COMPARAÇÃO ENTRE OS PROCESSOS DOS MÉTODOS ÁGEIS: XP, SCRUM, FDD E ASD EM RELAÇÃO AO DESENVOLVIMENTO ITERATIVO INCREMENTAL

COMPARISON BETWEEN THE PROCESSES OF METHODS ÁGEIS: XP, SCRUM, FDD AND ASD WITH RESPECT TO THE DEVELOPMENT OF AN INCREMENTAL ITERATIVE

Priscila Basto Fagundes
SENAIsc-Florianópolis, E-mail: priscila@ctai.senai.br

Janice Inês Deters
SENAIsc-Florianópolis, E-mail: jan@ctai.senai.br

Sandro da Silva dos Santos
SOCIESC, E-mail: sandro.silva@sociesc.com.br

Resumo. Cada vez mais os métodos ágeis têm despertado o interesse da comunidade de Engenharia de Software como uma alternativa para o processo de desenvolvimento de sistemas de uma maneira mais rápida, eficiente e que atenda as reais necessidades dos clientes. Existem no mercado vários métodos disponíveis que utilizam a abordagem ágil e que, por seguirem os princípios ágeis, apresentam uma série de atividades semelhantes no seu processo de desenvolvimento. Este artigo apresenta uma comparação entre os processos propostos pelos métodos ágeis XP, Scrum, FDD e ASD, de forma a auxiliar a equipe de desenvolvimento na escolha do método que melhor se adapte a suas expectativas.

Palavras-chave: Métodos Ágeis, XP, SCRUM, FDD, ASD

Abstract. *More and more often the Software Engineering community has paid attention to agile methods as a more efficient alternative for system development that supports customer real needs. There are several methods in the market based on the agile approach and, because these methods use the same agile principles, a lot of activities in their development processes resemble that approach. This paper presents a comparison of the processes proposed by the agile methods XP, Scrum, FDD, and ASD, aiming at helping development teams in the choice of the agile method that best meets their expectations.*

Key-words*: Agile methods XP; Scrum; FDD; ASD*

1 INTRODUÇÃO

À medida que as organizações tornam-se cada vez mais dependentes da indústria do *software*, ficam mais evidentes os problemas relacionados ao processo de desenvolvimento de sistemas: alto custo, alta complexidade, dificuldade de manutenção, e uma disparidade entre as necessidades dos usuários e o produto desenvolvido (SOMMERVILLE, 2003).

Acreditando que o processo utilizado é um dos motivos para a ocorrência desses problemas, um segmento crescente da Engenharia de Software vem defendendo a adoção de processos mais simplificados conhecidos como métodos ágeis, que visam a desburocratização das atividades associadas ao desenvolvimento (FOWLER, 2001).

Os métodos ágeis têm despertado um grande interesse entre a comunidade de desenvolvimento de sistemas. E, acredita-se que devido a esta demanda, uma considerável quantidade de métodos apresentando características ágeis têm surgido nos últimos anos.

Para auxiliar os profissionais, ligados ao desenvolvimento de software, interessados na utilização da abordagem ágil, na escolha de um método ágil que melhor se adapte às necessidades dos seus projetos, equipes e/ou organizações, este artigo apresenta uma comparação entre os processos de desenvolvimento de alguns destes: *Extreme Programming* (XP) (BECK, 2000), *Scrum* BEEDLE et al, 1998), *Feature Driven Development* (FDD) (DE LUCA, 2002) e *Adaptive Software Development* (ASD) (HIGHSMITH, 2002). Esta comparação é o resultado de uma análise das atividades adotadas por cada um dos métodos.

2 MÉTODOS ÁGEIS

A "indústria do *software*" sempre contou com métodos cujos processos de desenvolvimento eram baseados em um conjunto de atividades predefinidas, descritas como processos prescritivos (AMBLER 2004), onde muitas vezes, o trabalho começava com o levantamento completo de um conjunto de requisitos, seguido por um projeto de alto-nível, de uma implementação, de uma validação e, por fim, de uma manutenção (SOMMERVILLE 2003).

A partir da década de 90, começaram a surgir novos métodos sugerindo uma abordagem de desenvolvimento ágil onde os processos adotados tentam se adaptar às mudanças, apoiando a equipe de desenvolvimento no seu trabalho (BECK, et. al., 2001). Estes novos métodos surgiram como uma reação aos métodos tradicionais[1] de desenvolvimento de sistemas, ganhando com o passar dos anos um número cada vez maior de adeptos.

Com o intuito de definir um manifesto para encorajar melhores meios de desenvolver sistemas, em fevereiro de 2001 um grupo inicial de 17 metodologistas, entre eles, Robert C. Martin, Jim Highsmith, Kent Beck e outros, formou a Aliança para o Desenvolvimento Ágil de *Software*, que formulou um manifesto contendo um conjunto de princípios que definem critérios para os processos de desenvolvimento ágil de sistemas (AMBLER, 2004). Os doze princípios (BECK, et. al. 2001), aos quais os métodos ágeis devem se adequar são:

a) A prioridade é satisfazer ao cliente através de entregas contínuas e freqüentes;
b) Receber bem as mudanças de requisitos, mesmo em uma fase avançada do projeto;
c) Entregas com freqüência, sempre na menor escala de tempo.
d) As equipes de negócio e de desenvolvimento devem trabalhar juntas diariamente;
e) Manter uma equipe motivada fornecendo ambiente, apoio e confiança necessários;
f) A maneira mais eficiente da informação circular através de uma conversa face-a-face;
g) Ter o sistema funcionando é a melhor medida de progresso;
h) Processos ágeis promovem o desenvolvimento sustentável;
i) Atenção contínua a excelência técnica e a um bom projeto aumentam a agilidade;
j) Simplicidade é essencial;
k) As melhores arquiteturas, requisitos e projetos provêm de equipes organizadas;
l) Em intervalos regulares, a equipe deve refletir sobre como se tornar mais eficaz.

Com base nos princípios descritos acima, cada um dos métodos ágeis - incluindo os utilizados nesta comparação - apresenta um conjunto de atividades a serem adotadas durante o processo de desenvolvimento do sistema. É com base nestas atividades que é realizada a comparação apresentada neste trabalho. É importante mencionar que antes de iniciar a comparação propriamente dita entre os métodos foi realizado um estudo detalhado de cada um dos quatro métodos (XP, Scrum, FDD e ASD).

3 COMPARAÇÃO ENTRE OS MÉTODOS ÁGEIS

O texto a seguir mostra os critérios utilizados na comparação e os respectivos resultados.

3.1 CRITÉRIOS UTILIZADOS

Os critérios adotados para servirem de base para a identificação das atividades propostas pelos Métodos Ágeis são as atividades sugeridas pelo Desenvolvimento Incremental. Sommerville (2003) afirma que os métodos ágeis são fundamentados no Desenvolvimento Incremental. E conforme puderam ser observado, as atividades sugeridas durante o seu processo de desenvolvimento são bastante semelhantes aos Princípios Ágeis.

No Desenvolvimento Incremental (Figura 1) os clientes inicialmente identificam, em um esboço, os requisitos do sistema e selecionam quais são os mais e os menos importantes. Em seguida é definida uma série de iterações de entrega, onde em cada uma é fornecido um subconjunto de funcionalidades executáveis, dependendo das suas prioridades.

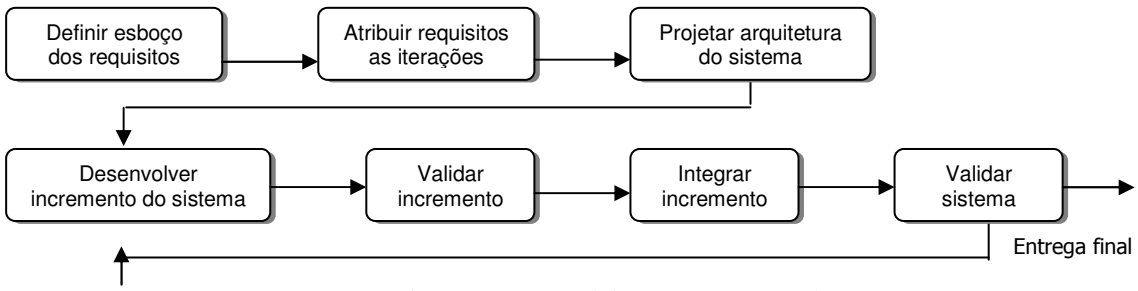

Figura 1. Desenvolvimento Incremental
Fonte: Adaptado de Sommerville (2003)

Após a identificação dos incrementos, as funcionalidades a serem entregues na primeira iteração são detalhadas e desenvolvidas. Paralelamente a este desenvolvimento, outras funcionalidades podem ser analisadas para fazerem parte dos outros incrementos. Uma vez que cada incremento é concluído e entregue, os clientes podem colocá-lo em operação. O fato dos clientes poderem experimentar o sistema gradualmente facilita o esclarecimento das funcionalidades para os incrementos subseqüentes e à medida que novos incrementos são concluídos, eles são integrados às iterações existentes, de modo que o sistema melhora a cada novo incremento entregue (SOMMERVILLE, 2003).

3.2 RESULTADOS DA COMPARAÇÃO

Para possibilitar uma melhor compreensão da comparação realizada, para cada uma das atividades do Desenvolvimento Incremental é apresentada uma tabela contendo um resumo da atividade correspondente em cada um dos métodos analisados, e, logo após, uma explicação

de como e quando estas atividades são realizadas durante o processo de desenvolvimento de cada um dos métodos.

3.2.1 Definição do Esboço dos Requisitos

No **XP**, a definição preliminar dos requisitos é feita a partir da escrita das *user stories* pelos clientes. As *user stories* são descrições textuais sucintas a respeito das funcionalidades do sistema (BECK, 2000). No ***Scrum***, de acordo com Schwaber e Beedle (2002), os requisitos conhecidos até o momento são listados dando origem ao *Product Backlog*. Estes requisitos são agrupados de acordo com as suas prioridades, apresentando as seguintes informações: descrição do requisito, tempo estimado para o desenvolvimento e responsável pelo desenvolvimento. No **FDD**, os requisitos já conhecidos são listados, definidos e documentados através de casos de uso ou *users stories*. Sugere-se também que sejam construídos um diagrama de classes UML e diagrama(s) de seqüência UML com o objetivo de auxiliar na compreensão do projeto (HIGHSMITH 2002). Segundo Highsmith (2002) e Abrahamsson (2002), o **ASD** não apresenta nenhuma atividade explícita em relação à definição preliminar dos requisitos, mas são realizadas sessões JAD[2] (*Joint Application Development)* com a presença dos representantes dos clientes, objetivando identificar os requisitos conhecidos até o momento. A Tabela 1 apresenta as atividades de definição do Esboço dos Requisitos.

Tabela 1 - Atividade - Definição do Esboço dos Requisitos

Método	**Especificação da Atividade**
XP	Clientes escrevem as *user stories*.
Scrum	Definição do *Product Backlog*
FDD	Geração de artefatos para a documentação dos requisitos
ASD	Requisitos definidos durante as sessões JAD

Fonte: Dos Autores (2008)

3.2.2 Atribuição dos Requisitos às Iterações

A atribuição dos requisitos às iterações no **XP** acontece com a participação do cliente sempre que uma nova iteração é iniciada (BECK 2000), onde são definidas as *user stories* que serão implementadas durante cada iteração. No ***Scrum*** os requisitos que serão desenvolvidos em cada *Sprint* são definidos a partir da lista de requisitos contida no *Product Backlog*. Durante a Reunião de Planejamento da *Sprint* acontece a alocação dos requisitos às *Sprints,* de acordo com a sua prioridade*,* dando origem ao *Sprint Backlog* (BEEDLE et al, 1998). No **FDD**, as características (requisitos) são agrupadas e priorizadas de acordo com a sua importância e dependência dando origem aos conjuntos de características que serão desenvolvidos durante as várias iterações (HIGHSMITH 2002). O **ASD** define a quantidade de iterações e quais os requisitos que serão implementados em cada uma delas. (ABRAHAMSSON 2002). As atribuições dos requisitos as iterações estão representados na Tabela 2.

Tabela 2 - Atividade - Atribuição dos Requisitos as Iterações

Método	Especificação da Atividade
XP	Equipe técnica e clientes definem as *user stories* que serão desenvolvidas nas iterações. As iterações duram de 1 a 4 semanas.
Scrum	Definição do *Sprint Backlog*. *As Sprints* (iterações) duram no máximo 30 dias.
FDD	As características são agrupadas, priorizadas e distribuídas aos responsáveis pelo seu desenvolvimento. As iterações duram no máximo 2 semanas.
ASD	Definição do número de iterações. As iterações duram de 4 a 8 semanas.

Fonte: Dos Autores (2008)

3.2.3 Projeto da Arquitetura do Sistema

O **XP** propõe que paralelamente à escrita das *user stories*, seja realizado o projeto da arquitetura do sistema (BECK 2000). Porém, não apresenta sugestões em relação a como este projeto deve ser feito, nem quais os artefatos devem ser gerados. O ***Scrum*** sugere que seja feito um projeto geral do sistema baseado nos itens do *Product Backlog* (SCHWABER E BEEDLE, 2002). E também não sugere nenhuma técnica associada a esta atividade. De acordo com Highsmith (2002a), De Luca (2002) e Abrahamsson (2002), o **FDD** sugere que seja construído um diagrama de classes da UML para representar a arquitetura do sistema. Para complementar o diagrama de classes também poderão ser gerados diagramas de seqüência da UML. O **ASD** não sugere nenhuma atividade relacionada ao projeto de arquitetura.

3.2.4 Desenvolver Incremento do Sistema

No **XP** a atividade de desenvolvimento dos requisitos é realizada durante as iterações (BECK 2000), onde, para cada conjunto de *user stories* selecionadas para fazerem parte da iteração são realizadas as atividades de modelagem e programação. No ***Scrum***, esta atividade acontece durante a *Sprint*, onde são implementados os requisitos contemplados no *Sprint .Backlog* Schwaber (1995) ressalta que o desenvolvimento dos itens dentro de uma *Sprint* não segue nenhum processo pré-definido. Durante cada uma das *Sprints* acontecem as Reuniões Diárias do *Scrum* que devem durar de 15 a 30 minutos e têm como objetivo que todos os envolvidos se mantenham informados sobre o progresso e as dificuldades encontradas. No **FDD** a implementação das classes e métodos correspondentes às características que serão desenvolvidas durante as iterações é antecedida pelas atividades de análise da documentação existente, geração de diagramas de seqüência para o conjunto de características, refinamento do modelo gerado durante as atividades de definição dos requisitos e do projeto da arquitetura do sistema (HIGHSMITH 2002). O **ASD** sugere os requisitos sejam desenvolvidos dentro das suas respectivas iterações e que, se possível, sejam desenvolvidos simultaneamente (ABRAHAMSSON 2002).

Tabela 3 - Atividade - Desenvolver Incremento do Sistema

Método	Especificação da Atividade
XP	Implementação das *user stories* que fazem parte da iteração corrente por duplas de programadores.
Scrum	Implementação dos requisitos contemplados no *Sprint Backlog* para a *Sprint* corrente.
FDD	Análise da documentação existente, geração de Diag. de Seqüência da UML, refinamento do modelo gerado nas atividades anteriores e implementação das características que serão desenvolvidas durante a iteração corrente.
ASD	Implementação dos requisitos que fazem parte da iteração corrente.

Fonte: Dos Autores (2008)

3.2.5 Validar Incremento

No **XP** a validação dos requisitos desenvolvidos durante a iteração ocorre através da execução dos testes de aceitação e dos testes de unidade. De acordo com Beck (2000), os programadores escrevem os testes de unidade e os clientes os testes de aceitação e ambos são escritos anteriormente ao processo de codificação e só após são executados. O ***Scrum*** sugere que a atividade de validação dos itens do *Sprint Backlog* implementados seja realizada no final da *Sprint* (SCHWABER E BEEDLE 2002). Segundo Abrahamsson (2002), no **FDD** os testes acontecem ao final de cada iteração, onde o conjunto de características implementado durante a mesma é testado através de testes de unidade pelos proprietários de classe – os programadores -. O **ASD** verifica a qualidade funcional do sistema gerado na iteração através da definição de grupos de clientes para revisar e testar a aplicação (HIGHSMITH 2002). Também a qualidade técnica é verificada, onde um par de programadores é responsável por revisar e avaliar o código do sistema. A validação do incremento é demonstrada na Tabela 4.

Tabela 4 - Validar Incremento

Método	Especificação da Atividade
XP	Os programadores executam os testes de unidade e os clientes executam os testes de aceitação.
Scrum	O *Scrum* não adota nenhum processo de validação pré-definindo.
FDD	Os testes e inspeções são executados pelos próprios programadores após a implementação.
ASD	São verificadas a qualidade técnica e funcional do sistema.

Fonte: Dos Autores (2008)

3.2.6 Integrar Incremento

No **XP**, Beck (2000) sugere que à medida que o código vai sendo gerado ele vai sendo integrado, evitando problemas de compatibilidade cedo. No ***Scrum***, a integração dos resultados das várias *Sprint*s acontece ao final de cada uma delas (SCHWABER E BEEDLE 2002). O *Scrum* não sugere como esta atividade deve ser realizada. No **FDD**, a integração do conjunto de características implementado durante a iteração corrente com os outros conjuntos acontece ao final de cada iteração, após os testes e inspeções (HIGHSMITH 2002). O **ASD** não sugere nenhuma atividade específica de integração dos incrementos desenvolvidos em cada. A Tabela 6 apresenta a atividade de Integrar o Incremento.

Tabela 6 - Atividade - Integrar Incremento

Método	Especificação da Atividade
XP	A integração acontece paralelamente ao desenvolvimento das *user stories*.
Scrum	Atividade realizada ao final de cada *Sprint*.
FDD	Atividade realizada após os testes no incremento.
ASD	-

Fonte: Dos Autores (2008)

3.2.7 Validar Sistema

Como o **XP** sugere que as integrações devem acontecer paralelamente às implementações das *user stories* e, conseqüentemente, aos testes, o mesmo não sugere atividades específicas de validação após a integração dos incrementos. Entretanto, ele coloca que o resultado da iteração pode ser posto em operação dentro da organização, onde será avaliado e poderá sofrer alteração de acordo com as considerações levantadas pelos clientes (BECK 2000). Conforme Schwaber e Beedle (2002), o ***Scrum*** valida a integração dos incrementos no último dia de cada *Sprint* através de uma reunião chamada Revisão da *Sprint*. Os participantes avaliam o incremento e decidem sobre as atividades seguintes. Após a entrega do incremento ao cliente, o *Scrum* sugere que seja realizado algum tipo de acompanhamento em relação à utilização do mesmo, onde são executados mais testes com o objetivo de garantir a qualidade do mesmo. Após a integração dos conjuntos de características, o **FDD** valida todos os incrementos gerados através das inspeções e dos testes de integração (DE LUCA 2002). O **ASD** não sugere nenhuma atividade específica de validação do sistema como um todo. A Tabela 7 apresenta a comparação da Atividade Validar Sistema.

Tabela 7 - Atividade - Validar Sistema

Método	Especificação da Atividade
XP	O sistema é disponibilizado ao cliente para que o mesmo realize validações.
Scrum	O cliente valida o sistema integrado em uma reunião no último dia da *Sprint.*
FDD	Esta atividade ocorre através das inspeções e dos testes de integração
ASD	-

Fonte: Dos Autores (2008)

3.2.8 Entrega Final

De acordo com Beck (2000), no **XP** a entrega final do sistema ocorre quando o cliente estiver satisfeito com o sistema e não tiver mais nada a acrescentar em relação às funcionalidades. No ***Scrum***, o sistema é entregue quando não existirem mais itens no *Product Backlog* a serem desenvolvidos (SCHWABER E BEEDLE 2002). O **FDD** sugere que todos os conjuntos de características devem passar pelas etapas de projeto e construção até que o sistema esteja pronto para ser entregue (HIGHSMITH 2002, ABRAHAMSSON 2002). E, por fim, no **ASD** esta atividade também só é realizada após não existirem mais requisitos a serem implementados (HIGHSMITH 2002). A Tabela 8 apresenta a Atividade de Entrega Final.

Tabela 8 – Atividade de Entrega Final

Método	Especificação da Atividade
XP	Cliente satisfeito com o sistema
Scrum	Todos os itens no *Product Backlog* desenvolvidos
FDD	O sistema é entregue após todos os conjuntos de características implementados
ASD	Todos os requisitos desenvolvidos.

Fonte: Dos Autores (2008)

4 CONCLUSÕES

Este trabalho apresentou uma comparação entre os processos XP, Scrum, FDD e ASD, levando em consideração o Desenvolvimento Incremental. Esta abordagem é diferente das apresentadas em (ABRAHAMSSON ET AL, 2003) e (COHN, 2002). Em (ABRAHAMSSON ET AL, 2003), é apresentado uma comparação entre os métodos ágeis ASD, AM, *Crystal Family*, DSDM, XP, FDD, ISD, PP e *Scrum* em relação aos seguintes critérios: Ciclo de vida de desenvolvimento de software; Administração de projeto; Princípios abstratos *versus* direção concreta; Predefinição universal *versus* situação adequada; e Apoio empírico. E em (COHN, 2002), é apresentado um estudo cujo objetivo é verificar se os métodos ágeis FDD, *Scrum*, XP, *Crystal Family* e DSDM atendem os princípios contidos no Manifesto Ágil.

De acordo com a comparação apresentada neste artigo, pode-se concluir que os métodos ágeis selecionados para fazerem parte deste estudo apresentam atividades bastante semelhantes em relação aos seus processos de desenvolvimento. Porém, algumas atividades apresentam particularidades, como por exemplo: a programação em dupla, as *user stories* escritas pelos clientes e a escrita dos testes antes da implementação do XP; as reuniões de planejamento, diárias e de revisão adotadas pelo *Scrum;* as inspeções de código de FDD; e as sessões JAD sugeridas pelo ASD.

Acredita-se que o presente estudo possibilita aos interessados em utilizar a abordagem ágil no desenvolvimento de seus sistemas, obterem uma visão geral das atividades propostas pelos métodos ágeis de maneira a auxiliá-los na escolha de qual método adotar ou mesmo, de reduzir o tempo gasto com estudos sobre métodos que não se adaptarão as suas necessidades.

Entretanto, sugere-se que a equipe que optar pela utilização de algum dos métodos apresentados obtenha maiores detalhes sobre as práticas adotadas durante cada uma de suas atividades, bem como dos papéis destinados a cada um dos envolvidos no projeto, de maneira a garantir uma maior eficiência em relação à aplicação do método. Em (FAGUNDES, 2005) pode-se encontrar uma descrição detalhada de cada um destes métodos, bem como um *framework* contendo as atividades sugeridas por cada um dos métodos de forma a permitir a escolha de quais delas utilizar durante o processo de desenvolvimento de um sistema.

REFERÊNCIAS

ABRAHAMSSON, P., Salo, O., RONKAINEN, J., & WARSTA, J. **Agile Software Development Methods**: Review and Analysis. Espoo 2002. VTT Publications 478, 2002.

ABRAHAMSSON, P., WARSTA, J., SIPONEN, M.T., & RONKAINEN, J. **New Directions on Agile Methods**: A Comparative Analysis. In: ICSE 2003, USA

AMBLER, S. **Modelagem ágil**: práticas eficazes para a Programação Extrema e o Processo Unificado. Porto Alegre: Bookman, 2004.

BECK, K. **Extreme Programming Explained**. Massachusetts: Addison-Wesley, 2000.

BECK, K. et al.**Agile Manifesto**. 2001. Disponível em: <http://www.agilemanifesto.org>. Acesso em: 24/01/2008

BEEDLE, M; et al. **SCRUM:** An extension pattern language for hyperprodutive software development. *In*: Pattern Languages of Programs'98 Conference, Monticello, 1998.

COHN, M. **Selecting an Agile Process**: Comparing the Leading Alternatives. 2002. Disponível em: <http://www.mountaingoatsoftware.com/pres/Selecting 021015.pdf.>. Acesso em: 20/01/2008

DE LUCA, J. **Feature-Driven Development (FDD) Overview Presentation**. 2002. Disponível em: <http://www.nebulon.com/articles/fdd/download/ fddoverview.pdf >. Acesso em: 10/07/2007

FAGUNDES, P. **Framework para Comparação e Análise de Métodos Ágeis.** Florianópolis: 2005. Dissertação (Mestrado em Ciência da Computação) . Universidade Federal de Santa Catarina, Florianópolis, 2005.

FOWLER, M. **The New Methodology**. 2001. Disponível em: <http://www.martinfowler.com/articles/newMethodology.html>. Acesso em: 12/12/2007.

HIGHSMITH, J. **Agile Software Development Ecosystems**. Boston: Addison Wesley, 2002.

Jeffries, R. **XP Magazine Contents**: What is Extreme Programming? 2001. Disponível em: <http://www.xprogramming.com/xpmag/index.htm>. Acesso em: 11/11/2007

RUMBAUGH , J. What Is a Method? **Journal of Object Oriented Programming.** October, v.26, n. 17 p. 10–16, 1995.

SOMMERVILLE, I. **Engenharia de software**. São Paulo: Addison-Wesley, 2003.

SCHWABER, K. Scrum Development Process. In: OOPSLA'95 Workshop on Business Object Design and Implementation. Springer-Verlag, Atlanta, 1995.

SCHWABER, K; BEEDLE, M. **Agile Software Development with SCRUM**. New Jersey: Prentice Hall, 2002.

[1] Segundo Fowler (2001), os métodos que utilizam processos prescritivos são conhecidos como métodos tradicionais.

[2] WOOD, J.; SILVER, D. **Joint Application Development**. 2nd ed. New York: Wiley, 1995.

Originais recebidos em: 11 fev. 2008.
Texto aprovado em: 18 mar. 2008.

SOBRE OS AUTORES

Priscila Basto Fagundes

Natural de Pelotas no Rio Grande do Sul. Bacharel em Análise de Sistemas pela Universidade Católica de Pelotas – UCPEL em 1999 e Mestre em Ciência da Computação pela Universidade Federal de Santa Catarina – UFSC em 2005, onde pesquisou sobre os Métodos Ágeis propondo um *Framework* para Análise e Comparação dos Métodos Ágeis. Atualmente atua como Coordenadora do Curso Superior de Tecnologia em Redes de Computadores e docente no Curso Técnico de Programação de Computadores na área de Análise e Projeto de Software e no Curso de Pós-Graduação em Engenharia de Software com UML na área de modelagem de negócios e modelagem de requisitos, todos estes oferecidos pela Faculdade de Tecnologia SENAI/SC - Florianópolis. Possui como principal área de interesse de pesquisa engenharia de requisitos e processo de desenvolvimento de software.
E-mail: priscila@ctai.senai.br

Janice Inês Deters

Possui graduação em Sistemas de Informação - Faculdades Franciscanas (1999) e mestrado em Ciências da Computação pela Universidade Federal de Santa Catarina (2003). Atualmente é professora mensalista - Serviço Nacional de Aprendizagem Industrial - Departamento Regional de Santa Catarina. Tem experiência na área de Ciência da Computação, com ênfase em Sistemas de Informação. Atua principalmente nos seguintes temas: Engenharia de Software, Linguagem de Programação, Sistemas Tutores Inteligentes, informática na educação.
E-mail: jan@ctai.senai.br

Sandro da Silva dos Santos

Natural de Pelotas no Rio Grande do Sul. Bacharel em Análise de Sistemas pela Universidade Católica de Pelotas – UCPEL em 1999 e Mestre em Ciência da Computação pela Universidade Federal de Santa Catarina – UFSC em 2002, onde pesquisou sobre os *Participatory Design* na implementação de sistemas de informação para pequenos agricultores. Atualmente atua como docente do Curso de Graduação em Administração de Empresas da Sociedade Educacional de Santa Catarina - FGV na área de Tecnologia da Informação, Sistemas de Informação e Análise e Projeto de Sistemas. Possui como principal área de interesse de pesquisa em Informática aplicada à educação e o processo de desenvolvimento de software.
E-mail: sandro.silva@sociesc.com.br