dificuldade intrínseco natural [5, 6]. Porém a maior dificuldade no aprendizado de programação reside no exercício da capacidade de abstrair o conhecimento adquirido para resolver problemas. As habilidades de abstração e de resolução de problemas são algo que só podem ser obtidos com prática contínua e esforço individual. Ter habilidades literárias bem desenvolvidas será de grande importância para bem fundamentar os processos cognitivos e dar suporte ao aprendizado em construção;
3) Apesar da informática, desde muito cedo, fazer parte do cotidiano dos estudantes, o nível de transparência embutido na relação usuário X computador atualmente envolve-os em uma falsa noção de autonomia inteligente dos sistemas informáticos. Por vezes é difícil fazê-los compreender a noção de transparência e responsabilidade da relação programador X computador.

As capacidades de abstração e de resolução de problemas estão geralmente pouco desenvolvidas nos estudantes que chegam do ensino secundário. A falta de excelência no seu desenvolvimento é ainda acrescida pela superficialidade das habilidades literárias básicas [7, 8], essenciais para desenvolver as habilidades técnicas propostas nos programas das disciplinas de programação.

De fato, o grande problema não chega a ser a linguagem ou o paradigma de programação, mas a dificuldade de desenvolver competências, que tornem o estudante capaz de contextualizar seu conhecimento para resolver problemas. É importante fazer o estudante entender que programar é, antes de tudo, um exercício consciente habilidades básicas de raciocínio (ler, escrever, calcular) e atos mentais (comparar, descrever, classificar) que se desenvolvem e oferecem um contexto apropriado para dar suporte ao desenvolvimento de diversas componentes cognitivos e habilidades de pensamento complexo e sofisticado (recursivo, autocorretivo, metacognitivo, investigativo, metódico) [9,10,11]. Aliás, na medida que se ganha excelência no desenvolvimento das habilidades mentais isso costuma influenciar o aprimoramento qualitativo das diversas habilidades literárias (ler, interpretar, sintetizar, criticar, etc) [11].

Assim, mais que conseguir definir uma metodologia de ensino-aprendizagem para programação, seria interessante conseguir estabelecer um conjunto de estratégias, que demonstrem aos estudantes que resolver problemas em programação é uma atividade que eles são plenamente capazes de realizar. É importante valorizar contextos e estabelecer uma dinâmica durante as aulas que possa motivar os estudantes ao trabalho em equipe, demonstrando que o aprendizado é possível, e que as dificuldades individuais são superáveis, na medida com que se conscientizem e se predisponham a "aprender a pensar" [12, 13].

Alguns pesquisadores apontam para uma reestruturação de currículos [14], mas esse ambiente pode ser implementado só pela modificação da forma de propor atividades e de avaliar o progresso da aprendizagem do estudante [15, 16]. Avanços podem ser obtidos por exemplo, com exercício regular de atividades ligadas a reflexão investigativa para a resolução de problemas, a abstração, a modelagem e a avaliação da qualidade de soluções algorítmicas [17, 18]. A partir daí, poderia haver um maior compromisso do estudante com seu aprendizado, até mesmo produzir mudanças comportamentais que melhorem sua performance ao longo do curso.

III. Motivação

A motivação constitui uma dos aspectos de grande impacto no desenvolvimento cognitivo do indivíduo, e um dos principais aspectos que determina o sucesso individual de um processo de aprendizado. Também está entre os aspectos mais fascinantes da psiquê humana, tendo na Hierarquização da Motivação de Maslow uma das suas principais origens de pesquisa [19]. Compreender a motivação para o aprendizado requer uma análise profunda de componentes sócio-cognitivos associados à personalidade (identificação com a instituição, com o curso, com a carreira, as condições de acesso ao ensino e a análise do sucesso escolar) e da qualidade dos

relacionamentos nos ambientes de convivência do indivíduo (identificação social e abordagens de aprendizagem) [20].

Ao longo da evolução do estudante no processo de ensino formal, desde o primário, ele vai sendo submetido a atividades, contextos, e metodologias de aprendizado. A partir dessas experiências o estudante vai desenvolvendo seu comportamento de estudo, descobrindo sua estratégia de aprendizagem, qual o método e as atividades que mais se associam em equilíbrio com sua personalidade, seus valores e suas crenças. No entanto, não é possível menosprezar o impacto exercido pela metodologia de aprendizado vivenciada, já que esta exerce grande influência no tipo, na quantidade e na qualidade das habilidades e competências literárias e acadêmicas que serão desenvolvidas. Isso acaba por se incorporar ao comportamento de estudo dos estudantes de toda uma geração.

A motivação se apresenta como um domínio de primordial impacto em projetos de ferramentas e metodologias de aprendizado [21], e diversas teorias e instrumentos já foram descritas para: classificar, medir, criar e manter a motivação dos estudantes, especialmente entre crianças e adolescentes [22]. Alguns instrumentos de avaliação cognitiva focam-se em aspectos específicos que afetam a motivação para o aprendizado. Com o avanço das pesquisas em ensino a distância (EAD), instrumentos inspirados no Modelo de Motivação ARCS (mnemónico para Atenção, Relevância, Confiança e Satisfação) de John Keller [23], tem sido muito utilizado na modelagem de cursos e ambientes de e-Learning.

Por exemplo, o Exame de Interesse do Curso (*Course Interest Survey* - CIS) e o Exame da Motivação dos Materiais Didáticos (*Instructional Materials Motivation Survey*-IMMS) [24], são dois instrumentos de avaliação da motivação inspirados no modelo ARCS. O CIS e o IMMS, foram desenvolvidos para avaliar, respectivamente, o nível de aceitação dos contextos e de satisfação das atividades propostas (por exemplo verificar a satisfação dos estudantes ao trabalharem com portifólios eletrônicos); e, a satisfação com os materiais didáticos (livro texto, apostilas, fichas) utilizados no curso da disciplina online. Segundo Keller, ambos podem ser adaptados para serem utilizados em disciplinas presenciais.

Outros instrumentos, se ocupam da avaliação de diversas medidas cognitivas que se relacionam com a motivação (conforto, auto-eficácia, satisfação, resistência), como o Inventário de Atitudes e Comportamentos de Estudo (IACHE) [25] por exemplo, podem auxiliar a desenvolver estratégias para avaliar aspectos relacionados com as estratégias de aprendizagem dos estudantes universitários. O IACHE é um teste de comportamento genérico, independente de uma disciplina, com o qual seria possível avaliar:

- Se uma metodologia de aprendizagem pôde ou não satisfazer um conjunto de requisitos sobre uma disciplina;
- Assinalar a existência de mudanças de atitude do estudante em relação a sua postura acadêmica;
- Estabelecer parâmetros estatísticos de uma população, identificando as proporções das suas dimensões cognitivas, motivacionais e comportamentais.

A análise do comportamento de estudo proposta pelo IACHE pode facultar algumas importantes informações sobre o comportamento dos estudantes, individualmente e enquanto grupo, possibilitando avaliar e/ou orientar a seleção de atividades, abordagens metodológicas e práticas pedagógicas.

No entanto, seria um erro supor que a motivação para algo fosse definitivo e imutável, da mesma maneira que seria ingênuo considerar que os estudantes são naturalmente desinteressados ou desmotivados com as atividades acadêmicas. Quase sempre o que se chama de desmotivação poderia na realidade ser chamado de resistência, uma reação de auto-preservação do estudante a uma potencial falha. A resistência dos estudantes a determinadas atividades ou assuntos podem encobrir um conjunto de fatores que afetam sua confiança para o êxito efetivo nessas atividades. Segundo Margolis [26], identificar o nível de resistência em atividades acadêmicas pode auxiliar na análise das questões de motivação que essas atividades despertam em alguns estudantes, e facilitar o trabalho de compreender o que está por trás de muitas dificuldades de aprendizagem. Muitas vezes as dificuldades de aprendizagem residem na dificuldade em exercitar habilidades literárias básicas, como leitura e interpretação.

O Questionário de Motivação Estudantil para a Resolução de Problemas (*The Student Motivation Problem Solving Questionnaire* - SMPSQ) [26] é diferente dos outros instrumentos descritos anteriormente, por não ser um instrumento científico formal. Não pode nem deve ser utilizado para prever comportamento, mas como uma forma incentivar o estudante a expressar suas expectativas de êxitos, pressupostos de falhas e o quão disposto está para investir tempo e energia para concluir uma atividade. Sua informação pode ser utilizada para identificar o nível de apreensão/confiança de cada estudante com uma atividade ou habilidade literária em particular, orientando o professor na elaboração de programas de reforço de habilidades aos estudantes através de atividades que causem menos resistência e aumentem seu envolvimento com a disciplina

Embora instrumentos como o IACHE e os testes baseados em ARCS tenham todos componentes para mensurar a motivação para o aprendizado, observou-se a necessidade de estabelecer uma medida de motivação mais especificamente associada com a aptidão dos estudante para aprender a programar, e que pudesse ser medida com total independência dos elementos observados nos instrumentos IACHE ou ARCS.

Essa medida de motivação é a auto-eficácia [27,28], avaliada a partir de escalas associadas diretamente com uma auto-análise da capacidade ou incapacidade de um indivíduo para executar alguma tarefa. As escalas de auto-eficácia para programação [29] são um instrumento formal que podem ser

utilizadas com alguma regularidade, e podem auxiliar a manter o estudante em alerta quanto à qualidade do seu aprendizado, ofertando uma outra perspectiva da avaliação da

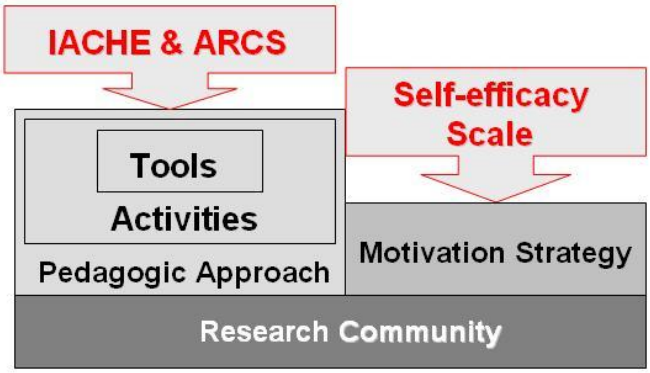

Fig. 1. Esquema da estratégia pedagógica proposta.

sua capacidade, diferente das notas em provas e trabalhos..

IV. UmA Proposta Para Auxiliar o Aprendizado de Programação

Esta pesquisa tem como objetivo conseguir definir um enquadramento teórico contendo um conjunto de recomendações de atividades didáticas, estratégias motivacionais e ferramentas computacionais que podem auxiliar o docente na definição de contextos de aprendizagem para disciplinas de programação conforme ilustrado na figura 1.

É uma proposta que se desenvolve sob uma abordagem pedagógica de comunidades de aprendizagem, tendo sido inspirada numa metáfora das Comunidades de Investigação de Matthew Lipman [11], o qual se considera uma abstração bastante pertinente para propostas que envolvem o desenvolvimento do pensamento crítico [30] e de habilidades literárias [31], e como estratégia para melhorar a capacidade de resolução de problemas em programação para estudantes universitários.

As habilidades literárias podem ser reforçadas em conjunto com o ensino de programação, pois como habilidades básica de raciocínio, darão suporte ao aprendizado das etapas de abstração e modelagem de uma solução algorítmica, no processo de desenvolvimento de software. Além disso, essa abordagem favorece o desenvolvimento e o exercício das habilidades de pensamento complexo, que serão necessárias aos estudantes para toda a vida, acadêmica, social e profissional. A dinâmica da proposta é implementar uma redefinição do que Lipman chamou de Comunidades de Investigação [11, 31], a partir da retomada do ensino socrático como abordagem didática.

As comunidades de aprendizagem constituem uma já conhecida perspectiva, tanto educacional quanto tecnológica, que permite a criação de contextos nos quais é possível potencializar a aprendizagem, pela associação entre atividades de colaboração e de produção de conhecimento [32]. A proposta de Lipman é um apelo aos professores para insistirem que suas práticas sejam focadas na orientação do estudante na busca pelo conhecimento: em estimular seus alunos a aprenderem a identificar conscientemente o que eles já sabem e o que precisam saber para solucionar um problema, inclusive para poderem refletir e atestar a qualidade de uma solução. Ainda que o discurso filosófico puro não constitua uma abordagem natural para a prática docente em programação, o pensamento metódico e reflexivo que provém do diálogo inspirado na filosofia da ciência lhe é bastante útil.

A dinâmica das Comunidades de Investigação define uma abordagem pedagógica que possibilitaria ao professor trabalhar com os mais diversos contextos: Aprendizado Baseado em Problemas (PBL) [33], *Just-in-Time Teaching* (JiTT) [34], projetos *hands-on* [35], jogos [36] e competições de programação [37]. Este é o primeiro alicerce dessa proposta, ou seja, o professor deve escolher um dos contextos pedagógicos sugeridos para determinar quais atividades didáticas irá desenvolver durante a disciplina. Estes têm aparecido na literatura como exemplos bem sucedidos para aprendizagem em programação e nos parecem adequados à abordagem que preconizamos.

Na escolha do contexto devem ser considerados alguns aspectos, como tamanho da turma, que pode tornar a utilização de um contexto mais trabalhosa que outros. Sugerimos fortemente a utilização dos projetos *hands-on* e do JiTT, uma vez que o primeiro pode facilitar o envolvimento e interesse dos estudantes na criação de trabalhos práticos variados, e o segundo disponibiliza um conjunto de atividades, em seu JiTT *classroom*, que estão em grande sintonia com os objetivos propostos pelas Comunidades de Investigação. Alias, JiTT oferecem oportunidades excelentes ao desenvolvimento de atividades investigativas e de pesquisa em uma disciplina de programação [38].

Após ter sido escolhido o contexto, a dinâmica das Comunidades de Investigação será desenvolvida com atividades didáticas projetadas para fortalecer o envolvimento do aluno com o processo de aquisição de conhecimento, através do trabalho cooperativo e do incentivo ao exercício das suas habilidades literárias. Uma certa quantidade de atividades que fomentem a pesquisa de conteúdo para resolução de problemas (desafios, defesas orais e seminários) e a avaliação de qualidade do processo e das soluções obtidas (incluindo auto-avaliação, do grupo, do mérito da atividade e dos resultados) serão apresentadas ao longo da disciplina. As atividades propostas nessa estratégia pedagógica objetivam incentivar tanto o desenvolvimento de competências para a resolução de problemas, quanto estimular o exercício habilidades literárias de diversas formas:

1) Atividades investigativas: questões de pesquisa, exercícios de avaliação discursiva da qualidade algorítmica de soluções, desafios práticos e/ou teóricos de programação em diferentes complexidades, etc;

2) Produção colaborativa/cooperativa de conhecimento: adoção de metodologias e ferramentas de colaboração em atividades de reforço, avaliação entre pares, incentivo à produção de portifólios, webquests, seminários e participação em competições;
3) Divisão de trabalho e acompanhamento: estimular o uso e o reconhecimento de monitorias voluntárias entre aluno-aluno e aluno-tutor;
4) Avaliação continuada: atividades de feedback regular entre professor, tutores e alunos sobre o andamento da disciplina.

O segundo alicerce da proposta é definido pela escolha das ferramentas tecnológicas, uma ferramenta de apoio ao aprendizado de programação e uma que dará suporte ao trabalho colaborativo. Consideramos uma mais valia a utilização de uma ferramenta de simulação algorítmica ou de programas, adequada à linguagem e ao paradigma em uso na disciplina (o RoboCod, o Alice, o BlueJ e o SICAS), bem como podem também ser usadas plataformas de competições e teste (o TopCoder, o Mooshake e o Online-Judge).

Além da disponibilidade de ferramentas dessa natureza ser grande, é interessante tentar minimizar o impacto que o uso de compiladores profissionais tende a causar, quando desvia a atenção dos estudantes da resolução de problemas para solucionar questões pertinentes ao ambiente de programação. Para estimular atividades fora dos horários de aula e apoiar as atividades de acompanhamento e avaliação continuada, é aconselhável que haja suporte de uma plataforma de gestão da aprendizagem (o Moodle, o Blackboard ou o AulaNet).

Finalmente, o último alicerce, se preocupa com necessidade de possuir uma estratégia para apoiar a manutenção dos índices de motivação para o aprendizado dos estudantes durante o trabalho em comunidade. Medidas de motivação como nível de conforto, auto-eficácia, confiança, satisfação e resistência com as atividades propostas devem ser verificadas com alguma regularidade, de modo a conseguir orientar o ritmo de trabalho e direcionar adequadamente os esforços do docente para ações de intervenção, no apoio e na prevenção a comportamentos de propensão a desistência entre os estudantes.

V. Uma Experiência com Alunos de Design e Multimídia

Uma vez que o número de estudantes nas disciplinas de programação no curso de Licenciatura em Engenharia e Informática (LEI) do Departamento de Engenharia Informática (DEI) da Faculdade de Ciência e Tecnologia da Universidade de Coimbra (FCTUC) costuma ser bastante elevado (entre 200 a 300 inscritos), a heterogeneidade dos perfis torna necessária uma avaliação cuidadosa na escolha das atividades e dos contextos a serem utilizados. Qualquer alteração na dinâmica pedagógica e na metodologia de trabalho em turmas numerosas implica em um cotidiano mais trabalhoso para o professor e um risco para propostas didáticas experimentais. Assim, para a definição e experimentação da estratégia proposta, foi utilizada uma disciplina do curso de Mestrado em Design e Multimídia (MDM).

Um primeiro teste foi realizado com os estudantes da disciplina de Programação (17 inscritos, sendo que 6 não chegaram a frequentar a disciplina), durante o período de Setembro de 2008 a Fevereiro de 2009. Os estudantes eram em sua maioria recém-formados de cursos de Licenciatura nas áreas de Design (Multimédia, Industrial ou Comunicação), além de uma licenciada em Arquitetura. A finalidade da disciplina é dar a esses estudantes um background mínimo em programação, facilitando a sua participação nas outras disciplinas do curso, as quais requerem conhecimento prévio em programação.

Foi utilizado como ferramenta de desenvolvimento a linguagem de programação Processing [39], uma ferramenta desenvolvida pelo MIT por e para designers gráficos, que facilita as capacidades de desenvolvimento de trabalhos artísticos da linguagem JAVA. Considerando o *background* artístico dos estudantes, a abordagem didática foi concebida para propiciar um aprendizado prático, tendo sido adotado um contexto voltado para projetos *hands-on* visuais de complexidade crescente.

O programa da disciplina foi desenvolvido para adotar a dinâmica das Comunidades de Investigação, privilegiando atividades de resolução de problemas em grupos, as quais envolviam uma certa necessidade de pesquisa e revisão dos conhecimentos matemáticos necessários à sua resolução, com acompanhamento constante do docente. No modelo de aulas da estratégia proposta, não houve distinção entre aulas teóricas, práticas e de prática-laboratorial, comuns ao modelo adotado na FCTUC. Assim, todas as aulas são espaços de construção de conhecimento e experimentação prática, perfazendo um total de 06 horas semanais de trabalho.

A disciplina contou ainda com um curso no servidor Moodle do DEI, no qual eram disponibilizadas os materiais e realizadas diversas das atividades da disciplina. Foi estimulada a avaliação qualitativa regular dos trabalhos desenvolvidos, bem como da satisfação dos estudantes com sua performance, com as atividades, os materiais e o ritmo das aulas. As atividades desenvolvidas incluíram: trabalhos práticos individuais e em grupos e um projeto final, além de avaliação entre pares, construção de portifólios individuais e reflexão quinzenal da evolução da disciplina e dos próprios estudantes.

A. Avaliação da Experiência

Ao final da disciplina os estudantes realizaram uma entrevista, composta por 8 questões relacionadas com as perspectivas do modelo ARCS, na qual apresentaram sua avaliação da disciplina, bem como de aspectos relativos à sua experiências na trajetória escolar. A análise das respostas levantadas durante as entrevistas, chamaram atenção:

• Das 05 disciplinas cursadas pelos estudantes naquele semestre, Programação e Técnicas de Internet foram as que causavam maior expectativa negativa, devido às experiências frustrantes que muitos relataram ter tido com programação durante a licenciatura;

• Os relatos de boas experiências e boas performances nas disciplinas do secundário e da licenciatura denunciam a importância da relação entre a didática e a postura do docente. Indicaram como positiva experiências nas disciplinas que se sentiram inspirados pelo docente, mesmo quando não se sentiam atraídos pelo conteúdo;

• Consideraram Programação uma grata surpresa entre as disciplinas que tinham cursado, destacando-a pela forma de condução das aulas, a ferramenta utilizada e sua forma de avaliação;

O resultado das entrevistas corrobora os aspectos identificados na análise das reflexões quinzenais (avaliação continuada). Entre os aspectos positivos apontados pelos estudantes, podemos destacar: as atividades de pesquisa e análise de código, o trabalho cooperativo e a dinâmica das aulas. Essas atividades são básicas para a abordagem proposta pelas Comunidades de Investigação, valorizando-a como uma abordagem interessante para o ensino de programação.

Surpreendentemente, as atividades de avaliação entre pares e a avaliação continuada, assim como os contextos artísticos propostos, aparecem como elementos avaliados positivamente, porém com uma frequência muito abaixo do esperado. Já entre os aspectos negativos mais pontuados na avaliação dos estudantes destacam-se: os exemplos matemáticos utilizados durante o conteúdo de *arrays*, o primeiro desafio de programação da disciplina envolvendo a animação de um modelo matemático de circunferência, e a falta de empenho individual dos estudantes. Nos aspectos negativos menos pontuados destacam-se: o atraso no *feedback* das notas pelo docente, o horário das aulas, funcionalidades da ferramenta Processing e aspectos sobre Programação Orientada a Objetos (POO).

Apenas um aluno não foi aprovado, e o docente se mostrou satisfeito tanto com os resultados obtidos quanto com a dinâmica da estratégia. Considerou que a forma de trabalho é mais produtiva, ainda que reconheça a existencia de um volume maior de trabalho que lhe foi exigida. Além disso, o docente indicou a necessidade de implementar modificações no modelo original da proposta. Uma das alterações substanciais a serem incorporadas na próxima configuração da disciplina e dos testes da estratégia inclui modificações:

1) Na organização do conteúdo, e;
2) No método de avaliação tanto dos estudantes em relação a disciplina, quanto na forma de avaliar o mérito da estratégia em pesquisa.

Quanto ao conteúdo, será feito um esforço para a introdução dos conceitos de POO mais cedo no programa, para que os estudantes tenham mais tempo para exercitar o uso do paradigma em trabalhos de programação específicos. Quanto a avaliação da disciplina, cogita-se a inclusão de mini-testes teóricos para além dos trabalhos individuais de programação e do projeto final em grupo. O mini-teste é uma oportunidade para avaliar individualmente o nível de abstração para a resolução de problemas desenvolvida pelos estudantes a medida que a disciplina avança.

B. Modificações na Avaliação da Estratégia

Esta experiência foi o primeiro teste de aproximação realizada para a estratégia relatada, sendo particularmente importante para esclarecer sob quais alicerces ela deveria evoluir. O reduzido número de estudantes em MDM, demonstrou-se propício para a avaliação e para se chegar a definição de boa parte dos elementos incluídas na proposta.

Do ponto de vista da avaliação do mérito da estratégia, modificações foram implementadas para o próximo teste. Para se obter a uma avaliação mais rigorosa e metódica sobre a eficiência desta estratégia enquanto alternativa mais cativante e efetiva para o aprendizado de programação, consideramos pertinente aferir algumas medidas cognitivas relacionadas à motivação para o aprendizado.

Assim, a estratégia proposta passará a fazer uso de alguns instrumentos psicológicos formais para avaliar diversos aspectos cognitivos envolvidos: um levantamento das características do comportamento de estudo através do IACHE, uma avaliação da motivação com o andamento do disciplina segundo os níveis cognitivos do modelo ARCS, os níveis de satisfação/resistência com as atividades realizadas através do SMPSQ, e uma medição dos níveis de auto-eficácia em programação dos estudantes.

Por não ser, a priori, relacionado a qualquer disciplina ou curso, os resultados do teste IACHE, podem ser utilizados para auxiliar o desenvolvimento de atividades para estudantes com determinadas características, bem como pode ser utilizado para avaliar a adequação de uma determinada abordagem de aprendizagem para uma determinada população. Neste caso, podemos utilizá-lo para melhor avaliar a validade da composição dos elementos (contextos, atividades, ferramentas de apoio) da estratégia proposta para um grupo de estudantes, inclusive avaliar o impacto da estratégia para a modificação do comportamento de estudo dessa população.

Assim como o IACHE, o CIS, pode identificar o nível de motivação dos estudantes à medida que a disciplina se desenvolve, a partir da verificação das dimensões cognitivas avaliadas pelo modelo ARCS. A análise das respostas do SMPSQ permitirá identificar o quanto de resistência/satisfação pode estar associada às atividades acadêmicas propostas. Finalmente, a aplicação regular de uma escala de auto-eficácia para programação pode conscientizar o estudante do seu nível de competência para executar tarefas com a linguagem e o paradigma em estudo, além de oferecer ao docente indicações sobre a necessidade ou não de medidas de intervenção.

O CIS será aplicado exatamente na metade do curso, e o SMPSQ após a realização das atividades investigativas e de produção colaborativa/cooperativa de conhecimento escolhidas pelo docente. Tanto o IACHE quanto a escala de auto-eficácia serão aplicados no esquema de pré e pós testes, no início e próximo ao término da disciplina, com os quais poderemos analisar aspectos cognitivos do grupo de estudantes antes e depois da experimentação de nossa estratégia.

VI. Conclusão

A avaliação geral deste primeiro experimento foi considerada positiva por parte dos estudantes, apesar de não contar com o apoio de todos instrumentos formais de avaliação cognitiva propostos pela estratégia atualmente. Sua realização foi particularmente importante para orientar as modificações necessárias e para planejar os novos cenários de teste. A experiência com os alunos de MDM pôde esclarecer a importância do uso de instrumentos formais de avaliação cognitiva como o IACHE, bem como estabeleceu a necessidade de utilizar uma medida de motivação, a auto-eficácia, e um instrumento formal para mensurá-la, para auxiliar o processo de manutenção do envolvimento dos estudantes durante o curso da disciplina.

É essencial destacar que o tamanho reduzido da turma foi determinante, pois tornou possível um acompanhamento quase individual aos estudantes. Tanto o *feedback* quanto o acompanhamento constante do docente evidenciaram ser extremamente importantes, já que permitiram uma melhor avaliação da qualidade e da aceitabilidade das atividades implementadas, bem como permitiu efetuar a tempo ações de adequação do ritmo das aulas e das atividades propostas. Essas ações foram necessárias para tratar questões de envolvimento e dispersão, especialmente conseguir recuperar estudantes com eminentes propensões à desistência da disciplina.

Alguns instrumentos e contextos propostos por essa proposta estão sendo testados dentro das disciplinas de programação dos cursos de Licenciatura em Engenharia Informática, de Licenciatura em Design Multimídia e nas Aulas de Apoio a Programação (Computer Programming Supporting Class). O apoio é um trabalho de voluntariado que oferece aulas de reforço em programação para todos estudantes da FCTUC. Um teste final dessa proposta será realizado novamente na disciplina de Programação do Mestrado em Design e Multimídia com início previsto para Setembro de 2009.

Agradecimentos

Os autores agradecem aos estudantes matriculados na disciplina de Programação do curso de Mestrado em Design e Multimídia do ano letivo 2008/2009, pela sua participação nas experiências da pesquisa em andamento.

Referências

[1] T. Jenkins, “On the difficulty of learning to program,” in 3rd Annual Conference of Learning and Teaching Support Network of Centre for Information and Computer Science LTSN-ICS, (United Kingdom), pp. 27-29, Loughborough University, The Higher Education Academy, Agosto 2002.

[2] E. Lahtinen, K. Ala-Mutka, and H.-M. Jaarvinen, “A study of difficulties of novice programmers,” in ITiCSE '05: Proceedings of the 10th annual SIGCSE conference on Innovation and technology in computer science education, (New York, NY, USA), pp. 14-18, ACM Press, 2005.

[3] N. B. Dohn, “Knowledge and skills for PISA -assessing the assessment,” Journal of Philosophy of Education, vol. 41, pp. 1-16, February 2007.

[4] E. Roberts, “Strategies for encouraging individual achievement in introductory computer science courses,” SIGCSE Bull., vol. 32, no. 1, pp. 295-299, 2000.

[5] B. C. Wilson and S. Shrock, “Contributing to success in an introductory computer science course: a study of twelve factors,” SIGCSE Bull., vol. 33, no. 1, pp. 184-188, 2001.

[6] S. Wiedenbeck, “Factors affecting the success of non-majors in learning to program,” in ICER '05: Proceedings of the First international workshop on Computing education research, (New York, NY, USA), pp. 13-24, ACM, 2005.

[7] D. Demerer, “Improving the learning environment in CS1 - experiences with communication strategies," SIGCSE Bulletin, vol. 25, pp. 31-38, September 1993.

[8] C. Bennett and T. Urness, “Using daily student presentations to address attitudes and communication skills in CS1,” in SIGCSE '09: Proceedings of the 40th ACM technical symposium on Computer science education, (New York, NY, USA), pp. 76-80, ACM, 2009.

[9] J. T. Havill and L. D. Ludwig, “Technically speaking: fostering the communication skills of computer science and mathematics students," in SIGCSE '07: Proceedings of the 38th SIGCSE technical symposium on Computer science education, (New York, NY, USA), pp. 185-189, ACM, 2007.

[10] H. A. Etlinger, “A framework in which to teach (technical) communication to computer science majors,” SIGCSE Bull., vol. 38, no. 1, pp. 122-126, 2006.

[11] M. Lipman, Thinking in Education. Cambridge University Press, 1st ed., 1991.

[12] M. Lindholm, “Object-orientation by immersion: Teaching outside the cs department,” in Report of ECOOP 2004 Eighth Workshop on Pedagogies and Tools for the Teaching and Learning of Object Oriented Concepts, (Berlin - Heidelberg), pp. 153-171, Springer-Verlag, 2004.

[13] G. Stegink, J. Pater, and D. Vroon, “Computer science and general education: Java, graphics, and the web,” SIGCSE Bull., vol. 31, no. 1, pp. 146{149, 1999.

[14] J. Mead, S. Gray, J. Hamer, R. James, J. Sorva, C. S. Clair, and L. Thomas, “A cognitive approach to identifying measurable milestones for programming skill acquisition,” in ITiCSE-WGR '06: Working group reports on Innovation and technology in computer science education, (New York, NY, USA), pp. 182-194, ACM, 2006.

[15] S. Kerka, “Techniques for authentic assessment. practice application brief.,” ERIC Clearinghouse on Adult, Career, and Vocational Education, Columbus, OH., 1995.

[16] M. K. Hamza, B. Alhalabi, and D. M. Marcovitz, “Creative pedagogy for computer learning: eight effective tactics,” SIGCSE Bull., vol. 32, no. 4, pp. 70-73, 2000.

[17] J. P. Denny, A. Luxton-Reilly, and B. Simon, “Evaluating a new exam question: Parsons problems,” in ICER '08: Proceeding of the fourth international workshop on Computing education research, (New York, NY, USA), pp. 113-124, ACM, 2008.

[18] J. Bennedsen and M. Caspersen, “Assessing process and product - a practical lab exam for an introductory programming course,” pp. 16-21, Oct. 2006.

[19] A. H. Maslow, Motivation and Personality. NY: Harper, 1st ed., 1954.

[20] M. V. Abreu, Cinco Ensaios sobre a Motivação. Livraria Almedina, 2nd ed., 2002. 57-96.

[21] P. C. Blumenfeld, E. Soloway, R. W. Marx, J. S. Krajcik, M. Guzdial, and A. Palincsar, “Motivating project-based learning: Sustaining the doing, supporting the learning,” Educational Psychologist, vol. 26, pp. 369-398, June 1991.

[22] S. de C. Martinelli and D. Bartholomeu, “Escala de motivação acadêmica: uma medida de motivação extrínseca e intrínseca,” Avaliação Psicológica, vol. 6, pp. 21-31, junho 2007.

[23] J. M. Keller, “First principles of motivation to learn and e3-learning,” Distance Education, vol. 29, pp. 175-185, August 2008.
[24] J. M. Keller, Motivational Design for Learning and Performance: The ARCS Model Approach (handbook). New York, NY: Springer, 1st, 2009.
[25] S. Monteiro, R. M. Vasconcelos, and L. S. Almeida, “Rendimento acadêmico: Influência dos métodos de estudos,” in Atas do VII Congresso Português de PsicoPedagogia, Universidade do Minho, 2005. Braga.
[26] H. Margolis. (January,2009). Student Motivation: A Problem Solving Focus [Online], Available http://www.reading2008.com/Motivation-Problem_Solving_Questionnaire-HowardMargolis-2009Jan1-c.pdf
[27] A. Bandura, “Self-efficacy: Toward a unifying theory of behavioral change,” Psychological Review, vol. 84, pp. 191-215, March 1977.
[28] H. Margolis and P. McCabe, “Improving Self-Efficacy and Motivation: What to Do, What to Say,” Intervention in School and Clinic., vol. 41, no. 4, pp. 218-227, March, 2006.
[29] V. Ramalingam, D. LaBelle, and S. Wiedenbeck, “Self-efficacy and mental models in learning to program,” SIGCSE Bull., vol. 36, no. 3, pp. 171-175, 2004.
[30] W. G. Huitt, “Critical thinking: An overview,” in Critical Thinking Conference in Barnesville, GA, (http://chiron.valdasta.edu/whuitt/col/cogsys/critthnk.html), Gordon College, may 1998.
[31] S. W. Martins, Um modelo computacional de apoio ao desenvolvimento do pensamento crítico. PhD thesis, Universidade Federal de Santa Catarina - UFSC, Brasil, Maio 2005.
[32] J. E. Minkler, “ERIC review: Learning communities at the community college,” Community College Review, vol. 30, no. 03, pp. 46-63, 2002.
[33] E. Nuutila, S. Törmä, P. Kinnunen, and L. Malmi, “Learning programming with the PBL method experiences on PBL cases and tutoring,” pp. 47-67, 2008.
[34] G. M. Novak, A. Gavrin, and C. Wolfgang, Just-in-Time Teaching: Blending Active Learning with Web Technology. Upper Saddle River, NJ, USA: Prentice Hall PTR, 1999.
[35] S. P. Linder, D. Abbott, and M. J. Fromberger, “An instructional scaffolding approach to teaching software design,” J. Comput. Small Coll., vol. 21, no. 6, pp. 238-250, 2006.
[36] E. W. G. Clua, “A game oriented approach for teaching computer science,” Anais do XXVIII Congresso da SBC, pp. 10-19, July 2008.
[37] F. Manne, “Competing in computing (poster session)”,in ITiCSE '00: Proceedings of the 5th annual SIGCSE/SIGCUE ITiCSE conference on Innovation and technology in computer science education, (New York, NY, USA), p. 190, ACM, 2000.
[38] T. Bailey and J. Forbes, “Just-in-time teaching for CS0,” SIGCSE Bull., vol. 37, no. 1, pp. 366-370, 2005.
[39] I. Greenberg, Processing: Creative Coding and Computational Art. Friendsof Ed., 1st ed., 2007..

Scheila Wesley Martins é bacharel em Ciência da Computação pela Universidade Federal de Mato Grosso (1998), especialista em Informática em Educação pela Universidade Federal de Lavras (2001) e mestre em Ciência da Computação pela Universidade Federal de Santa Catarina (2005). Atualmente é aluna de doutorado do Centro de Informática e Sistemas da Universidade de Coimbra (CISUC). Sua experiência docente em Computação tem ênfase em disciplinas de introdução à programação e algoritmos, e em informática em educação. Como pesquisadora, atua principalmente nos seguintes temas: ensino de programação, ambientes educacionais, pensamento crítico e teoria da atividade. Desde 1996 é membro da Sociedade Brasileira de Computação (SBC).

Antonio José Nunes Mendes é, desde 1996, Doutor em Engenharia Electrotécnica, especialidade de Informática, pela Universidade de Coimbra, tendo apresentado uma tese sobre Sistemas Autor de Programas Educativos. É Professor Auxiliar do Departamento de Engenharia Informática da Faculdade de Ciências e Tecnologia da Universidade de Coimbra, sendo responsável por disciplinas da área da Programação. É membro do Centro de Informática e Sistemas da Universidade de Coimbra (CISUC), coordenando as actividades do Educational Technology Lab, parte do Cognitive and Media System Group. Neste contexto desenvolve investigação na área das TIC no Ensino Superior, com particular destaque para a utilização de meios tecnológicos no suporte à aprendizagem de programação.
É autor ou co-autor de mais de 100 publicações, em revistas e conferências, a sua grande maioria de âmbito internacional.
Foi coordenador internacional da RIBIE - Rede Ibero Americana de Informática

Antonio Dias Figueiredo doutorou-se em "Computer Science" pela Universidade de Manchester em 1976 e obteve Agregação em Engenharia Informática pela Universidade de Coimbra em 1982. Entre 1984 e 2007 foi professor catedrático da Faculdade de Ciências e Tecnologia da Universidade de Coimbra, onde fundou, em 1994, o Departamento de Engenharia Informática. É professor catedrático aposentado do Departamento de Engenharia Informática da Universidade de Coimbra e investigador do Centro de Informática e Sistemas da Universidade de Coimbra (CISUC), onde se dedica à investigação em "Sistemas de Informação nas Organizações" e "Tecnologias da Informação e da Comunicação na Aprendizagem". Exerce também actividade de consultoria em regime independente..