

easYProcess:

Um Processo de Desenvolvimento para Uso no Ambiente Acadêmico

Francilene Procópio Garcia, Aliandro Higino Guedes Lima, Danilo de Sousa Ferreira, Fábio Luiz Leite Júnior, Giselle Regina Chaves da Rocha, Gustavo Wagner Diniz Mendes, Renata França de Pontes, Verlaynne Kelley da Hora Rocha, Vinicius Farias Dantas

Universidade Federal de Campina Grande, Departamento de Sistemas e Computação
Av. Aprígio Veloso, s/n, Bloco CN, Sala 201, 2º andar, CEP 58.109-970,
Campina Grande – PB – Brasil

{garcia,aliandro,danilo,fabio,giselle,gustavo,renata,verla,vinicius,yuska}@dsc.ufcg.edu.br

Abstract

To reach success in a software project, with a resulting quality product delivered in the predicted time, one must make use of a development process. However, the processes available in the market are targeted mainly at big-size projects, demanding a large number of principles and artifacts, increasing complexity and making their use difficult in small-size projects such as the ones found in the academia. This complexity may complicate the learning of Software Engineering concepts in the academy environment. In this context we introduce easYProcess, a simplified software development process based on RUP, XP and Agile Modeling principles. In this paper we present easYProcess as a process focused on the academia, with a report of its application in the Department of Systems and Computing of the Federal University of Campina Grande.

Resumo

Para alcançar sucesso em um projeto de software, obtendo um produto de qualidade e entregue no prazo previsto, é necessário o uso de um processo de desenvolvimento. Entretanto, os processos existentes no mercado têm como alvo principal projetos de grande porte, exigindo um grande número de práticas e artefatos, aumentando a complexidade e dificultando sua utilização em projetos de menor porte como os encontrados na academia. Essa complexidade tende a dificultar a aprendizagem dos conceitos de Engenharia de Software no ambiente acadêmico. Neste contexto apresentamos o easYProcess, um processo de desenvolvimento de software simplificado apoiado em práticas de RUP, XP e Agile Modeling. Este trabalho mostra o easYProcess como um processo para a academia, com um relato de sua aplicação no Departamento de Sistemas e Computação da Universidade Federal de Campina Grande.

1. Introdução

Segundo Humphrey (1990) [1], “Um processo de desenvolvimento de software é um conjunto de ferramentas, métodos e práticas usados para construir um produto de software”.

A produção de um software envolve aspectos teóricos e práticos, embasamento prévio de engenharia de software e a utilização de um processo de desenvolvimento, de um bom suporte ferramental, assim como de tecnologias aceitas pelo mercado.

A academia é o ambiente propício para que a teoria seja aliada à prática a fim de proporcionar ao estudante uma visão real do trabalho com Engenharia de Software. Para tanto, faz-se necessária a boa aplicação dos conhecimentos adquiridos ao longo do curso no desenvolvimento de um produto. O sucesso é mais facilmente alcançado quando se pode fazer uso de um processo de desenvolvimento de software adequado.

Em Hazzan e Dubinsky (2003) [2], é dito que “Processos de engenharia de software são difíceis de serem implantados, até mesmo na academia”. Comprovamos esta afirmação em disciplinas de Engenharia de Software do Curso de Ciência da Computação [3] da Universidade Federal de Campina Grande (UFCG) [4]. Vimos que a utilização de processos tais como *Rational Unified Process* (RUP) [5] e *eXtreme Programming* (XP) [6], em sua íntegra, não resultou em um nível aceitável de projetos bem sucedidos e de desenvolvedores capacitados.

Neste contexto, apresentamos o *easYProcess* (YP) [7], um processo de software simplificado que se apóia em práticas do XP, RUP e *Agile Modeling* (AM) [8]. O YP tem como objetivo auxiliar tanto a gerência do desenvolvimento de aplicações em disciplinas de engenharia de software quanto a aprendizagem dos conceitos desta disciplina na graduação, podendo também ser utilizado em projetos de pequeno e médio porte.

Para validar o seu uso na academia, o YP está sendo aplicado no Departamento de Sistemas e Computação (DSC) [9] da UFCG desde maio de 2003.

Este trabalho descreve como o processo foi formulado e os resultados obtidos com a utilização do YP no meio acadêmico, confrontando-o com práticas anteriormente utilizadas. O restante deste artigo está dividido da seguinte forma: inicialmente detalhamos o *easYProcess*, seu fluxo de trabalho, apresentando como se deu seu surgimento e exibindo algumas de suas principais características. Em seguida, descrevemos o uso do YP na academia. Por fim, tecemos algumas considerações sobre o trabalho realizado, citando alguns dos trabalhos futuros.

2. O Processo

A necessidade de se criar um novo processo surgiu devido às dificuldades encontradas em se adaptar os processos já existentes para o uso na academia, dificuldades estas observadas no âmbito do Curso de Ciência da Computação da UFCG. Para se adequar aos requisitos de utilização no meio acadêmico, um processo deveria:

- ♦ Ser adequado às características dos projetos desenvolvidos, tais como um escopo pequeno (já que os projetos são desenvolvidos geralmente no período de tempo

de um semestre letivo – 4 meses), área de aplicação com a presença de clientes reais, e uso de tecnologias consideradas estado da arte, em sintonia com a grade do curso. Além disso, o processo deve possibilitar uma avaliação contínua e iterativa do progresso do projeto;

- ♦ Diminuir a curva de aprendizado dos alunos, considerando que será a primeira aplicação prática de um processo. Deve-se focar na aprendizagem do processo com alguns elementos qualificadores: uma boa produtividade, bom uso do ferramental de apoio e geração mínima de artefatos, além de se buscar uma consolidação do entendimento das práticas e conceitos da Engenharia de Software;
- ♦ Ser de fácil entendimento e simples de se seguir, mas robusto e completo o suficiente para gerar produtos de qualidade, de forma a se moldar ao perfil do futuro profissional que irá utilizá-lo. Ao mesmo tempo, deve se adequar ao perfil dos alunos do curso, com boa prática em programação e embasamento teórico em Engenharia de Software, mas ainda sem maturidade para gerenciar o desenvolvimento de software sem um acompanhamento adequado.

O YP foi desenvolvido a partir de três fases: o estudo, a concepção e a implantação.

A fase de estudos foi direcionada pela observação da carência da utilização de processos de desenvolvimento de software em empresas, pelas características de processos já existentes e pelo que estava sendo apresentado na literatura referente à Engenharia de Software. A partir desse procedimento, percebeu-se que o mercado propunha a utilização de processos de desenvolvimento mais leves, com maior concentração na produção de código. Esse fato, somado à realidade dos estudantes de Computação da UFCG, que se mostravam motivados à prática da codificação, recomendou que o YP valorizasse aspectos da geração de código.

Com relação às práticas já existentes - RUP, XP e AM - foram analisados seus artefatos e condutas, a fim de se extrair os pontos relevantes, importantes de serem adotados em projetos de porte pequeno e médio. A escolha se deu após várias reuniões, onde as peculiaridades de cada proposta eram comparadas, ouvindo-se relatos de experiências de alunos com o uso de outros processos, e recomendando-se o mais adaptável para o ambiente acadêmico.

Os processos até então utilizados não satisfaziam as necessidades reais dos profissionais em formação. Algumas características que dificultaram a boa utilização de tais processos são enumeradas abaixo:

- ♦ RUP: foi considerado muito extenso e complexo para as exigências presentes. Havia uma canalização de esforços em apenas um dos três segmentos: processo, ferramentas, ou aplicação, não gerando um produto satisfatório, ou seja, que atendesse às necessidades e anseios de seus clientes. Considera que o desenvolvedor tem grande domínio dos conceitos e práticas de Engenharia de Software, o que nem sempre mostra-se verdade;
- ♦ UP1 [10]: uma adaptação do Processo Unificado, foi desenvolvido por alunos do Mestrado de Informática da UFCG [11], na forma de estágio docência. A adaptação não foi suficiente, uma vez que muito tempo era despendido na geração de artefatos e na aprendizagem do próprio processo;

- XP: requer uma equipe de desenvolvimento madura, deixando muitas práticas importantes em aberto. Apesar dos alunos estarem motivados a focar na produção de software de qualidade, ainda não estavam aptos para tomar algumas das decisões de acompanhamento do processo, tais como duração de iterações ou escolha dos artefatos a serem gerados.

Passamos então para a segunda fase, onde o YP começava a tomar forma. Inicialmente, foi formulada uma proposta para o processo e para cada parte de seu fluxo. Sua real viabilidade também foi avaliada. Definições e exemplos foram escritos por alunos que já utilizavam outros processos, usando uma linguagem comum e de simples assimilação, de forma a diminuir a curva de aprendizagem inerente, sempre buscando enfatizar a importância e a motivação das práticas mencionadas no processo. Exemplos reais também foram colocados à disposição dos usuários do processo, facilitando o aprendizado. O fluxo básico do YP está ilustrado na Figura 1.

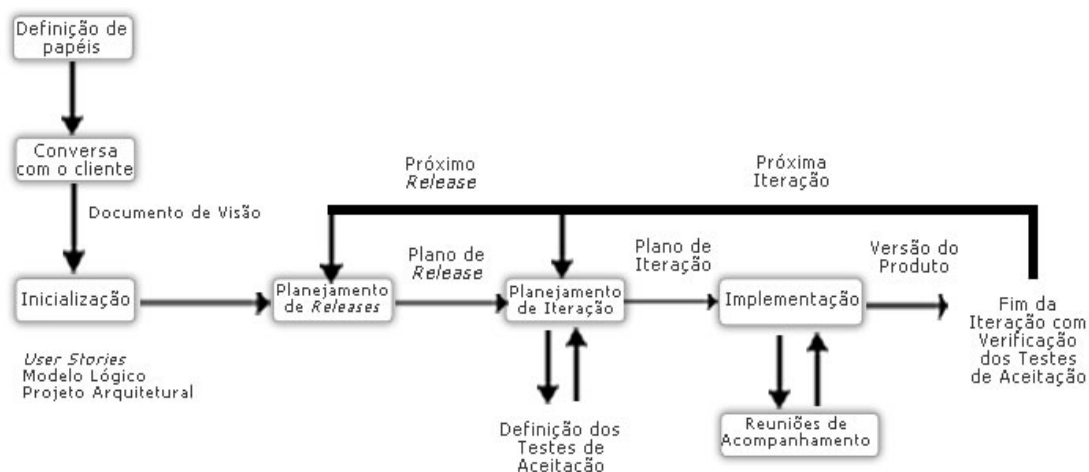


Figura 1 – Fluxo do easYProcess

A primeira etapa do processo consiste na **Definição de papéis**. O YP sugere os seguintes papéis: cliente, usuário, testador, desenvolvedor e gerente, podendo uma mesma pessoa desempenhar mais de um papel dentro do processo, principalmente quando se tratam de equipes de desenvolvimento pequenas. Em seguida deve ser realizada uma **Conversa com o cliente**, onde informações sobre o escopo do problema são adquiridas. A partir de então, a equipe encontra-se apta a gerar o documento de visão, que após ser validado pelo cliente, funciona como um acordo de trabalho entre cliente e equipe de desenvolvimento.

Na fase de **Inicialização** o cliente define as *User Stories* [12] e são elaborados o projeto arquitetural e o modelo lógico de dados, este último apenas se necessário. O cliente deve priorizar as *User Stories* e a equipe deve fazer uma estimativa inicial do tempo para implementação de cada uma delas. Baseado nessa estimativa pode-se então verificar a viabilidade de desenvolvimento do projeto no escopo e tempo definidos.

Parte-se então para o **Planejamento**, fase composta por dois planos, o de *release* [12] e o de iteração. Ambos possuem tempo fixo com variação de escopo permitida. Tratando-se do ambiente acadêmico são sugeridos três *releases*, cada um com duas iterações de duas semanas, por semestre letivo. O planejamento de um *release* só ocorre após o término do anterior, e da mesma forma para as iterações. No planejamento de *release* alocamos as *User Stories* de acordo com a priorização do cliente. No

planejamento de iteração as *User Stories* alocadas são quebradas em tarefas, e o cliente deve definir os testes de aceitação para cada *User Story*. Para auxílio na gerência o processo faz uso da Tabela de Alocação de Tarefas (TAT), na qual especificamos as *User Stories* envolvidas, tarefas, responsáveis, estimativas de tempo, tempo real consumido e *status* da tarefa. Estes dois últimos atributos são preenchidos apenas no fechamento da iteração.

Para a **Implementação**, o processo prega o uso de algumas práticas, tais como: *Design* Simples, Padrões de Codificação, Padrões de Projeto, Refatoramento e Propriedade Coletiva de Código, a fim de produzir um código com mais qualidade. Há uma grande ênfase na parte de testes, tanto de unidade, que validam pequenos módulos do sistema, como de aceitação, que de fato representam a satisfação ou não do cliente diante do que foi desenvolvido.

O andamento do processo deve ser coordenado pelo gerente através da **Reunião de Acompanhamento** semanal que visa recolher e analisar métricas. Nesta reunião faz-se uso do **Big Chart**, que deve ser gerado no mínimo uma vez por semana, da TAT, para o acompanhamento das tarefas, e da **Tabela de Riscos**, onde deve ser realizado o acompanhamento dos riscos anteriormente levantados e dos novos riscos que surgirem.

Nas reuniões de acompanhamento que coincidem com o fim de iteração é necessária a presença do cliente, pois este fará a verificação dos testes de aceitação, podendo dar uma *User Story* por concluída ou não.

A cada plano de iteração ou *release* o ciclo referente ao planejamento e implementação se repete.

Algumas características importantes sugeridas no YP:

- ♦ Participação efetiva do cliente: fato imprescindível para o sucesso do projeto;
- ♦ Papéis diferentes desempenhados pela mesma pessoa: necessário quando trabalhamos com equipes pequenas, como as encontradas nos projetos acadêmicos;
- ♦ *Releases* e iterações curtas: tratando-se do ambiente acadêmico são sugeridos três *releases*, cada um com duas iterações de duas semanas, de forma a propiciar uma avaliação contínua durante o semestre letivo;
- ♦ Variação no escopo e não no tempo, tanto no release quanto na iteração;
- ♦ Forte enfoque nos testes, em boas práticas programação, propriedade coletiva de código, e refatoramento;
- ♦ Acompanhamento do progresso do projeto através de métricas pré-definidas (*user stories* alcançadas, classes produzidas, testes realizados) reunidas no *Big Chart*;
- ♦ Manutenção de repositório de código com controle de versão.

3. Relato de Experiência

A disciplina de Laboratório de Engenharia de Software (LES) [13] vem sendo oferecida pelo DSC para o Curso de Ciência da Computação da UFCG desde o período 2000.1.

LES é uma disciplina obrigatória, sendo lecionada em todos os períodos letivos. Sua inserção na grade curricular do curso foi motivada pela necessidade de se colocar em prática conhecimentos sobre Engenharia de Software, a partir do desenvolvimento de projetos reais e com o apoio de ferramentas em todas as fases do projeto.

As turmas têm em média 20 alunos, que se organizam em equipes de quatro pessoas para a realização de um projeto, geralmente desenvolvimento para a *Web*, fazendo uso de banco de dados, necessitando de tecnologias tais como Java [14], JSP/Servlet [14], e uso de *frameworks* [15], entre outras. O desenvolvimento se dá sob a supervisão da professora da disciplina. Os projetos possuem clientes reais e externos à disciplina, normalmente professores do próprio departamento. O escopo e perfil dos projetos têm sido mantidos ao longo dos períodos, apesar de terem sido feitas adaptações no processo de desenvolvimento de software, na plataforma de desenvolvimento e em algumas das ferramentas de apoio.

As mudanças no processo de desenvolvimento adotado buscaram melhorar a produtividade das equipes de desenvolvimento, tanto no que diz respeito à curva de aprendizado das equipes, quanto com relação à conclusão satisfatória dos projetos.

No primeiro ano, a disciplina foi ministrada fazendo uso do Processo Unificado, ainda sem suporte ferramental. O grau de automação era irrisório, dificultando o trabalho da equipe de desenvolvimento. No ano seguinte, houve a inserção do suporte ferramental da Rational [5]. Analisando os resultados obtidos nesta fase, percebeu-se uma concentração de esforços na geração de artefatos, muitas vezes desnecessários aos projetos desenvolvidos, na sua maioria pequenos. Observou-se, em várias situações, que as equipes perdiam a dimensão do processo, restringindo a liberdade para trabalhar no produto a ser desenvolvido. Neste momento, ocorreu a primeira mudança com relação ao processo utilizado. No ano seguinte, fez-se uso do UP1. UP1 foi usado com suporte ferramental da Rational durante dois períodos, e no último período de sua utilização também contou com ferramentas livres. A escolha das ferramentas ficou a critério de cada equipe.

O uso do UP1 trouxe um ganho quantitativo/qualitativo no andamento da disciplina, percebeu-se melhoria, em especial, na gerência do projeto. A ausência de tempo para a realização de testes (unidade e funcionais) resultou em produtos de baixa qualidade. A baixa satisfação dos clientes em relação aos produtos contribuiu para que uma nova adaptação fosse feita, redimensionando o processo em uso pela disciplina. Um novo processo surge - o YP, adotado na disciplina nos períodos 2003.1 e 2003.2.

A implantação do novo processo se deu com a participação de alguns membros da equipe que o concebeu, atuando como monitores da disciplina. Frequentes reuniões eram feitas com toda a equipe YP, visando uma melhoria contínua no processo.

Após a primeira aplicação do YP na disciplina de Laboratório de Engenharia de Software foi feito um relatório da experiência, no qual pudemos observar ganhos na aplicabilidade de um processo de desenvolvimento por parte dos alunos e na qualidade do produto final. Estes ganhos se deram devido ao fato de que mais esforços foram concentrados no desenvolvimento e gerência, visto que o YP não demanda uma grande quantidade de artefatos, apenas o suficiente para manter uma boa gerência. Além disso, a simplicidade do processo faz com que o mesmo seja de fácil assimilação.

A facilidade de compreensão do processo e a diminuição do número de artefatos a serem gerados fizeram com que a utilização do YP diminuísse a curva de aprendizado dos alunos. Constatou-se que com a utilização dos processos anteriormente implantados na disciplina a curva de aprendizado registrava um período de seis semanas, enquanto que com a aplicação do YP esse número foi reduzido para três semanas.

Quanto ao número de artefatos, os processos aplicados antes do YP traziam uma carga maior de artefatos a serem produzidos. O processo UP1, por exemplo, exige a produção de artefatos tais como: diagramas de *Use Cases*, de classes, de seqüência, de colaboração, plano de testes, documento de visão, glossário e *post-mortem*. Enquanto o YP recomenda apenas a geração de nove artefatos, consumindo cerca de 8 horas para as atividades iniciais e cerca de 4 horas semanais para os artefatos utilizados pela gerência.

Além disso, o YP diz apenas quais artefatos devem ser gerados e não como fazê-los, deixando a equipe de desenvolvimento livre para utilizar a ferramenta que lhe for mais conveniente. Enquanto que os processos utilizados anteriormente fixavam a produção do artefato a uma determinada ferramenta, como por exemplo, o plano de testes com o *TestManager* [5] e o documento de visão com o *Requisite Pro* [5].

A Tabela 1 compara os números de artefatos produzidos por cada processo.

Tabela 1 – Número de artefatos produzidos x Processo utilizado

	RUP	UP1	YP
Número de artefatos	18	15	19

Nos oito períodos letivos de LES, em média 70 projetos foram realizados, abrangendo um total de 120 alunos. Em uma análise pontual pode-se verificar que o grau de sucesso dos projetos desenvolvidos na disciplina aumentou ao longo dos períodos, tendo seu maior índice com a utilização do *easYProcess*. Veja a Tabela 2:

Tabela 2 – Grau de sucesso dos projetos x Processo utilizado.

	RUP	UP1	YP
Grau de sucesso dos projetos	10%	50%	90%

Os problemas encontrados estavam relacionados, principalmente, com a inexperiência dos alunos na utilização plena de um processo e de alguns ambientes de apoio ao desenvolvimento até então desconhecidos pelas equipes, além da dificuldade de exercer alguns papéis presentes no processo. A ênfase na gerência e na comunicação com o cliente, pontos fortes do processo, foi um aspecto que demandou esforço maior das equipes.

Pode-se perceber que o YP alcançou as expectativas e anseios de um processo que pudesse ser utilizado em uma disciplina de forma objetiva e simples, contemplando os aspectos essenciais de um processo ágil. As equipes, em sua grande maioria, conseguiram realizar um bom trabalho com sua utilização e não se depararam com maiores problemas, pois a especificação do processo está de forma clara e de fácil acesso para todos os alunos. Os problemas encontrados foram contornados com a conduta de gestão da disciplina e com pequenas melhorias no processo, como a apresentação mais detalhada de alguns exemplos.

Um ganho marcante que deve ser citado é que os alunos que fizeram uso do YP chegaram à disciplina Projeto em Computação I [16] mais capacitados, uma vez que a conduta de um processo já estava bem estabelecida. Este conhecimento prévio permitiu

a diminuição do tempo necessário para que os alunos utilizassem outros processos além do *easYProcess*.

4. Avaliação Final

De acordo com os resultados obtidos após a implantação do *easYProcess* podemos concluir que o objetivo inicial de criar um processo simples e de fácil aprendizado foi alcançado. Os alunos de LES que utilizaram o YP demonstraram, ao final da disciplina, um preparo e conhecimento maior das práticas de Engenharia de Software do que alunos de turmas anteriores. O sentimento positivo sobre os alunos que aplicaram YP, devido ao sucesso final em seus projetos, foi fator importante para o processo de aprendizagem experimentado.

Com um maior enfoque na gerência, uma linguagem acessível e a apresentação de exemplos abrangentes, o YP permite um aprendizado mais consistente do uso correto e da importância de um processo durante o desenvolvimento de software. Por ser um processo leve, o YP possibilita aos integrantes das equipes concentrar seus esforços no produto que está sendo gerado, já que não precisam gastar muito tempo com a criação e manutenção de artefatos.

Outro aspecto importante a ser observado é que o YP apresenta uma plataforma de desenvolvimento clara e baseada em software livre. Com isto, quando os alunos saírem para o mercado estarão mais maduros, com a bagagem de um processo e ferramental gratuito, bem mais adequado ao uso pelo mercado, diminuindo custos (aquisição e manutenção de licenças de software) e aumentando a produtividade (rotina de desenvolvimento disciplinada).

5. Trabalhos Futuros

Passada a primeira fase de implantação do *easYProcess*, pretendemos agora continuar com os estudos e aprimoramentos do processo, de forma a mantê-lo sempre atualizado e de acordo com as necessidades de seus usuários e do mercado. Para este ano o aperfeiçoamento do *easYProcess* será focado em quatro áreas: melhorias do processo através do *feedback* obtido, extensões com a introdução de novos artefatos de usabilidade [17], criação de uma plataforma integrada de software e implantação em empresas locais.

Devido ao êxito obtido nos semestres anteriores, o YP continuará sendo utilizado na disciplina de Laboratório de Engenharia de Software, permitindo-nos acompanhar semestralmente a aceitação e o desempenho do processo. As críticas e sugestões apresentadas pelos alunos são analisadas, bem como as dificuldades por eles encontradas, de forma que o *easYProcess* se encontra em um processo contínuo de melhoria.

Outro objetivo será o de adaptar o YP para cenários aos quais ele ainda não atende completamente. Um exemplo é o suporte aos requisitos de usabilidade, que originalmente não era coberto pelo processo, mas que acabou se mostrando necessário. Tal suporte foi introduzido através de uma extensão, desenvolvida por um aluno do Mestrado em Informática da UFCG. As extensões permitem adicionar novas

características ao processo, tornando-o mais robusto, mas mantendo seu núcleo leve e simples. Portanto, a demanda por novas extensões será estudada.

Para dar suporte ferramental ao *easYProcess* será montada uma plataforma integrada baseada em software livre, com as ferramentas necessárias ao processo. Apesar de estas ferramentas já estarem presentes no processo, não há entre elas uma integração que facilite a gestão do desenvolvimento. Será mantido o foco em software livre e de qualidade.

Tratando-se de um processo de desenvolvimento que foca na melhoria da qualidade do software gerado por desenvolvedores em formação, espera-se, ainda, difundir no âmbito desses projetos os desafios explicitados pela adoção de ISO 9001 [18] e CMM [19]. Estes aspectos são tratados na disciplina de Engenharia de Software II [20].

Por fim, após a implantação do YP no meio acadêmico iremos testar sua aplicação em empresas, no acompanhamento de projetos de pequeno e médio porte. Já foram feitos contatos com empresas incubadas pela Fundação Parque Tecnológico da Paraíba [21]. Nosso intuito é oferecer às empresas uma solução de Engenharia de Software simples e de fácil utilização, capaz de melhorar o desenvolvimento de software, ao mesmo tempo em que difundimos no mercado a metodologia utilizada pelos alunos ainda em sala de aula, facilitando seu posterior ingresso no mercado de trabalho.

Referências

- [1][Humphrey-1990] Humphrey, W. S. (1990) “Managing the Software Process”
- [2][Hazzan e Dubinsky-2003] Hazzan, O. e Dubinsky, Y. (2003) “eXtreme Programming as a Framework for Student-Project Coaching in Computer Science Capstone Courses”
- [3][Curso de Ciência da Computação] - <http://www.ccc.ufcg.edu.br>
- [4][Universidade Federal de Campina Grande] - <http://www.ufcg.edu.br>
- [5][Rational Software] - <http://www.rational.com>
- [6][eXtreme Programming] - <http://www.extremeprogramming.org>
- [7][easYProcess] - <http://www.dsc.ufcg.edu.br/~yp>
- [8][Agile Modeling] - <http://www.agilemodeling.com>
- [9][Departamento de Sistemas e Computação] – <http://www.dsc.ufcg.edu.br>
- [10][UP1] - <http://www.dsc.ufcg.edu.br/~pasqueli/Les/up1.htm>
- [11][Mestrado em Informática da UFCG] – <http://www.dsc.ufcg.edu.br/~copin>
- [12][Glossário YP] - <http://www.dsc.ufcg.edu.br/~yp/glossario.htm>
- [13][Laboratório de Engenharia de Software] - <http://www.dsc.ufpb.br/~garcia/cursos/LES>
- [14][Java] – <http://java.sun.com>
- [15][frameworks] - Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (1994), *Design*

Patterns, Elements of Object Oriented Software, Addison Wesley, Reading, MA

[16][Projeto de Computação I] - <http://www.dsc.ufcg.edu.br/~garcia/cursos/ProjetoI>

[17][YP com usabilidade] - <http://www.dsc.ufcg.edu.br/~yp/sinteseYPUabilidade.htm>

[18][ISO-9001] - <http://www.iso.ch>

[19][CMM] - <http://www.sei.cmu.edu/cmm/cmm.html>

[20][Engenharia de Software II] - <http://www.dsc.ufcg.edu.br/~patricia/esii/>

[21][Fundação Parque Tecnológico da Paraíba] - <http://www.paqtc.rpp.br>