



UM FRAMEWORK PARA O ENSINO DE DESENVOLVIMENTO DISTRIBUIDO DE SOFTWARE EM CURSOS DE GRADUAÇÃO

Alexandre L'Erario – alerario@utfpr.edu.br

Universidade Tecnológica Federal do Paraná, Campus Cornélio Procópio
Avenida Alberto Carazzai, 1640
86300-000 – Cornélio Procópio - PR

Mariana Orlandini Ferreira – mari-orlandini@hotmail.com

Universidade Tecnológica Federal do Paraná, Campus Cornélio Procópio
Avenida Alberto Carazzai, 1640
86300-000 – Cornélio Procópio - PR

Alessandro Silveira Duarte – duarte@utfpr.edu.br

Universidade Tecnológica Federal do Paraná, Campus Cornélio Procópio
Avenida Alberto Carazzai, 1640
86300-000 – Cornélio Procópio - PR

José Augusto Fabri – fabri@utfpr.edu.br

Universidade Tecnológica Federal do Paraná, Campus Cornélio Procópio
Avenida Alberto Carazzai, 1640
86300-000 – Cornélio Procópio – PR

Henrique Dalavale Fabretti – henrique.fabretti@gmail.com

Universidade Tecnológica Federal do Paraná, Campus Cornélio Procópio
Avenida Alberto Carazzai, 1640
86300-000 – Cornélio Procópio – PR

Resumo: *Este artigo apresenta uma proposta de um framework para ensinar desenvolvimento distribuído de software (DDS) em cursos de graduação. O escopo deste modelo centra-se em cursos na área de computação. Tal framework envolve dois aspectos; o primeiro deles é a institucionalização das teorias relacionadas ao DDS; e o segundo aspecto trata-se de como um projeto de desenvolvimento distribuído de software pode ser aplicado em cursos de graduação. Este segundo tem o objetivo de promover a prática de DDS entre os alunos. Um primeiro experimento encontra-se em execução na Universidade Tecnológica Federal do Paraná – Campus Cornélio Procópio. Os resultados preliminares são utilizados para a consolidação deste framework.*

Palavras-chave: *Ensino de Desenvolvimento Distribuído de Software, Ensino de Engenharia de Software.*

1. INTRODUÇÃO

O desenvolvimento distribuído de software é caracterizado pelo desenvolvimento de um mesmo produto, por equipes remotamente distantes (L'ERARIO, 2012). Esta modalidade de desenvolvimento torna-se cada vez mais crescente. São vários motivos que conduzem as

Realização:



Organização:





organizações a se engajarem nesta modalidade. Dentre eles destacam-se: o custo de produção, a proximidade com clientes, a disponibilidade de mão de obra, entre outros citados por L'Erario (2009).

Porém, com o crescimento do DDS, surgiram novos desafios para o desenvolvimento até então inexistentes ou amenos. Novas particularidades passaram a serem consideradas como fatores de risco: diferenças culturais, fuso horário, idioma entre outros, uma vez que para enfrentar esses novos obstáculos, os profissionais da área de produção de software precisam adequar-se a esta modalidade de desenvolvimento.

A fim de viabilizar a formação dos profissionais na área de DDS, propõe-se o ensino deste em cursos de graduações, com o objetivo de aproximar os alunos a uma experiência de desenvolvimento. As habilidades esperadas na prática do DDS podem contribuir até mesmo para o desenvolvimento local, já que o profissional agrega noções de trabalho em equipes, comunicação, motivação, planejamento e exploração de novas ferramentas.

O objetivo deste trabalho é apresentar um framework para o ensino de práticas de desenvolvimento distribuído de software em cursos de graduação para garantir que os alunos tenham um contato mais real com DDS, ainda dentro da universidade. A motivação deste, está alicerçada na demanda do mercado por profissionais que conheçam tais práticas. O desafio almejado neste trabalho aborda a institucionalização dos problemas relacionados ao DDS nos cursos de graduação e também o desenvolvimento de um projeto distribuído de software pelos alunos. Tais medidas

2. DESENVOLVIMENTO DISTRIBUÍDO DE SOFTWARE

DDS é definido pela colaboração e cooperação entre departamentos de organizações e pela criação de grupos de desenvolvedores que trabalham em conjunto, porém distantes fisicamente (PRIKLADNICKI, 2006).

De acordo com Marquardt e Horvath (2001), DDS é composto por equipes globais, que são grupos de pessoas de diferentes países que trabalham em conjunto no desenvolvimento de um projeto. Karolak (1998) aborda o ambiente distribuído como o resultado da união de organizações virtuais, que se referem a entidades que desenvolvem partes de um projeto em locais dispersos, porém encarando-o como um projeto local.

Prikladnicki (2003) categoriza o DDS de acordo com a distância geográfica entre os sites (cada unidade de trabalho), classificando-o em níveis de dispersão nacional, continental e global, com equipes localizadas em um mesmo país, em países diferentes situados em um único continente e com alocações em mais de um país, respectivamente. Os dois últimos casos caracterizam desenvolvimento global de software (GSD).

Projetos centralizados possibilitam um gerenciamento mais focado, baseado em observação e comunicação informal. Já a produção distribuída pode instigar nos grupos maior vontade de inovação, mais motivação por entrar em contato com pessoas e culturas diferentes, além de possibilitar estratégias diferentes para a resolução de um mesmo problema, advindas de contextos distintos.

Outro fator que deve ser levado em conta é a comunicação constante, para que ela aconteça de forma eficiente, o ambiente distribuído deve conter uma estrutura estável, onde o repasse de dados seja confiável e de desempenho ágil. Se as equipes estiverem fortemente interagidas entre si e com a gerencia, trocando informações e artefatos de forma rápida e eficaz, problemas tais como integração dos elementos e alocação do trabalho serão amenizados, favorecendo o andamento do projeto.



Consideradas as dificuldades envolvidas na aplicação de DDS, uma das técnicas mais eficazes para evitar problemas do ambiente distribuído é a gerência flexível e efetiva do projeto. Gerentes devem estruturar minuciosamente o escopo e a arquitetura do projeto, para que ele possa ser dividido entre os grupos de desenvolvimento; além de oferecer suporte às equipes e acompanhá-las com uma gerência de riscos constante. Os principais aspectos enfocados por Carmel (1999) são a gerência de conflitos, utilização de métricas, formas de reconhecimento e bonificação, e escolha de um gerente com perfil para atuar em projetos distribuídos.

Para o desenvolvimento distribuído de software existem tecnologias que devem ser exploradas ao ensiná-lo, estas tecnologias podem ser usadas para facilitar a gestão e o desenvolvimento do projeto. Softwares livres para controle de versões, bibliotecas para desenvolvimento integrado, ferramentas síncronas e assíncronas de comunicação, ferramentas CASE (Computer-Aided Software Engineering), entre outros, podem aperfeiçoar e auxiliar o DDS.

Para Patil (2011), desenvolvimento distribuído de software agrega um conjunto de percepções relacionadas a localidade física de cada unidade de desenvolvimento, tal como a cultura, língua entre outros. Em sua pesquisa, Patil (2011) identifica um conjunto de ferramentas de comunicação, tais como: e-mail, chat, calendários compartilhados e etc. Além destes fatores, Guzman (2010), propõe um framework de melhoria de processo de desenvolvimento distribuído.

3. O ENSINO DE DDS EM CURSOS DE GRADUAÇÃO

O desenvolvimento global de software tem tomado grandes proporções, conforme informam L'Erario (2007) e Carmel e Agarwal (2001), por esta razão, desperta interesse por parte de instituições de ensino em treinar seus discentes em tal modalidade de desenvolvimento, como por exemplo, apresentado por L'Erario (2007). Na literatura podem ser encontrados vários trabalhos que citam experiências de ensino de DDS, como Carrington (2008), por exemplo. O foco destas experiências tem sido os desafios em torno desta aplicação e a aproximação dos alunos do ambiente real de desenvolvimento em empresas, como experimentado por Faulk, Young, Weiss e Yu (2011).

Estas pesquisas relatam que: o desenvolvimento distribuído pode dar aos alunos uma experiência prática em gestão de situações em um cenário distribuído e que as competências adquiridas neste ambiente são diferenciais mesmo no desenvolvimento local. De acordo com Prikladnicki (2011), as habilidades desenvolvidas e soluções encontradas para as barreiras no ambiente distribuído, tornam-se grandes diferenciais em todos os cenários de desenvolvimento de software.

A abrangência e a natureza da colaboração entre unidades de conhecimento e trabalho mudaram significativamente na última década (Patil, Kobsa, John e Seligmann, 2011). Essas mudanças prometem uma série de benefícios potenciais para as organizações, como alocação de profissionais qualificados independentemente de onde eles residem, aproveitando-se dos baixos custos de trabalho, economizando tempo, dinheiro em viagens de funcionários e encurtando os períodos de desenvolvimento de produto, espalhando assim o trabalho em diferentes fusos horários (Olson e Olson, 2000). Porém, essas vantagens também estão atreladas a desafios sociais, como, diferenças culturais, problemas de comunicação; coordenação, pouco conhecimento acerca do trabalho desenvolvido por outros membros da equipe.



Por estas razões, empresas que produzem software passaram a se preocupar com estas questões, valorizando profissionais que possuem experiência nesse tipo de desenvolvimento. Porém, conforme Crnković, Land, Bosnić, Cavrak e Žagar (2010), as empresas queixam-se da fundamentação pouco prática dos profissionais que chegam até elas, argumentando que possuem alto nível de conhecimento teórico, mas, pouco conhecimento prático sobre o ambiente distribuído real. Por esta razão, o DDS desperta interesse por parte de instituições de ensino, em treinar seus discentes em tal modalidade.

Um dos papéis dos cursos relacionados à engenharia de software é preparar os estudantes para o desenvolvimento de software, como ele é feito em empresas. A exposição dos alunos aos problemas de campo possibilita que eles obtenham experiência prática e se sobressaiam na hora de lidar com estes problemas no mercado de trabalho, como, abordado por Nordio, Ghezzi, Meyer, Di Nitto, Tamburrelli, Tschannen, Aguirre, e Kulkarni (2010). O DDS é uma realidade, e com o aumento desta área, é necessário preparar profissionais para desenvolver software distribuído.

Embora o ensino de engenharia de software enfatize a prática, como mostrado por Shaw (2000), os cursos de DDS ainda são raros. De acordo com Bosnić, Čavrak, Orlić, Žagar e Crnković (2010), a grande dificuldade é estabelecer ambientes distribuídos comuns a mais de um site, onde os principais obstáculos são preparar um curso comum, cumprir regras administrativas locais de cada lado e fornecer apoio técnico eficaz durante a execução do curso.

A fim de viabilizar a formação dos profissionais com familiaridade na área de DDS, propõe-se o ensino deste método em cursos de graduação, ajudando os alunos a aproximarem-se de uma experiência de trabalho e adquirirem prática em um âmbito bastante promissor. O ensino do desenvolvimento distribuído de software, em cursos de graduação tem sua importância alicerçada no aumento de organizações que desenvolvem software de maneira distribuída. O aluno egresso que tenha familiaridade com tal modalidade tende a enquadrar-se com mais agilidade no mercado de trabalho. Tal ensino pode ser constatado nos trabalhos de Carrington (2008) e L'Erario (2007).

A principal questão desta pesquisa é como ensinar os alunos de graduação a desenvolverem software de maneira distribuída. Tal questão conduz a uma série de variáveis, que precisam ser identificadas. Primeiramente, um conjunto de conceitos, identificados neste trabalho são necessários para desenvolver software de maneira distribuída. Tais conceitos foram levantados a partir de L'Erario (2009) e são indicados na tabela 1.

Tabela 1 - Conceitos de DDS

Conceito	Descrição
Coordenação	Mecanismo que define as propriedades dos ambientes distribuídos, englobando ajuste mútuo, padronização e supervisão direta.
Granularidade do repasse	Indica o nível em que as informações são trocadas entre os sites (unidades de produção).
Difusão do processo	de compatibilidade dos processos entre os sites
Quantidade de sites	É cada unidade independente de produção de software distante das demais (L'ERARIO, 2007), e para DDS, o número mínimo de sites é dois
Mecanismos de comunicação	Uso de ferramentas síncronas e assíncronas e a integração destes com softwares para desenvolvimento de código



Tais conceitos, descritos na tabela 1, devem ser difundidos na graduação. Neste projeto optou-se de duas maneiras: a primeira dela é incluir tais conceitos nas disciplinas relacionadas à área de engenharia de software. Neste caso, o aluno teve contato como o conceito e não o aplicou efetivamente; a segunda maneira foi criar uma estrutura de projeto multidisciplinar, ou seja, um mesmo projeto de software a ser desenvolvido por alunos de várias disciplinas/cursos com o objetivo de aplicar tais conceitos. A relação entre estas duas variáveis é expressa na figura 1.

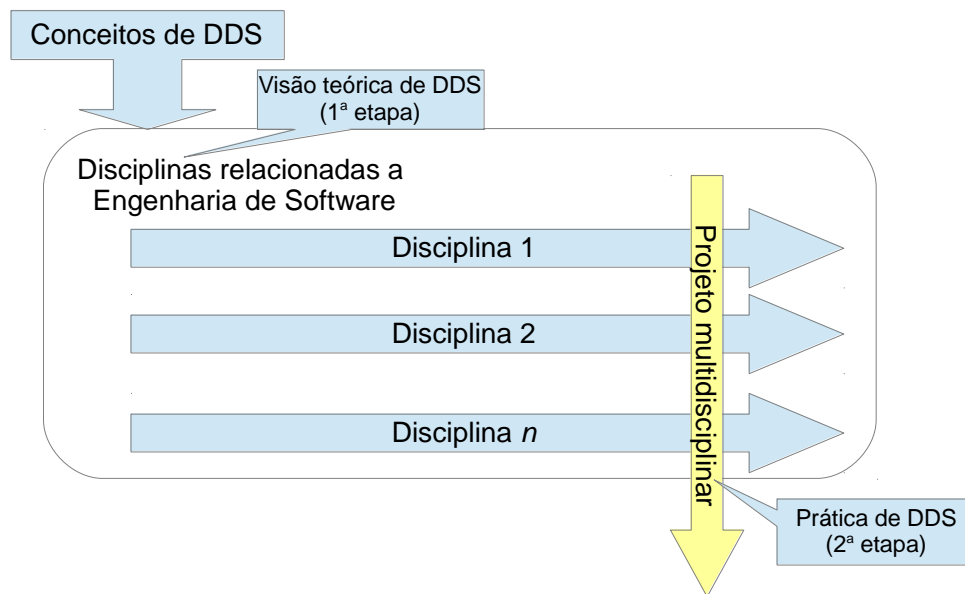


Figura 1 - Como ensinar conceitos de DDS na graduação

A figura 1 representa como o DDS pode ser ensinado em cursos de graduação. Em uma primeira etapa, os conceitos de DDS são distribuídos pelas disciplinas relacionadas a engenharia de software. A próxima etapa é executar um projeto de software multidisciplinar. A primeira etapa contém as seguintes atividades:

1. Identificar as disciplinas chaves: analisar as grades curriculares das instituições/cursos e selecionar as disciplinas que participarão do projeto de DDS. Tornar compatíveis os conteúdos das disciplinas e os projetos, ajustando quais conceitos serão ensinados e em quais momentos. Este procedimento é explicado por L'Erario (2008)
2. Difundir os conceitos de coordenação, granularidade do repasse, difusão do processo e comunicação (conceitos descritos na tabela 1) no conteúdo programático das disciplinas.

Na primeira etapa, há uma grande independência entre as disciplinas. Trata-se de apresentar o conteúdo teórico adequado para o discente. A segunda etapa, que é o desenvolvimento de um projeto de software multidisciplinar gera uma grande dependência entre disciplinas. Nesta pesquisa foram identificados três possíveis cenários de projetos de DDS, que podem ser executados por cursos de graduação. Tais projetos visam incorporar



no(s) curso(s) os conceitos de DDS e também servir como mecanismo de ensino multidisciplinar, promovendo então, a difusão dos conceitos de desenvolvimento distribuído de software entre os discentes. Estes cenários foram definidos em três níveis que são:

Nível 1

Neste nível, o projeto de software é desenvolvido entre alunos de disciplinas distintas de um mesmo curso. É a primeira maneira de se desenvolver um projeto em DDS em um mesmo curso de graduação, pois uma mesma equipe é composta por alunos que cursam disciplinas diferentes. Um dos objetivos deste nível é preparar os docentes, alinhar o conteúdo programático e ajustar os sistemas de avaliação dos alunos. Além disso, neste nível já se faz necessário o preparo e a utilização de ferramentas CSCW (*Computer Supported Cooperative Work*).

Para o projeto deste nível, poucos docentes são envolvidos, possibilitando maior convergência entre as disciplinas ministradas. A possibilidade dos alunos se conhecerem também é maior, facilitando a comunicação entre eles, porém, esta característica torna o projeto menos próximo a realidade. Neste cenário as disciplinas são sequenciais e a grade é contínua.

Nível 2

O projeto executado neste nível agrega alunos de disciplinas e de cursos distintos. Neste caso é necessário alinhar as grades curriculares a fim de nivelar os alunos, mesmo em cursos diferentes, para que possam trabalhar em conjunto na mesma fase do projeto. A comunicação entre os discentes pode ser mais dificultosa neste cenário, necessitando de mais interações por meio de ferramentas ou reuniões para gerir os grupos e as atividades. As disciplinas podem exigir modificações no conteúdo para torná-las compatíveis.

Nível 3

Neste nível, o projeto é executado entre alunos de instituições e cursos distintos. Este é o nível mais próximo da realidade. Neste caso, alunos tendem a não se conhecerem e as reuniões presenciais tendem a ocorrer com uma frequência menor.

Há possibilidade de disparidade temporal entre cursos, por exemplo, um curso tem seu período semestral, enquanto que outro anual. Por esta razão é necessário ajustar os conteúdos disciplinares e os conceitos de DDS, de forma a possibilitar que alunos de disciplinas precedentes possam trabalhar em conjunto com alunos mais avançados na grade dos cursos. O fato dos cursos terem divisões temporais diferentes, pode ocorrer situações em que algumas disciplinas necessitem de artefatos que só serão confeccionados em outras, que ainda não tiveram início, comprometendo a sequência das atividades. A fim de minimizar este tipo de risco, as diretrizes dos cursos devem ser alinhadas para não comprometerem o andamento das disciplinas de ambos. Esta compatibilização pode ser feita por meio de ajuste mútuo.

A relação entre estes três possíveis níveis de projeto de DDS em curso de graduação está apresentada na tabela 2.



Tabela 2- Níveis de projeto de DDS na graduação e suas variáveis

Variáveis	Nível 1	Nível 2	Nível 3
Envolve várias disciplinas	Sim	Sim	Sim
Envolve vários períodos	Sim	Sim	Sim
Envolve disciplinas de um mesmo período	Provável	Improvável	Improvável
Envolve disciplinas de períodos diferentes	Necessário	Provável	Provável
A periodicidade das disciplinas é igual	Sim	Provável	Provável
Envolve cursos distintos	Não	Sim	Sim
Quantidade de docentes envolvidos	Baixa	Média	Alta
Docentes de uma mesma instituição	Sim	Sim	Não
As disciplinas iniciam/terminam juntas	Sim	Provável	Provável
Os professores conhecem todos os alunos	Sim	Provável	Não
Possibilidade de comunicação presencial	Alta	Média	Baixa
Possibilidade de ajustar o processo de software	Alta	Média	Baixa
Repasse de artefatos após o término da disciplina	Não	Improvável	Provável
Proximidade com um modelo real	Baixa	Média	Alta
Disparidade no nível dos alunos	Baixa	Moderada	Alta
O mesmo grupo foi utilizado em outro trabalho	Sim	Improvável	Não
Disparidade temporal	Não	Improvável	Possível

A tabela 2 apresenta as variáveis identificadas nos três possíveis cenários de um projeto de DDS, desenvolvido por alunos de graduação. O nível 3 aproxima-se mais da realidade, enquanto que o nível 1, menos.

Independente do nível do projeto, antes de iniciá-lo com os discentes é necessário criar um ambiente básico de trabalho. Tal ambiente deve ser composto por um conjunto de ferramentas e processos de trabalhos. A figura 2 elícita os elementos e o fluxo das atividades durante o decorrer de um primeiro experimento (execução de um primeiro projeto de DDS em um curso de graduação).

A próxima atividade é a difundir tal conjunto de ferramentas e processos de trabalho nas disciplinas. Essa difusão pode ser considerada um adicional ao conteúdo programático. A partir desta primeira iteração, os discentes conhecerão o primeiro conjunto de processos idênticos para desenvolver software. Nesta primeira interação é realizada uma difusão do processo por meio de ajuste mútuo.

O resultado é um conjunto de padrões, ferramentas e seus respectivos processos de uso, que serão utilizados durante o andamento das disciplinas. Os docentes responsáveis pelas disciplinas envolvidas dividirão os alunos em grupos, estabelecendo as atividades e a compatibilidade entre os processos que serão executados, dando início assim ao projeto.

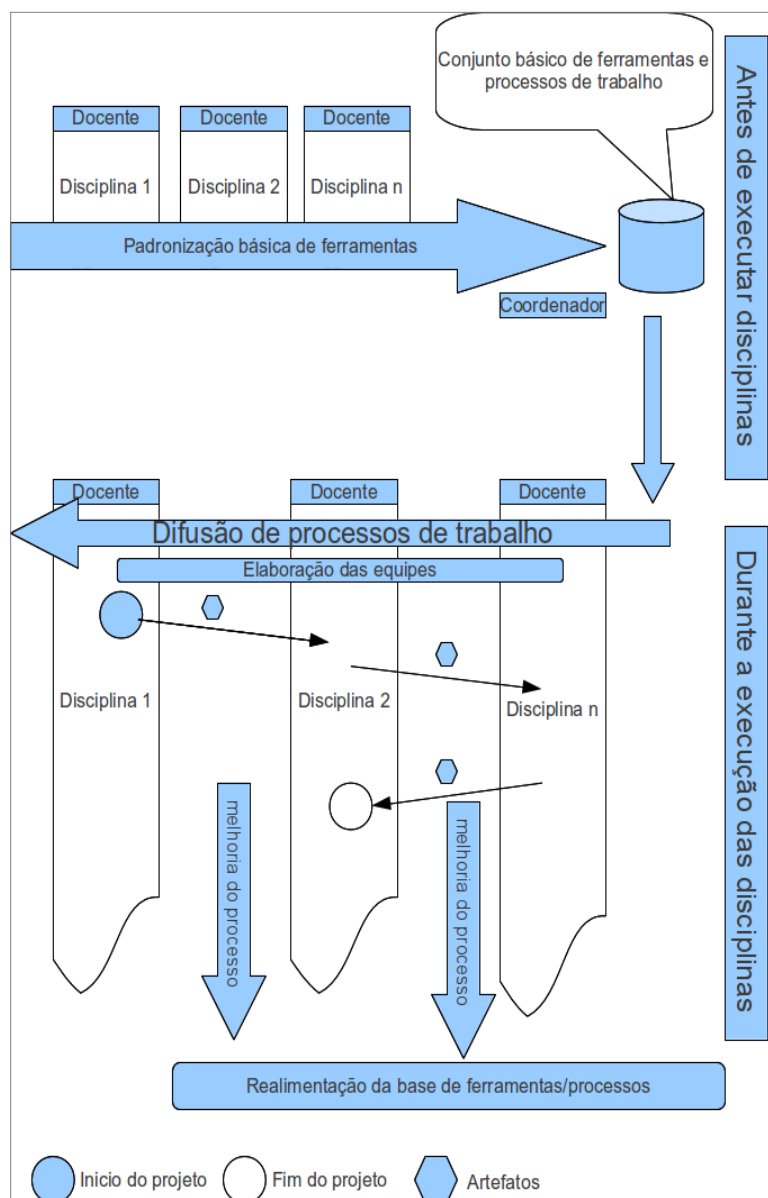


Figura 2 - Conjunto de ferramentas e processos de trabalho

As atividades atravessarão as disciplinas no momento em que cada uma abordar a fase em andamento do projeto, enviando os artefatos de saída para comporem uma nova atividade ou darem continuidade a uma já iniciada, na mesma disciplina ou em outras. Ao final do projeto, faz-se necessária análise de desempenhos, verificando os pontos que podem ser melhorados, gerando uma nova base de ferramentas e processos a serem trabalhados.

Há possibilidade de impasse, quando um grupo de uma disciplina recebe uma atividade que necessita de um conhecimento ainda não lecionado. Neste caso, o professor deve interferir e conduzir a um resultado. Tal impasse é ilustrado pela figura 3. O tempo de impasse, indicado na figura 3 pode ser minimizado de duas maneiras: a primeira delas é o bloqueio do projeto, até que tal conteúdo seja ministrado; a segunda é a orientação do professor na execução da atividade. A primeira solução prorroga a execução do projeto, mas, torna o discente independente das orientações diretas do professor. A segunda, cria certa



dependência no conhecimento do professor, que de certa maneira orienta a solução do trabalho por supervisão direta. Este problema é abordado por L'Erario (2008).

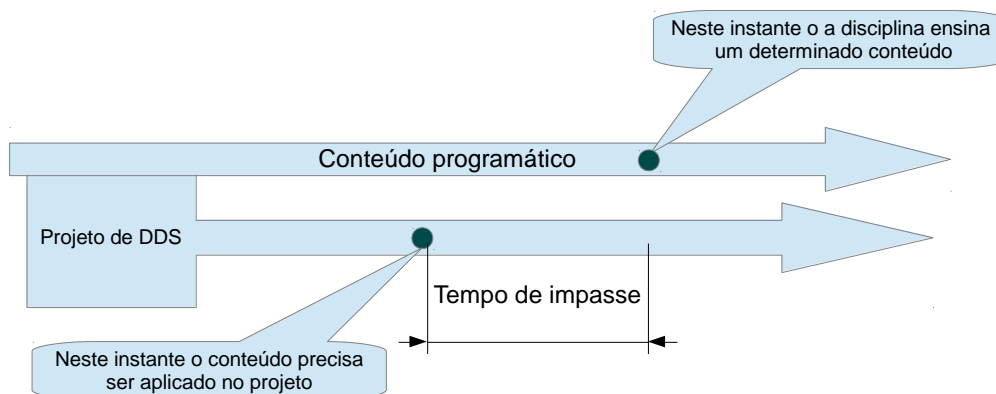


Figura 3 - Impasse no projeto

4. CONFIGURAÇÃO DO EXPERIMENTO

Dois cursos de graduação foram analisados com intuito de aplicar este framework. Embora os cursos sejam oferecidos na mesma instituição/campus, um curso é de Tecnologia em Análise de Desenvolvimento de Sistemas, oferecido somente no período noturno e outro curso é o de engenharia de computação, oferecido no período integral. Um primeiro experimento está em andamento, contemplando o curso de Tecnologia em Análise e desenvolvimento de Sistemas.

A ideia básica deste experimento é validar/realimentar o framework. Neste caso, a experimentação deve ser controlada e medida. Em um experimento, Fachin (2001) afirma que as variáveis são manipuladas de maneira pré-estabelecida e seus efeitos são suficientemente controlados e conhecidos pelo pesquisador. A ideia de experimento também abrange o conceito de simulação. Simulação é um experimento realizado em um modelo físico representativo de processo real. A qualidade das inferências retiradas da simulação, dependerá essencialmente da qualidade do modelo físico e de quanto ele se aproxima da realidade.

No primeiro experimento, em andamento, foram criados 8 grupos, agrupando em 4 disciplinas. O objetivo de cada grupo, constituído de alunos de várias disciplinas, é entregar no final do semestre, um projeto de software de e-commerce. Um ambiente de controle de versões foi configurado, para que os grupos pudessem utilizar. Há uso intenso de ferramentas de comunicação entre os integrantes do grupo. Uma disciplina chave foi identificada. Trata-se da disciplina denominada de Projeto Integrador. Esta disciplina proporciona um refinamento no mecanismo de coordenação entre os membros do grupo. De certa maneira, o processo de desenvolvimento é refinado constantemente e o grupo gera uma nova versão do projeto a cada semana. Este primeiro experimento é configurado como nível 1.

Um segundo experimento está programado para o segundo semestre de 2012, na Universidade Tecnológica Federal do Paraná – Campus Cornélio Procópio. Tal experimento, enquadra-se no nível 2. Os cursos envolvidos são: Tecnologia em Análise e Desenvolvimento de Sistemas e Engenharia de Computação. As disciplinas envolvidas são as disciplinas relacionadas a engenharia de software (processo de software), banco de dados, arquitetura de software, programação para a web e projeto integrador.



No total 9 grupos serão compostos por alunos de diferentes turmas e curso. Um ambiente de controle de versões foi configurado com o intuito de acompanhar o trabalho das equipes e facilitar a integração o desenvolvimento do projeto entre eles.

5. CONCLUSÕES

Este artigo apresentou a proposta de um framework para o ensino de desenvolvimento distribuído de software em cursos de graduação. Embora o primeiro experimento ainda esteja em andamento, foi possível analisar os dados do sistema de controle de versões. Todos os grupos foram direcionados a documentarem suas reuniões, gerando atas e controlando fielmente seus cronogramas.

No primeiro contato entre os integrantes do grupo, mesmo presencialmente, a disciplina de projeto integrador, conduziu todos a criarem uma primeira versão de seu processo. O projeto foi descrito por todos os grupos no formato de um documento de visão (documento de visão do RUP, conforme BERGSTROM, 2004). Uma primeira versão do processo, empregado pelo grupo foi criada. Neste caso, cada grupo é composto por subgrupos. Cada subgrupo representa um conjunto de alunos que cursam a mesma disciplina. Cada grupo alocou racionalmente as atividades para cada subgrupo/integrante de acordo com a(s) disciplina(s) que cursava(m).

Alguns grupos precisaram realocar atividades no decorrer do projeto. Fato este ocorreu, pois o grupo se preocupou inicialmente mais com o produto do software do que com o processo de desenvolvimento.

Houve uma preocupação muito grande por parte dos alunos ao assumirem o papel de gerente de projeto. Notou-se que apenas 2 dos 8 grupos elegeram um gerente de projetos na primeira reunião. Semanalmente as equipes geram um relatório em formato de ata, ou o registro da troca de e-mail, ou o registro da troca de mensagens instantâneas. O processo é refinado constantemente e o projeto ganha novas versões a cada semana.

O sistema de controle de versões é usado extensivamente. Os grupos foram direcionados a elaborar um documento de como seu projeto deve ser versionado, eles elaboraram um documento de gestão de configurações. A cada troca de informação entre os subgrupos, o processo e o projeto foram refinados. Notou-se que o procedimento de gestão adotado por cada um dos grupos foi consolidado em poucas semanas.

O uso das ferramentas de controle de versão e CSCW passaram a ser comum. Apenas 2 grupos não formalizaram (criaram um documento descritivo) o uso destas. Estes 2 grupos apresentam dificuldades no uso de tais ferramentas até então.

Além de várias habilidades tecnológicas necessárias para desenvolver um projeto distribuído, os alunos precisam aprender rapidamente a avaliar a sua habilidade e a de outros, adaptarem-se a um novo ambiente, dividir as tarefas, estabelecer laços de confiança, comunicar-se eficazmente, resolver problemas que possam surgir no grupo e praticar suas habilidades sociais. Com estas práticas, evidencia-se a valia do ensino de desenvolvimento distribuído para formação de um profissional melhor qualificado para o desenvolvimento de software.

Como trabalhos futuros, está em andamento um estudo da gestão de conhecimento de tal framework, cujo objetivo é manter o conhecimento mesmo com a rotatividade de discentes/docentes. Um estudo está em andamento analisando os processos desenvolvidos pelos grupos e o conjunto de artefatos gerados. Tal estudo visa rastrear os artefatos gerados com relação a sua evolução, acesso e aplicação dentro de cada grupo.



Agradecimentos

Agradecemos a Fundação Araucária (apoio ao desenvolvimento científico e tecnológico do Paraná) e a Secretaria de Estado da Ciência, Tecnologia e Ensino Superior do Paraná por ter viabilizado o desenvolvimento deste trabalho.

REFERÊNCIAS BIBLIOGRÁFICAS

BERGSTROM, Stefan. ADOPTING THE RATIONAL UNIFIED PROCESS: SUCCESS WITH THE RUP: Addison Wesley, 2004. 272 p

CARMEL E.; AGARWAL, R.. Tactical Approaches for Alleviating Distance in Global Software Development”. IEEE Software, v. 18, n. 2. março/abril 2001. p. 22-29.

CARMEL, E. “Global Software Teams – Collaborating Across Borders and Time Zones”. EUA: Prentice Hall, 1999. 269 p.

CARRINGTON, D.A. What can software engineering students learn from studying open source software? International Journal of Engineering Education, 24(4):729–737, 2008.

FACHIN, Odília. Fundamentos de metodologia. 3. ed. São Paulo: Saraiva, 2001.

FAULK, S.; YOUNG, M.; WEISS, D.; YU, L. ”Collaborative teaching of globally distributed software development: Community building workshop (CTGDSD 2011)”. Proceedings – International Conference on Software Engineering, pp. 1208-1209, 2011.

GUZMÁN, Javier García et al. How to get mature global virtual teams: a framework to improve team process management in distributed software teams. Software Quality Journal, v. 18, n. 1, p.409-435, 2010. DOI: 10.1007/s11219-010-9096-5

KAROLAK, D. W. “Global Software Development – Managing Virtual Teams and Environments”. Los Alamitos, EUA: IEEE Computer Society, 1998. 159 p.

L’ERARIO, A. PESSÔA, M. S. P.. Um método de ensino e práticas de desenvolvimento distribuído de software para cursos de graduação. In: COBENGE - CONGRESSO BRASILEIRO DE EDUCAÇÃO EM ENGENHARIA, 2008, São Paulo. Anais do XXXVI Congresso Brasileiro de Educação em Engenharia, 2008.

L’ERARIO, A.. “M3DS: um modelo de dinâmica de desenvolvimento distribuído de software”. 175 p. Tese (Doutorado) – Escola Politécnica da USP. Universidade de São Paulo, São Paulo, 2009.

L’ERARIO, A.; PESSOA, M. S. P. “An Analysis of the Dynamics and Properties of the Distributed Development of Software Environments: A Case Study”. In: SOFTWARE ENGINEERING RESEARCH AND PRACTICE. Los Angeles. p. 471 - 477, 2007.



L'ERARIO, Alexandre et al. A Distributed Software Development Environment Dynamics Model. In: ICGSE -INTERNATIONAL CONFERENCE ON GLOBAL SOFTWARE ENGINEERING, 7., 2012, Porto Alegre. Anais ICGSE. Porto Alegre: IEEE, 2012.

MARQUARDT, M. J.; HORVATH, L.. "Global Teams: how top multinationals span boundaries and cultures with high-speed teamwork". Davies-Black Publishing. Palo Alto, EUA. 2001. 246 p.

PATIL, S.; KOBZA, A.; KOBZA, A.; JOHN, A.; SELIGMANN, D. 2011. Methodological reflections on a field study of a globally distributed software project. *Inf. Softw. Technol.* 53, 9 (September 2011), 969-980. DOI=10.1016/j.infsof.2011.01.013 <http://dx.doi.org/10.1016/j.infsof.2011.01.013>

PRIKLADNICKI, R. "Can distributed software development help the practitioners to become better software engineers? Insights from academia". Paper presented at the Proceedings - International Conference on Software Engineering, 16-19, 2011.

PRIKLADNICKI, R.; AUDY, J. L. N. "MuNDDoS: Um Modelo de Referência para Desenvolvimento Distribuído de Software". 12 f. 2003, Seminário de Andamento, FACIN – PPGCC, PUCRS, Porto Alegre Ago. 2003

PRIKLADNICKI, R.; AUDY, J. L. N. "Uma análise comparativa de práticas de Desenvolvimento Distribuído de Software no Brasil e no exterior". In: XX SIMPÓSIO BRASILEIRO DE ENGENHARIA DE SOFTWARE. Florianópolis: SBES, 2006. p. 255 – 270.

A FRAMEWORK FOR TEACHING DISTRIBUTED SOFTWARE DEVELOPMENT IN UNDERGRADUATE COURSES

Abstract: *This paper presents a proposal for teaching distributed software development (DSD) framework in undergraduate courses. The scope of this model focuses on computing courses. This framework involves two aspects. The first is the institutionalization of theories related to the DDS. The second aspect is as how a project of distributed software development can be developed by students. The second aims to promote the practice of DSD among students. A first experiment is running on UTFPR-CP. Preliminary results are used to consolidate this framework.*

Keywords: *Teaching Distributed Software Development, Teaching DSD*