P@PSI *(Processo Ágil para Pequenos Sistemas)*

Processo de Desenvolvimento de Software Adaptado para o Ensino nos Cursos de Graduação

Marla Geller[1], Carlos A. P. Araújo[1], João Elias[2], Mythian Bastos[2], Tayna Roma[2], Pedro Martos[2], Clóvis Knebel[2]

[1]Centro Universitário Luterano de Santarém – CEULS/ULBRA, professores orientadores do projeto de pesquisa

[2] Centro Universitário Luterano de Santarém – CEULS/ULBRA, acadêmicos do Programa de Iniciação Científica

geller_marla@hotmail.com; pedroso_araujo@yahoo.com.br
joaobentes.junior@yahoo.com.br; mythian@bol.com.br; craniuzin@gmail.com; taynaroma@gmail.com

Abstract

This paper describes the work of a research group – GTA - Grupo of Agile Work, which treats of adaptations of methodologies of software development for small systems. The research group is formed by students of the degree course, under a teacher's orientation. The objective is to form a process that adapts to the needs of the academic context with small and inexperienced teams, facilitating the learning of development of systems in an organized way. Among the studied methodologies, SCRUM, the Programming Extreme and the Unified Process was used to create the process with agile and organized characteristics. The resulting process of the study is used in the several disciplines of the course of Systems of Information.

Resumo

Este artigo descreve o trabalho de um grupo de pesquisa - GTA - Grupo de Trabalho Ágil, o qual trata de adaptações de metodologias de desenvolvimento de software para pequenos sistemas. O grupo de pesquisa é formado por alunos do curso de graduação, sob a orientação de docentes. O objetivo é formar um processo que se adapte às necessidades do contexto acadêmico com equipes pequenas e inexperientes, facilitando o aprendizado de desenvolvimento de sistemas de forma organizada. Dentre as metodologias estudadas, o SCRUM, a Programação Extrema e o Processo Unificado foram utilizadas para criar o processo com características ágeis e organizado. O processo resultante do estudo é utilizado nas diversas disciplinas do curso de Sistemas de Informação.

1. Introdução

Os Cursos de Graduação da área da computação possuem disciplinas que trabalham diretamente com desenvolvimento de software, entre as quais podem ser citadas Engenharia de Software, Gerência de Projetos e Desenvolvimento de Sistemas e que estão entre as disciplinas que formam um analista de sistemas. Uma das qualidades de um bom desenvolvedor ou analista é a preocupação em utilizar um processo para o desenvolvimento, que possa ser aperfeiçoado a cada experiência e adaptado de acordo com as características dos sistemas, habilidades e tamanho da equipe, entre outras.

Muitos modelos de processos para orientar o desenvolvimento de software são abordados nos cursos de computação. Modelos tradicionais, como o modelo cascata, ou modelos iterativos e incrementais, como o Processo Unificado, estão entre os mais utilizados. Porém, todos são direcionados a atender grandes projetos. Sendo modelos definidos, possuem uma proposta pesada e exigente quanto às práticas da engenharia de software. Segundo Ambler (AMBLER,2004), o problema dessas perspectivas é que elas enfocam procedimentos prescritivos e os produtos que devem ser criados. São considerados métodos “pesados“ por fundamentar-se em regras definidas, inertes

a mudança dos requisitos e por possuírem um ciclo de desenvolvimento pouco adaptável e que devem ser completamente executados.

O que acontece, porém, no meio acadêmico, é que os sistemas possuem características próprias, como tamanho reduzido, pequeno número de participantes dos projetos, curto espaço de tempo, inexperiência dos participantes, entre outras. Esses fatores impõem uma busca por processos customizados que atendam as especificidades do contexto acadêmico. De acordo com Martin Fowler (FOWLER, 2007), em um processo adaptável o cliente tem melhor controle sobre o processo de desenvolvimento de *software*, e a cada iteração, pode tanto acompanhar o progresso quanto mudar a direção do desenvolvimento, tendo uma relação bem próxima com o desenvolvedor.

Para incrementar a pesquisa na área de Processos de Software para uso na academia, criou-se o GTA[1] que tem por objetivo estudar os diversos processos de desenvolvimento de software que surgem para facilitar e agilizar o desenvolvimento de sistemas. Sendo o foco desta pesquisa, pequenos projetos e pequenas equipes, o processo resultante do estudo é aplicado nas disciplinas do curso, como Engenharia de Software e Desenvolvimento de Sistemas bem como para orientar os Trabalhos de Conclusão de Curso.

A pesquisa do grupo inicia com o estudo do Scrum, Programação Extrema e Processo Unificado, na tentativa de criar um processo com características ágeis, porém organizado. A prática de desenvolvimento de sistemas com o uso de um processo se dá em geral, com o auxílio do Processo Unificado, processo prescritivo e definido. Ao mesmo tempo, incorporam-se ao mundo do desenvolvimento, os Processos Ágeis, que sugerem práticas e valores a serem utilizados no trabalho do dia a dia do desenvolvedor. Tem-se dessa forma, de um lado, os modelos tradicionais e preditivos de desenvolvimento de sistemas voltados para documentação, que são opções, geralmente, pesadas para pequenos projetos. Por outro lado têm-se os métodos adaptativos como o Scrum ou a Programação Extrema. Estes métodos não possuem uma prescrição de organização relacionada ao tempo, mas possuem definição de práticas e princípios bastante rigorosos, o que pode não ser suficiente para equipes sem muita experiência em desenvolvimento.

Neste cenário é bastante comum equipes iniciantes em desenvolvimento, como acadêmicos dos cursos de graduação, não adotarem processo algum, por não se adaptarem ao pesadelo da documentação e nem conseguirem se organizar sem um mínimo de prescrição do que deve ser feito, ou seja, estão entre os dois extremos (CHARETTE, 2001).

A proposta apresentada neste trabalho pelo GTA, é adaptar as metodologias ágeis para serem utilizadas com o Processo Unificado. Enquanto as metodologias ágeis sugerem as práticas o Processo Unificado organiza os fluxos. O objetivo é criar um processo que facilite às pequenas equipes, com pequenos projetos, com pouca experiência, e isso inclui os alunos iniciantes em desenvolvimento de sistemas, a tarefa de desenvolver sistemas com agilidade, flexibilidade, organização e com documentação suficiente para facilitar a reutilização das práticas positivas.

O projeto de pesquisa teve início em 2007, onde algumas características das necessidades e dificuldades dos alunos foram levantadas e os processos existentes no mercado foram avaliados. Como resultado da pesquisa em 2007, obteve-se um esboço do novo processo chamado *P@PSI (Processo Ágil para Pequenos Sistemas).* Este processo foi testado por alunos do Curso de Sistemas de Informação durante o segundo semestre de 2007 e primeiro semestre de 2008 e considerado de fácil utilização.

---

1 GTA – Grupo de Trabalho Ágil: formado por dois professores e alunos bolsistas que desenvolvem uma pesquisa com objetivo de criar um processo ágil para pequenos sistemas, enquadrando-se nesse contexto os sistemas desenvolvidos dentro da academia.

O trabalho aqui apresentado está organizado em tópicos, onde o item dois faz a fundamentação teórica das metodologias adotadas, o item três descreve as adaptações necessárias para criação do processo, o item quatro descreve como o processo foi aplicado, e finalmente nas considerações finais são apresentados os resultados obtidos.

3. Scrum, Programação Extrema e Processo Unificado

A opção por estas metodologias deve-se a análises preliminares feitas com alguns processos de desenvolvimento de software disponíveis e conhecidos no mercado. Dos processos estudados já consolidados na literatura pode-se citar o Processo Pessoal de Software, Processo Espiral, Prototipagem, Crystal, Scrum, XP[2], Desenvolvimento Adaptativo de Software (PRESSMAN, 2006) entre outros. Estudos correlatos ao tema foram também objetos de pesquisa para fundamentar o tema e conhecer outras propostas de soluções para o mesmo problema, podendo-se citar (PAIVA, 2004), (GARCIA, 2004), (SCHNEIDER, 20030], (HAZZAN, 2003), (ALVES, 2006).

3.1 Scrum

A metodologia Scrum compartilha com XP a adoção das práticas definidas no "Manifesto para o Desenvolvimento Ágil de Software" (Agile Manifesto, 2001). O foco da metodologia Scrum é a flexibilidade, a adaptabilidade e a produtividade. O resultado do processo deve ser um software que é realmente útil para o cliente (SCHWABER, 2002).

O ciclo de vida do Scrum é baseado em três fases: fase de planejamento, fase de desenvolvimento (*Sprint*) e fase de encerramento. A primeira e a última fase consistem em processos definidos, onde o fluxo de atividades é linear, podendo haver algumas iterações na fase de planejamento (SCHWABER, 1995). Nesta fase define-se o *Product Backlog*[3], que consiste em uma lista priorizada de todas as estórias de usuários que o sistema deve atender. A fase de desenvolvimento é dividida em ciclos chamados S*prints*, que são realizados em tempo determinado (não mais que 30 dias). Um *Sprint* contém uma funcionalidade ou funcionalidades *(Sprint Backlog*[4]*)* que devem ser implementadas. São os casos de uso priorizados. O *Sprint Backlog* é dividido em tarefas *(Sprints Diários*[5]*)* que não devem exceder 24 horas. Tem-se a cada final de *Sprint* um release ou uma parte do produto a ser avaliado ou até mesmo utilizado pelo cliente. Na fase de encerramento são feitas reuniões para analisar o progresso do projeto e demonstrar o software atual para os clientes. Nesta fase são feitas as etapas de integração, testes finais e documentação (PRESSMAN, 2006).

Uma equipe Scrum é formada pelo *Scrum Master* (gerente do projeto), que é responsável pela aplicação das regras do Scrum, que garante a plena funcionalidade e produtividade da equipe e que representa o escudo nas interferências externas; pelo *Scrum Owner* (dono do produto), que define as funcionalidades do produto, decide datas de lançamento do conteúdo, e pela equipe de desenvolvedores que deve ser pequena e multifuncional (programadores, testadores, desenvolvedores de interfaces, entre outros).

Pode-se fazer uma analogia do Scrum com o jogo onde todos trabalham ao mesmo tempo para conseguir o mesmo objetivo. "O estilo de corrida de revezamento onde é dada a cada participante da equipe a responsabilidade de uma parte do projeto, quando aplicado ao desenvolvimento de produtos pode conflitar com os objetivos de velocidade e flexibilidade máximas" (TAKEHUSHI,

2 XP – Extreme Programming

3 Product Backlog: conjunto de requisitos do sistema.

4 Sprint Backlog: itens do *Product Backlog* que a equipe se compromete em desenvolver em um período de até 4 semanas.

5 Sprints diários: definição da atividade a ser realizada em um dia e que faz parte do *Sprint Backlog*.

1986). Na interpretação do texto de Takehuchi (TAKEHUSHI, 1986) transcrito acima, tem-se a descrição do que acontece nos processos tradicionais. Ao invés disto, "um estilo holístico, onde a equipe busca, como em um jogo de futebol, de forma integrada, chegar ao gol, com passes de bola, pode servir melhor às atuais necessidades competitivas" (TAKEHUSHI, 1986).

3.2 Programação Extrema - *eXtreme Programming* - XP

XP é uma metodologia ágil para equipes pequenas a médias que desenvolvem software com requisitos vagos ou que mudam freqüentemente (BECK, 2004). Baseia-se em quatro valores que são: comunicação, simplicidade, feedback e coragem.

A XP possui práticas que são aplicadas a qualquer outro processo e que agregam valor à equipe de desenvolvimento. Entre estas práticas estão: programação em pares, cliente presente, reuniões breves, testes freqüentes, refatoração do código, integração contínua, semanas de 40 horas (TELES, 2004).

A XP assim como o Scrum é um processo adaptativo, ou seja, adota a mudança, ou melhor, considera a mudança uma característica intrínseca aos sistemas (WILLIAMS, 2003). Portanto, ao invés de prever o que pode acontecer no futuro, adapta-se às mudanças, baseando-se em situações concretas, que realmente acontecem. XP recebe críticas, por ter uma análise de requisitos muito informal e por não valorizar tanto a modelagem como o Processo Unificado.

3.3 Processo Unificado

Processo Unificado é uma estrutura genérica de processo que pode ser customizado adicionando-se ou removendo-se atividades com base nas necessidades específicas e nos recursos disponíveis para um projeto (KRUCHTEN, 2003). Possui quatro fases: Concepção, Elaboração, Construção e Transição. Dentro do Processo Unificado, fluxos de trabalho atravessam as fases do processo: Requisitos, Análise, Projeto, Implementação, Testes, (AMBLER, 2004). O Processo Unificado é um processo definido e preditivo que utilizado com a Linguagem de Modelagem Unificada, sugere a documentação do sistema através de modelos (SCOTT, 2003). Apesar de ter característica iterativa e incremental, prescreve as fases com planejamento e documentação mais extensa.

3.4 Escolha das Metodologias de Desenvolvimento

Seguindo a teoria de que um processo de desenvolvimento de software pode ser melhorado através da captura do que de melhor se adapta ao contexto, unindo práticas de processos conhecidos (KEENAN, 2004), o GTA tem por objetivo experimentar essas práticas em software para o meio acadêmico.

O objetivo do projeto determina que é necessário criar um processo de fácil utilização para os acadêmicos, bem como possibilitar o seu uso para fixar o conteúdo teórico das disciplinas relacionadas à Engenharia de Software.

O processo proposto une os princípios e práticas de métodos empíricos, como o Scrum e o XP, com o auxílio da organização do Processo Unificado que é um processo definido. Entende-se por processo definido aquele que define o que deve ser feito, quando e como enquanto um processo empírico define as ações baseando-se em situações concretas que acontecem durante as iterações (TELES, 2004). Optou-se por utilizar as duas metodologias - tradicional e ágil, pois se entende que são complementares.

4 Customização do Processo

O processo resultante - *P@PSI (Processo Ágil para Pequenos Sistemas)*, está representado na figura um através de um diagrama de atividades. É descrito como sendo gerenciado pelo Scrum, adotando práticas XP e com fluxos organizados do Processo Unificado. Para facilitar a atividade de gerenciamento, considerando o contexto em que está inserida a pesquisa, que são equipes inexperientes, entende-se que uma simplificação de um ciclo organizado facilita esse trabalho.

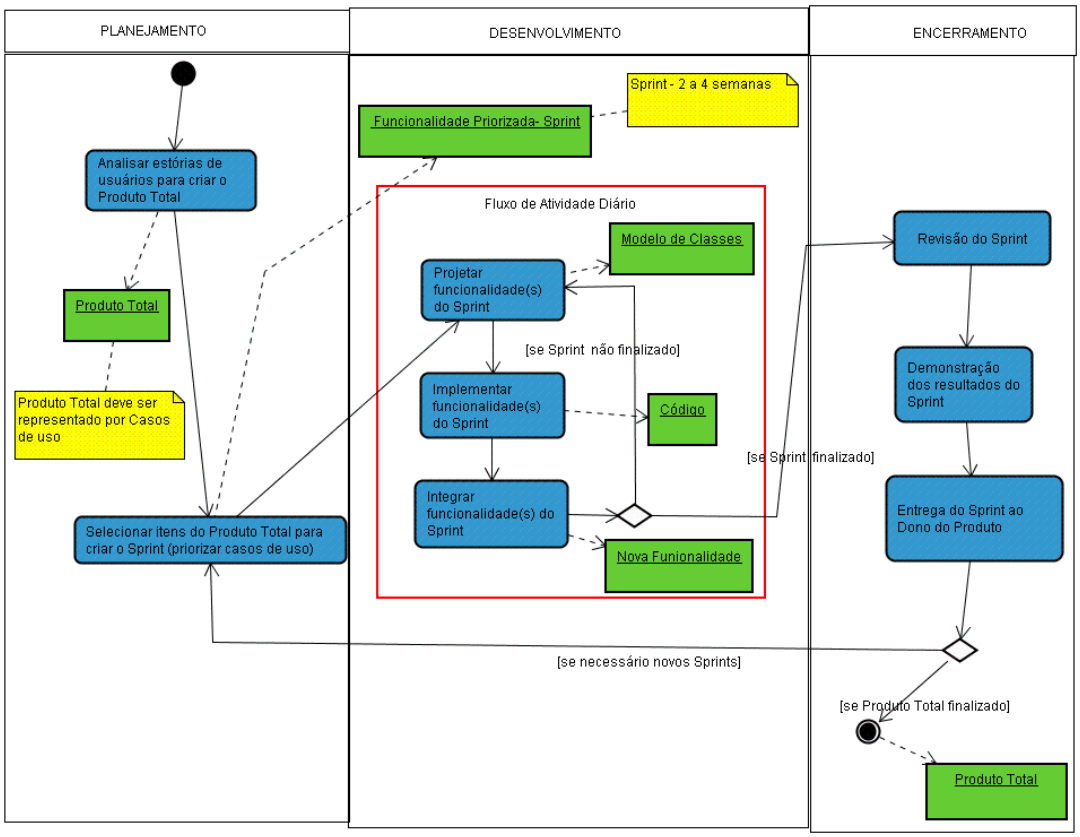

Figura 1. Diagrama de atividades para o processo resultante com as fases de Planejamento, Desenvolvimento e Encerramento com as atividades (em azul) e artefatos criados (em verde).

O processo constitui-se de três etapas representadas em raias: Planejamento, Desenvolvimento e Encerramento que possibilitam a iteratividade. Inicia-se com a fase de Planejamento, onde a equipe de desenvolvimento trabalha na captura de requisitos para definir o “Produto Total”. Os recursos utilizados para esta atividade são reuniões, entrevistas com os clientes, questionários, entre outras, que resultam em um diagrama de casos de uso inicial. Seguindo para a fase de Desenvolvimento, prioriza-se do Produto Total, uma funcionalidade ou pequenas funcionalidades, representadas por um ou mais casos de uso. Essa funcionalidade deve ser desenvolvida em um período determinado, (aconselhável não mais que quatro semanas). A esse pequeno pacote com os ca-

sos de uso priorizados chama-se *Sprint*[6]. O *Sprint* é solucionado em um fluxo diário de projeto, implementação e teste, ou seja, executa-se o projeto, desenhando e detalhando as classes, implementa-se o código e testa-se a unidade funcional diariamente, podendo este fluxo ser iterativo. As atividades desenvolvidas na fase de Encerramento incluem a revisão, demonstração e entrega do *Sprint* ao cliente. Neste ponto o processo retorna executando uma nova iteração com um novo conjunto de funcionalidades priorizadas, criando um novo *Sprint*, passando por todas as fases. Quando o Produto Total estiver finalizado o processo encerra-se.

Desta forma, com o objetivo de adaptar as práticas já existentes de desenvolvimento de software para projetos acadêmicos, propõe o GTA a customização entre XP, Scrum e PU. Observa-se que a complementaridade entre as metodologias acontece em diversas características e práticas de cada uma, como exemplo: Enquanto XP não prevê análise de riscos, um *Sprint* diário do Scrum ao ser organizado pelos fluxos do Processo Unificado, analisa os riscos diariamente. As práticas do XP facilitam à equipe inexperiente o comportamento e relacionamento da equipe no dia a dia, como programação em pares, propriedade coletiva do código, reuniões em pé, integração diária, programação orientada por testes enquanto que a ordem dos fluxos dada pelo Processo Unificado orienta as tarefas na seqüência do tempo. A natureza gerencial do Scrum possibilita a visão das atividades como um todo, fazendo com que a equipe trabalhe com um objetivo bem focado.

Princípios a serem seguidos:

- O projeto deve ser feito de forma simples, ou seja, apenas para a funcionalidade a ser solucionada no dia*;*
- O tempo de trabalho da equipe deve ser estabelecido de forma a não prejudicar as outras atividades acadêmicas;
- Reuniões diárias e rápidas onde são discutidos os problemas ocorridos no dia anterior, as soluções e os trabalhos a serem feitos no dia;
- Programação orientada a testes que facilita a criação do código conforme a solicitação dos requisitos, evitando surpresas;
- Programação em pares que já se constitui em uma prática adotada em todas as disciplinas de programação no curso e que facilita a construção da lógica da solução do problema, bem como visualiza erros de digitação;
- Os módulos desenvolvidos a cada dia devem ser integrados ao restante do software que é testado como um todo;
- A base de código deve ser unificada para que todos possam utilizá-lo.

5 Experimentos e Resultados - Aplicação do Processo no Curso de Sistemas de Informação

O objetivo de um Curso de Bacharelado em Sistemas de Informação é formar profissionais que possam usar os recursos computacionais para a solução de problemas de outras áreas do conhecimento, de maneira multidisciplinar. Para tanto é necessário que o aluno tenha oportunidade de vivenciar as diversas situações em que há exigência da solução dos problemas utilizando-se de seus conhecimentos (ALVES, 2006). As disciplinas que trabalham com desenvolvimento de sistemas como Engenharia de Software, Desenvolvimento de Sistemas de Informação, Linguagem de Programação Web, Gerência de Projetos e outras afins são bastante voltadas às práticas de desen-

6 Sprint: termo utilizado pela metodologia SCRUM para designar um pacote de funcionalidades de um sistema a serem desenvolvidas em um determinado tempo.

volvimento de sistemas para solução de problemas, e sendo assim, necessitam de um processo para que o sistema seja modelado e documentado.

A aplicação do processo no curso iniciou-se com a divulgação do mesmo pelos componentes do grupo de pesquisa GTA. O grupo fez a exposição do processo com o auxílio do diagrama de atividades, sugerindo as práticas diárias para que a agilidade no desenvolvimento seja um ponto a ser considerando em todas as fases. As dificuldades foram discutidas e anotadas para que melhorias no processo fossem feitas. Algumas práticas foram criadas para que as atividades se desenvolvessem de forma dinâmica dentro da sala de aula:

- A turma foi dividida em equipes de no máximo 4 alunos;
- Cada equipe escolheu um projeto de acordo com o conhecimento de um dos componentes com um ramo de negócios específico, portanto a presença do cliente foi caracterizada pelo próprio aluno;
- Outros papéis dentro da equipe foram distribuídos conforme a habilidade de cada membro; geralmente uma dupla cuidava da modelagem e outra dupla desenvolvia o código;
- O controle de tarefas de cada *Sprint* dentro do processo foi feito através da utilização do quadro de tarefas *Kanban*[7]. Desta forma a equipe sempre tinha uma visão geral do andamento do processo;
- Os encontros aconteciam em dois momentos: no horário da disciplina sob orientação do professor e no decorrer da semana conforme a necessidade de finalizar o *Sprint*.

Os conteúdos a serem contemplados nas disciplinas foram sendo apresentados conforme a necessidade para a boa execução do processo. Assim, por exemplo, foi necessária inicialmente uma fundamentação teórica sobre processo e análise e projeto orientado a objetos. Processos específicos foram pesquisados pelos próprios alunos como tentativa de entender as características, vantagens e desvantagens de cada um, sendo que alguns foram estudados com maior ênfase. Os diagramas UML foram apresentados gradativamente de acordo com o modelo a ser criado para cada fase do processo. Diagramas de casos de uso foram estudados na fase de planejamento, já os diagramas de classes e de interação foram introduzidos na fase de desenvolvimento. Desta forma os temas foram fixados de forma prática e gradativa, tendo-se como produto final um sistema analisado, projetado e desenvolvido pelos alunos.

6 Considerações finais e Trabalhos Futuros

No mundo criativo do desenvolvimento de sistemas encontram-se diversidades que exigem observação e estudos criteriosos. O que se pode constatar é que adaptações são necessárias em todo processo de desenvolvimento, pois quando muito definido e especificado torna-se engessado para as subjetividades inerentes ao próprio sistema, ao desenvolvedor e as características da organização. E, quando muito informal, não consegue organizar o trabalho de maneira que o cliente sinta-se seguro e participante do projeto.

Observa-se que a dificuldade de seguir um processo tradicional e definido em pequenos projetos com equipes inexperientes tem sido um fator propulsor de pesquisas para encontrar um meio termo entre o desenvolvimento caótico e a rigidez das regras da Engenharia de Software. Inclui-se também nesta problemática, como já citado, o ensino da prática do desenvolvimento de sistemas nos cursos de graduação, onde é necessário que se tenha organização e agilidade nos projetos acadêmicos.

---

7 Kanban: método que usa unidades padrão com cartões de tarefas afixados em cada uma. Esses cartões de tarefas vão sendo deslocados pelas unidades à medida que o trabalho vai sendo feito. Representam-se três unidades, nesta ordem: “a fazer”, “sendo feito” e “feito”.

As dificuldades encontradas no ensino de algumas disciplinas nos cursos de graduação sugerem maior dedicação à busca pela solução de problemas nessa área. Uma tentativa é a adaptação de processos para facilitar os pequenos projetos acadêmicos. No entanto, esse esforço não deve estar isolado, devendo acontecer na forma de projetos interdisciplinares, envolvendo as disciplinas de Engenharia de Software, Gerência de Projetos e Desenvolvimento de Sistemas com controle e avaliação constantes. O Grupo de Trabalho Ágil propõe atividades permanentes para que boas experiências possam ser reproduzidas, relatadas e aperfeiçoadas a cada período.

A aplicação do *P@PSI* no curso obteve um nível muito bom de aceitação e os comentários anotados resumem-se em: práticas sugeridas fáceis de serem aplicadas em uma sala de aula; representação gráfica do processo fácil de entender; visão do andamento das atividades facilita a obediência a um cronograma; agilidade no fluxo de projeto. As dificuldades referidas foram: grande intervalo de tempo entre os encontros perdendo-se às vezes o fluxo de trabalho; pouco tempo para estudar o processo e aplicá-lo, sugerindo que se faça em dois semestres utilizando duas disciplinas seguidas na grade; e algumas práticas custaram mais a tornarem-se habituais. Essas dificuldades encontradas no decorrer do desenvolvimento foram anotadas, avaliadas e soluções foram propostas, pois sendo um projeto experimental com o objetivo de encontrar melhores práticas, é fundamental o relatório das peculiaridades do sistema escolhido, do ambiente que está inserido, da equipe envolvida e das ferramentas utilizadas.

Como trabalhos futuros a consolidação do processo deve seguir com sua aplicação dentro da academia e a proposta de um template objetivo e claro, com todas as etapas descritas, suas atividades, operadores e artefatos a serem criados, para orientar o aluno como desenvolver um projeto com documentação ágil.

É necessário que um estudo seja feito considerando os desafios explicitados pela adoção da CMM aplicados ao processo. Estudo com esse objetivo já foi iniciado por um Trabalho de Conclusão de Curso de um aluno da graduação.

Como continuidade da formação do aluno para o mercado de trabalho, deve-se enfatizar a aplicabilidade do processo nas empresas de pequeno porte para que a solução proposta pelo processo seja também válida para esse contexto no mercado.

O Grupo de Trabalho Ágil espera com esta pesquisa encontrar não uma solução, mas experimentar práticas capazes de facilitar a solução para projetos acadêmicos de desenvolvimento de sistemas, buscar recursos para a eficiência no ensino das disciplinas nos cursos de computação, e em conseqüência caminhar para a formação de bons analistas e desenvolvedores.

Referências

AGILE MANIFESTO,2001.http://www.agilemanifesto.org/, acesso em 20/05/2007.

ALVES, A. and BENITTI F. Processo de Desenvolvimento Integrado de Disciplinas de Engenharia de Software. In: Anais do XXVI Congresso da SBC. WEI – XIV workshop sobre educação em Computação, 2006

AMBLER, S. Modelagem Ágil – Práticas Eficazes para a Programação Extrema e o Processo Unificado. Porto Alegre: Bookmann, 2004.

BECK, K.,. Programação Extrema Explicada – Acolha as Mudanças. Porto Alegre: Bookman, 2004.

CHARETTE, R.Fair Fight? Agile Versus Heavy Methodologies Cutter Consortium E-project Management Advisory Service, 2, 13, 2001.

FOWLER, M. The New Methodology. Disponível em: http://www.martinfowler.com/articles/newMethodology.html> Acesso em: 12 maio,2007.

GARCIA, F. P. et al. easYProcess: Um Processo de Desenvolvimento para Uso no Ambiente Acadêmico. Anais do Workshop de Educação em Informática – WEI, 2004.
HAZZAN,O e DUBINSKY Y. Teaching a Software Development Methodology: The Case of Extreme Programming. Proceedings of the 16th Conference on Software Engineering Educations and Training [CSEE&T 2003], Madrid, Spain, 2003.
KEENAN. Agile Process Tailoring and probLem analYsis (APTLY). In *Proceedings of the 26th International Conference on Software Engineering (ICSE'04)*
KRUCHTEN, P. Rational Unified Process made easy: A practioner's guide to the RUP, Addison-Wesley, 2003.
PAIVA, D.M.B et al. Definido Implantando e Melhorando Processos de Software em Ambiente acadêmico. VI Simpósio Internacional de Melhoria de Processo de Software. São Paulo:2004.
PRESSMAN, R. Engenharia de Software. 6a. ed. São Paulo: McGraw-Hill, 2006.
SCHNEIDER, J.G and JOHNSTON Lorraine. eXtreme Programming at Universities – An Educacional Perspective. Proceedings of de 25th Internacional Conference on Software Engineering. Portland, Oregon, 2003.
SCHWABER, K. Scrum Development Process, OOPSLA'95 Workshop on Business Object Design and Implementation. Springer-Verlag, 1995.
SCHWABER, K., BEEDLE, M. Agile Software Development with SCRUM, Prentice-Hall, 2002.
SCOTT, K.O Processo Unificado Explicado. Porto Alegre: Bookman, 2003.
TAKEHUSHI, H. et al.The New Product Development Game. *Harvard Business Review,January, 1986.*
TELES, Vinícius M. Extreme Programming. São Paulo: Novatec,2004.
WILLIAMS, L. et al.Agile Software Development.It's About Feedback and Change *in IEEE Computer Society, june, 2003*