Ensinando Construção de Software Aplicada a Sistemas de Informação do Mundo Real *

Márcio de Oliveira Barros[1] Renata Mendes de Araujo[1,2]

[1] Programa de Pós-Graduação em Informática, UNIRIO
[2] Núcleo de Pesquisa e Prática em Tecnologia, UNIRIO

marcio.barros@uniriotec.br, renata.araujo@uniriotec.br

Abstract. In this paper we describe a case study conducted along the software construction course at the Information Systems Post-Graduate program at the Federal University of the Rio de Janeiro State (UNIRIO). In this course, we challenged students a set of tasks associated to an actual problem being faced by real organizations. Students should develop a set of functional requirements but also to adapt the system to answer to events happening in the real world, notified by open communications and the media. Therefore, we created a learning environment closer to the industrial situation that students would find in their professional career.

Keywords: case study, information systems education, programming education.

Resumo. Neste artigo, apresentamos um estudo de caso realizado durante um curso de construção de software no Mestrado de Sistemas de Informação do PPGI/UNIRIO. Neste curso, apresentamos aos alunos um conjunto de trabalhos associados a um problema contemporâneo e que também estava sendo tratado por empresas reais. Os alunos deveriam desenvolver um conjunto de requisitos funcionais, mas também adaptar o sistema em desenvolvimento para responder a eventos acontecendo no mundo real e sendo notificados através de comunicados abertos ou na mídia. Com isto, criamos um ambiente de aprendizado mais próximo a uma situação industrial, que os alunos encontrarão em suas carreiras.

Palavras-chave: estudo de caso, ensino de sistemas de informação, ensino de programação.

1 Introdução

A construção de software [Mc Connel, 1993] se refere às atividades intermediárias do ciclo de vida de desenvolvimento de software, que compreendem o projeto detalhado, a codificação e os testes de unidade. Ela é precedida pelo levantamento de requisitos e pelo projeto arquitetural, sendo sucedida pela implantação e manutenção do sistema. Se analisada isoladamente, a construção de software compreende o conjunto de atividades de engenharia em que as habilidades técnicas (relacionadas ao computador) são aplicadas de forma mais intensa que as habilidades sociais e de comunicação (relacionadas aos seres humanos).

Lehman [Lehman, 1996] propõe um esquema para classificação de sistemas de software no qual estes são divididos em três classes: S-systems, P-systems e E-systems. A classe dos S-systems compreende o conjunto de sistemas que podem ser formalmente definidos por uma especificação e derivados a partir desta – isto é, o problema a ser resolvido por estes sistemas é completamente definido para todo conjunto de circunstâncias nas quais ele pode ser usado e a solução para este problema é conhecida [Pfleeger, 1998]. A classe dos P-systems inclui os sistemas nos quais o problema a ser resolvido não pode ser totalmente descrito ou a implementação completa de uma solução teórica para este problema não é viável com os recursos computacionais disponíveis no momento. Assim, a implementação de um P-system é limitada a uma aproximação da solução teórica, eventualmente aplicada a um subconjunto das situações em que o problema pode se apresentar no mundo real. Finalmente, os E-systems são sistemas nos quais o problema a ser resolvido sofre mudanças contínuas: enquanto nos S- e P-systems o mundo real é considerado estável e constante, um E-system incorpora a natureza mutável do mundo real e precisa ser continuamente atualizado para permanecer útil aos seus usuários.

Observamos que grande parte dos sistemas de informação reais – ou seja, sistemas utilizados para apoiar processos de negócio de uma indústria ou setor econômico real – podem ser caracterizados como E-systems, cujos requisitos podem mudar em função de eventos externos ao projeto de desenvolvimento do sistema e cuja utilidade depende da capacidade da equipe de desenvolvimento de atualizar continuamente os sistemas, mantendo-os funcionais ao longo do tempo, à medida que seus requisitos mudam.

Assim, o desenvolvimento de sistemas de informação reais é uma tarefa complexa, na qual as atividades de construção de software e levantamento de requisitos estão entremeadas: como os requisitos mudam ao longo do ciclo de desenvolvimento, não existe uma divisão clara entre as atividades de construção e levantamento de requisitos. Desta forma, o conhecimento técnico (como princípios de projeto ou proficiência em linguagem de programação) e as habilidades relacionadas a seres humanos (como capacidade de comunicação, pró-atividade, planejamento, entre outras) devem ser vivenciadas em conjunto, permitindo que problemas observados em um aspecto ampliem o efeito do outro, exigindo a tomada de decisão com informação incompleta e a cuidadosa avaliação das premissas a partir das quais o desenvolvimento será conduzido.

Por outro lado, a educação de profissionais que trabalharão com desenvolvimento de sistemas geralmente separa o aprendizado de aspectos técnicos do aprendizado de atividades relacionadas com seres humanos. Geralmente os responsáveis pelo ensino de aspectos técnicos apresentam exercícios em que os alunos recebem especificações

precisas (ou seja, vivenciam o desenvolvimento de S-systems), de forma que eles possam aplicar o conhecimento técnico adquirido independente de qualquer mal entendimento sobre os requisitos. Os responsáveis pelo ensino de aspectos humanos, por outro lado, comumente apresentam estas atividades sem que um produto real seja construído a partir do resultado de sua aplicação, ou seja, sem que haja a necessidade de negociar a introdução de novos requisitos ou mudanças no plano de projeto entre os clientes e os desenvolvedores. Muitos educadores envolvidos com cursos técnicos reduzem o valor das atividades relacionadas ao ser humano ("papel aceita tudo! papel não compila!"), enquanto o segundo grupo coloca os aspectos técnicos em segundo plano ("não faz sentido escrever um sistema que ninguém quer utilizar"). Desta forma, desenvolvedores inexperientes são geralmente incapazes de lidar com aspectos técnicos e aspectos humanos ao mesmo tempo.

Acreditamos que os envolvidos no ensino de Engenharia de Software, em particular de Sistemas de Informação, deveriam apresentar aos alunos trabalhos e estudos de caso mais próximos a sistemas do mundo real (ou seja, E-systems ao invés de S-systems). Neste artigo apresentamos um estudo de caso executado em uma disciplina de Construção de Software do curso de Mestrado do Programa de Pós-Graduação em Informática da Universidade Federal do Estado do Rio de Janeiro (UNIRIO). Ao longo do curso, os alunos foram avaliados através de trabalhos relacionados com um problema que foi alvo da atenção de diversas software-houses brasileiras no mesmo período: a implementação da plataforma regulamentar Basiléia II em instituições financeiras nacionais. Os trabalhos exigiram que os alunos desenvolvessem um sistema para dar suporte a um conjunto de funções relacionadas à plataforma Basiléia II, visando auxiliar o trabalho da agência regulamentadora de instituições financeiras – no caso, o Banco Central do Brasil. A necessidade de adaptação do sistema para mantê-lo útil à medida que o Banco Central apresentava mudanças na plataforma regulamentar foi característica importante e considerada parte do trabalho, de modo que os alunos foram diretamente influenciados por eventos do mundo real. Assim, os alunos vivenciaram a construção de um E-system, visto que a especificação do trabalho mudava em função de eventos fora do seu controle e a estrutura interna do sistema deveria ser alterada para apoiar estas mudanças.

Este artigo está organizado em cinco seções. A primeira seção contém esta introdução. A seção 2 descreve a plataforma regulamentar Basiléia II, que é a base do estudo de caso. A seção 3 descreve os objetivos e organização do curso PPGI017, cujos alunos participaram no estudo de caso relatado. A seção 4 apresenta a cronologia de eventos relacionados à plataforma Basiléia II e ao curso PPGI017, ressaltando como os eventos disparados pelo Banco Central do Brasil influenciaram os trabalhos propostos ao longo do curso. A seção 5 resume as lições aprendidas com o estudo de caso e apresenta algumas conclusões que pudemos obter a partir de sua análise.

2 A Plataforma Regular Basiléia II

Em junho de 2006, o Comitê da Basiléia para Supervisão Bancária publicou a versão final da plataforma regulamentar Basiléia II. Essa plataforma foi desenvolvida como resultado de sete anos de trabalho para garantir convergência internacional no tocante a regulamentações governamentais sobre exigências de capital de instituições financeiras ativas e com negócios realizados no exterior. A plataforma Basiléia tem como objetivo criar um padrão internacional a ser seguido por agências regulamentadoras de diversos países quando elas definirem regras relacionadas a quanto capital as instituições

financeiras devem depositar para assegurar as posições em instrumentos financeiros de risco de mercado, de crédito e operacional. Com isto, as agências regulamentadoras poderiam evitar que as instituições concretizem estratégias de investimento com risco acima de sua capacidade de restabelecimento.

A plataforma Basiléia II descreve um conjunto de medidas e um padrão mínimo de exigência de capital que as autoridades nacionais deveriam implementar sob a forma de leis (domésticas) e procedimentos de comunicação de estratégias realizadas e posições de investimento [Comitê da Basiléia para Supervisão Bancária, 2008]. A plataforma procura aprimorar as regras anteriores, alinhando a exigência de capital com os riscos incorridos pelas instituições financeiras (tais como bancos, gerenciadores de fundos de pensão, empresas de seguro e resseguro, entre outras). Adicionalmente, a plataforma pretende promover procedimentos de supervisão que não se baseiem apenas em informações passadas, mas nas possibilidades de perda potencial dos investimentos de uma instituição em diferentes cenários sobre o futuro, encorajando as instituições financeiras a identificarem claramente os riscos que estão correndo e aprimorarem seus mecanismos de controle sobre estes riscos. Como resultado deste esforço, a plataforma Basiléia II é flexível e capaz de se adaptar às mudanças freqüentes nos procedimentos utilizados pelo mercado para medir e controlar riscos.

Por outro lado, a plataforma Basiléia II não pretende ser um acordo internacional no sentido jurídico: não há assinaturas ou contratos entre os países participantes. As agências regulamentadoras que tiverem interesse em adotar a plataforma podem endossar o padrão internacional e implementá-lo, criando leis que sigam seus princípios.

O Banco Central do Brasil, agência regulamentadora do sistema financeiro nacional, endossou a plataforma Basiléia desde sua primeira versão (a plataforma Basiléia I foi adotada pelo Banco Central do Brasil em 1994). No entanto, em 2004 o Banco Central adotou uma política mais conservadora em relação à segunda versão da plataforma Basiléia, criando regras que exigiam que "a maior parte das instituições financeiras adotem uma versão simplificada do padrão internacional, aprimorando sua implementação da primeira versão da plataforma e incluindo elementos que, utilizando instrumentos financeiros próprios para mitigar riscos de crédito, permitissem um melhor alinhamento entre o capital exigido e a exposição a risco" [Banco Central do Brasil, 2004]. Posteriormente, as instituições financeiras deveriam preparar seus mecanismos de controle interno – e os sistemas que oferecem suporte a estes mecanismos – de modo a serem capazes de implementar novas regras relacionadas com a plataforma Basiléia II. No período entre 2007 e 2008, a transição de uma implementação simplificada para uma implementação mais ampla da plataforma foi acelerada e o Banco Central lançou um conjunto de regras exigindo que as instituições financeiras informassem suas exposições a risco de mercado em um relatório mensal (o DRM, ou Demonstrativo de Risco de Mercado).

O DRM e os eventos que ocorreram entre janeiro e junho de 2008 compõem o contexto para o estudo de caso relatado nas próximas seções. No estudo de caso, um grupo de alunos projetou e implementou um sistema de informação para processar os valores adquiridos de um conjunto de relatórios DRM (valores aleatórios formatados como relatórios DRM), sendo influenciados por mudanças sofridas por este relatório no mundo real.

3 O Curso sobre Construção de Software

A disciplina PPGI017 – Técnicas Avançadas de Construção de Software – tem como objetivo ensinar o desenvolvimento de sistemas de alta qualidade, aplicando padrões reconhecidos nas atividades de projeto detalhado e codificação de software. Os alunos aprendem a importância de manter uma boa organização e identação no código do sistema, princípios de projeto de software, características observáveis em bons e maus projetos, tópicos relacionados à introdução de mudanças em um sistema, princípios de projeto orientado a objetos, caracterização de classes, projeto de pacotes de classes e padrões de projeto. Inclui também temas práticos relacionados à construção de software, como gerência de configuração, registro de defeitos e testes de unidade.

PPGI017 é uma disciplina opcional no currículo do curso de Mestrado em Sistemas de Informação do Programa de Pós-Graduação em Informática da Universidade Federal do Estado do Rio de Janeiro (UNIRIO). A edição de 2008 da disciplina ocorreu no primeiro semestre deste ano e foi atendida por apenas seis alunos (um número pequeno, porém esperado, de alunos de pós-graduação cujos interesses incluem projeto e codificação de software). Os instrumentos de avaliação do curso incluem uma prova escrita (realizada no meio do curso e com peso unitário) e quatro trabalhos de implementação (com peso dois), cujas notas são somadas para compor a média da disciplina com a nota da prova. Essa estratégia permite determinar se os alunos desenvolveram conhecimento teórico nos assuntos tratados pela disciplina, sem perder de vista a aplicação prática destes conceitos.

Uma das diretrizes do curso de Sistemas de Informação é que seja dada ênfase ao lado prático, ou seja, à aplicação do conhecimento adquirido pelos alunos. Neste sentido, a aplicação prática dos conceitos ensinados na disciplina PPGI017 sempre foi um forte direcionador na seleção dos projetos de curso a serem desenvolvidos pelos alunos. No entanto, é difícil identificar oportunidades de projetos que sejam suficientemente grandes para ser representativos do mundo real, mas pequenos o suficiente para que seu desenvolvimento seja viável nos quatro meses que compõem o período do curso. Além disso, seria desejável apresentar trabalhos com características próximas às encontradas em projetos do mundo real – como requisitos imprecisos, mudanças tardias nos requisitos, competição entre provedores de solução e datas inflexíveis para a entrega dos subprodutos –, ao invés de projetos com requisitos completos e estáveis. Seria desejável submeter os alunos a ambientes de desenvolvimento "não tão controlados", de modo que eles possam vivenciar problemas no ambiente de rede, falhas de comunicação, mudanças não agendadas na plataforma de trabalho, atualizações inesperadas de compiladores e ferramentas de desenvolvimento, entre outras. Afinal, estes são cenários que podem acontecer na indústria e os alunos devem estar preparados para enfrentá-los de forma adequada.

Na preparação da oferta da disciplina em 2008, buscamos um problema real que ainda estivesse em aberto e que nos desse a oportunidade de explorar a dinâmica entre a construção do sistema e o surgimento de novos requisitos, sem que os alunos pudessem justificar as imperfeições presentes nos requisitos iniciais dos trabalhos. A implementação da plataforma Basiléia II se apresentou como um exemplo perfeito para pano de fundo dos trabalhos do curso. Propusemos que os alunos desenvolvessem uma ferramenta para ajudar o Banco Central do Brasil a analisar os relatórios DRM enviados pelas diversas instituições financeiras sob sua supervisão. Cada relatório contém milhares de valores financeiros e uma ferramenta computacional que suportasse as análises destes números poderia auxiliar o Banco Central em sua tarefa

de supervisor. Para efeito de avaliação dos alunos na disciplina, o desenvolvimento da ferramenta poderia ser dividido em três ou quatro trabalhos de curso distintos.

A especificação parcial da ferramenta proposta, apresentada como descrição do primeiro trabalho, introduziu os alunos ao domínio das finanças – apenas um aluno tinha experiência anterior neste domínio – e enumerou as funções que a ferramenta deveria oferecer aos seus usuários. Nesta especificação, foram indicados links para um conjunto de documentos publicados pelo próprio Banco Central do Brasil, incluindo layouts e especificações de fórmulas para calcular os valores apresentados no relatório DRM. Estes documentos foram rigorosamente os mesmos utilizados por software-houses reais desenvolvendo sistemas para atender à implementação da plataforma Basiléia II. Os alunos puderam enviar questões e dúvidas diretamente aos responsáveis pelo projeto no Banco Central, da mesma forma desenvolvedores e líderes de projeto poderiam fazer na indústria. Atrasos decorrentes da falta de respostas puderam ser vivenciados, tal como no mundo real. Os alunos puderam ler sobre o assunto do seu trabalho nos jornais e sentir como se estivessem trabalhando em software-houses reais, desenvolvendo um software para instituições financeiras ou para o próprio Banco Central – e levaram isto muito a sério no desenvolvimento dos trabalhos. A motivação foi fascinante, tal como nunca havíamos observado em outras disciplinas de programação!

Além disso, um esquema de avaliação competitivo foi colocado em prática. Os alunos foram divididos em três grupos (A, B e C), cada qual composto por dois alunos. Cada grupo deveria apresentar resultados para cada parte do trabalho. Estes resultados foram avaliados por sua funcionalidade e também por suas propriedades internas, como legibilidade do código, simplicidade, organização do código e do projeto detalhado. A melhor solução receberia uma nota que poderia atingir 10 (dez); a segunda melhor solução receberia uma nota que poderia atingir apenas 8 (oito), enquanto a terceira estaria limitada a um máximo de 6 (seis). Empates seriam permitidos e atrasos seriam punidos com a perda de um ponto por dia útil de atraso após a data programada para a entrega. Estas regras reforçaram o senso de competição que se observa no mercado e este senso foi bem recebido pelos grupos.

4 O Relatório DRM – Cronologia dos Eventos

A seguir apresentamos os eventos que compreenderam a definição e os requisitos regulatórios do relatório DRM e como influenciaram a disciplina PPGI017, permitindo aos alunos vivenciarem o desenvolvimento de um sistema de informação sujeito a mudanças do mundo real.

- Jan/2008 – a Associação Brasileira de Instituições Bancárias, associação que apóia o Banco Central na definição dos instrumentos regulatórios relacionados ao relatório DRM, lança uma nota pública sobre o relatório, estabelecendo a maioria das fórmulas de cálculo de exposição ao risco que devem ser seguidas e o formato no qual as organizações financeiras devem entregá-lo ao Banco Central;
- Mar/2008 – A disciplina é iniciada e os alunos são apresentados ao estudo de caso da plataforma Basiléia II. Foi realizada uma apresentação sobre como a informação de exposição de risco deve ser fornecida ao Banco Central e os alunos receberam a nota pública da Associação Brasileira de Instituições Bancárias e os formatos propostos para o relatório;

- Mar/2008 (uma semana depois) – a primeira tarefa é apresentada aos alunos. Nela, eles devem desenvolver (projetar e codificar) um conjunto de classes que modelem a informação sobre exposições de riscos a serem apresentadas ao relatório DRM;
- Abr/2008 (duas semanas depois) – os alunos entregam os resultados da primeira tarefa e são avaliados pelo professor da disciplina. O grupo A apresenta a melhor entrega, apesar da organização interna do código ainda apresentar possibilidades de melhoria. Recebe a nota 9 (nove), enquanto que os grupos B e C se mantêm em segunda posição, recebendo nota 7 (sete). Um resumo da avaliação de cada grupo foi disponibilizado para todos os alunos, bem como as implementações. Isto permitiu que aprendessem com os erros alheios e se preparassem melhor para a segunda tarefa. A versão entregue pelo grupo A foi aperfeiçoada para se tornar a implementação de referência, a ser utilizada por todos nas tarefas seguintes;
- Abr/2008 (uma semana depois) – a segunda tarefa é apresentada aos alunos, na qual deveriam desenvolver classes para leitura das informações de exposição de risco de um relatório DRM para a estrutura de classes construída na tarefa anterior. Exemplos de relatórios DRM, gerados utilizando planilhas, foram entregues como base de testes para a implementação (acrescidos de um erro de formatação conhecido em uma das entradas);
- Abr/2008 (uma semana depois) – dois grupos detectaram o erro de formatação e comunicaram ao professor (A e B). Foi solicitado que corrigissem o erro manualmente no relatório e informassem ao terceiro grupo. O grupo B também identificou um erro não esperado (um erro real, não intencionalmente introduzido pelo professor da disciplina), tratado da mesma maneira. Além disso, algumas regras de negócio que não haviam sido claramente definidas na especificação (tais como o tamanho e o conteúdo de alguns campos do relatório), são estabelecidas pelo professor;
- Mai/2008 (uma semana depois) – dois grupos entregam as tarefas no prazo (B e C), enquanto que o terceiro (A) entrega com dois dias de atraso. Os três grupos se posicionam na primeria posição, embora todos pudessem ter aprimorado o projeto das classes utilizadas para implementar a funcionalidade requerida. Cada grupo recebe a nota 8 (oito) e o grupo A sofre uma penalidade de 2 pontos pela entrega tardia (terminando com nota 6). Assim como na tarefa anterior, o resumo de cada avaliação e as versões de implementação entregues são publicados e uma implementação de referência é desenvolvida para as próximas tarefas. Desta vez, no entanto, a implementação de referência é codificada "do zero" pelo professor;
- Mai/2008 (uma semana depois) – a terceira tarefa é apresentada aos alunos, pedindo que desenvolvam uma ferramenta que ofereça uma visualização gráfica dos valores de exposição a risco de um grupo de relatórios recebidos em um mesmo período (de um mesmo mês e ano). A representação gráfica deve ser customizável, permitindo aos usuários selecionar partes do relatório DRM que devem ser apresentados na tela. A ferramenta foi descrita de forma pouco detalhada e a discussão durante a apresentação da tarefa contribuiu para "aparar as arestas" de sua especificação;
- Mai/2008 (mesma semana) – O Banco Central publica uma norma regulatória descrevendo a versão final do relatório DRM. Essa versão tem diversos aspectos distintos do proposto em Janeiro. Entre estes aspectos, (a) o relatório deve incluir uma nova seção, agregando a exposição a risco de algumas de suas seções; (b) o relatório, que deveria ser publicado em formato texto, deve agora ser publicado em formato XML; (c) um vértice extra foi incluído em cada curva de mercado na qual a

instituição financeira pode apresentar exposição a risco; e (d) fatores de risco atômicos (como preços de ações, commodities e taxas de conversão de moedas) devem ser tratados como curvas com um único vértice;

- Jun/2008 (três semanas depois) – a terceira tarefa é entregue aos alunos. Eles esperam uma quarta tarefa que seja uma simples extensão da terceira, muito provavelmente uma nova estratégia para visualizar a informação que eles já sabem ler e manipular através das estruturas de classes criadas nas tarefas 1 e 2. Entretanto, eles recebem o instrumento regulatório oficial liberado em Maio, em seu formato original, e que deverão atualizar a ferramenta de forma a que ela continue útil. Eles têm duas semanas para entregar a nova versão da ferramenta;
- Jun/2008 (duas semanas depois): em final de período, os três grupos se esforçam para entregar a última tarefa do trabalho. O grupo B recebe a melhor nota, enquanto os grupos A e C ficam praticamente empatados na avaliação. Como resultado do curso, temos uma ferramenta completa e que pode ser realmente interessante para a agência regulamentadora. No entanto, o principal resultado não é tangível e está representado na experiência vivida pelos alunos (e pelo professor) na condução da disciplina.

5 Lições Aprendidas e Conclusões

A expectativa acerca da disciplina era que os alunos veriam o valor das discussões teóricas sobre organização do código e princípios de projetos no desenvolvimento do projeto proposto. Na quarta tarefa, quando precisariam rever o código escrito meses atrás, deveriam apreciar os comentários escritos na época e o esforço extra para tornar o código legível. Esperávamos que compreendessem o valor de prever mudanças futuras e dividir o sistema em componentes que limitem os efeitos colaterais das mudanças sofridas pelo software. Nestes aspectos, como pudemos observar em conversas com os alunos ao final do curso, não nos frustramos. No entanto, conduzir uma experiência de projeto como a descrita neste artigo não é uma tarefa fácil. A seguir, são discutidas as dificuldades encontradas:

- É difícil encontrar problemas reais, como a implementação da plataforma Basiléia II. Problemas reais são mais complexos do que o razoável para uma disciplina de curta duração. Por outro lado, uma situação similar pode ser simulada ou o problema complexo pode ser particionado no tamanho certo para as necessidades e limitações da disciplina;
- Uma vez que o mundo real está sendo tratado, o cenário que influencia as tarefas propostas aos alunos não pode ser totalmente controlado pelo professor: não há garantia de que questões relevantes possam aparecer. Além disso, eventos podem ocorrer em uma direção diferente do que a esperada. Esta última situação é mais fácil de ser tratada, com o professor atuando como um filtro para as novidades vindas do ambiente externo. A primeira situação não pode ser prevista ou controlada e, portanto, o professor pode precisar simular a ocorrência de eventos relevantes. Isto pode afetar negativamente a motivação, uma vez que a proximidade com o mundo real, e com eventos que realmente aconteceram, gera motivação nos alunos;
- É bastante útil trabalhar com entregas parciais, a serem analisadas pelo professor e detalhadas em implementações de referência para as tarefas na sequência. A avaliação de cada fase permite aos alunos aprender a partir de seus erros e obter

melhores resultados nas entregas subseqüentes. A avaliação compartilhada permite que os alunos aprendam com os erros cometidos pelos demais. Finalmente, a implementação de referência contribui para manter a competição, uma vez que os alunos partem do mesmo ponto a cada nova entrega;

- O esforço de organizar o projeto de curso é muito grande. Deve ser contabilizado tempo para: selecionar um projeto apropriado, avaliar as entregas enviadas pelos alunos (uma vez que elas serão a base para tarefas futuras, elas precisam ser corrigidas pelo professor o mais rápido possível), aprimorar a melhor solução em uma implementação de referência, acompanhar e filtrar os eventos vindos do ambiente externo. Estas atividades se somam à preparação de aulas, preparação e correção de provas (no caso desta disciplina, uma única prova), preparação de exercícios, etc. O professor deve estar preparado para dispender mais tempo durante a execução da disciplina e para "colocar as mãos na massa" para preparar as implementações de referência, mesmo em um cenário em que nenhum dos grupos pôde entregar uma solução para uma dada tarefa;
- Apesar do custo de preparar e executar a disciplina, a reutilização do material elaborado e das tarefas sugeridas para uma nova oferta no futuro é restrita. Uma vez que o projeto real é temporário (deveria ser, pelo menos), os eventos e notícias relacionados a ele supostamente terminam dentro de um período de tempo. Então, após este período, o projeto (junto com seu material relacionado) será parte do passado. Desta forma, perderá algumas de suas relações com o mundo real, apesar de permanecer um problema concreto a ser tratado pela indústria: será uma tarefa melhor do que um projeto abstrato e genérico.

Agradecimentos: agradecemos aos alunos que participaram deste estudo de caso – Daniel Serrano, Fernando Netto, João Gonçalves, Leonardo Jobim, Martin Pereira e Wallace Ugulino.

Referências

BANCO CENTRAL DO BRASIL. **Comunicado nº 12.746 .** Disponível através de busca por número de documento em: https://www3.bcb.gov.br/normativo/pesquisar.paint?method=pesquisar Acesso em maio de 2008.

COMITÊ DA BASILÉIA PARA SUPERVISÃO BANCÁRIA. Disponível em: http://www.bis.org Acesso em maio de 2008.

LEHMAN, M. Laws of Software Evolution Revisited. Lecture Notes in Computer-Science, Springer, Berlin/Heidelberg, v. 1149, p. 108 – 124, 1996.

Mc CONNEL, STEVEN. **Code Complete: A Practical Handbook of Software Construction**. Microsoft Press, Redmond, WA, 1993. 960 p.

PFLEEGER, Shari L. **Software Engineering: Theory and Practice**. Prentice-Hall Inc. Upper Saddle River, NJ, 1998. 659 p.