Uma abordagem para o desenvolvimento de adaptações em ERPs baseada em métodos ágeis

Cleiton Garcia[1,2], Daniel Schreiber[1], Sheila Reinehr[1]

[1]Programa de Pós-Graduação em Informática Aplicada da Pontifícia Universidade Católica do Paraná – PUCPR - Curitiba – Paraná - Brasil

[2] WEG Equipamentos Elétricos S.A. - Jaraguá do Sul – SC – Brasil
{cleitonsg,xiraba}@gmail.com, sheila.reinehr@pucpr.br

Resumo. *Empresas de todos os tamanhos buscam gerenciar, padronizar e otimizar seus processos de negócio a partir da implantação de sistemas ERP (Enterprise Resource Planning). No entanto, estes sistemas nem sempre são aderentes a todos os processos de negócio. Por isso, faz-se necessário o desenvolvimento, muitas vezes massivo, de novas funcionalidades e ainda adaptações às existentes. As alterações realizadas exigem flexibilidade, tanto dos processos da empresa, quanto dos processos do sistema. Este artigo relata a experiência obtida na utilização de métodos ágeis para a construção de um processo de desenvolvimento, com definição das atividades de planejamento e execução para adaptações em sistemas ERP.*

Abstract. *Organizations of any size are seeking to manage, standardize and optimize their business processes by implementing Enterprise Resource Planning Systems (ERP). However, these systems are not always compliant to all business processes. This leads to a sometimes massive development of new features or updates to existing features. The necessary updates demands flexibility, either from business processes, as from the systems processes. This paper describes the experience in using agile methods to build a development process to define planning and execution activities to adapt ERP systems.*

1. Introdução

O processo de engenharia de software possui um importante papel na construção de produtos de software e exerce grande influência na sua qualidade. A ausência de um processo de software bem definido resulta em uma equipe trabalhando de modo empírico para gerar um produto, mas de maneira imprevisível e muitas vezes desordenada. Nesta situação, o êxito do trabalho depende do esforço das pessoas envolvidas, que muitas vezes tornam-se heróis pelo resultado e sucesso de seu trabalho. Entretanto, esta não é uma condição sustentável para a organização (Kruchten 2003).

Organizações maduras empregam processos bem definidos e que suprem as suas necessidades para o desenvolvimento de sistemas de modo consistente e independente do indivíduo que os produziu. Isto é especialmente importante quando se leva em consideração que a alta rotatividade das pessoas é uma realidade de mercado, devido à oferta crescente aos profissionais de Tecnologia da Informação (TI). A dependência de profissionais altamente capacitados tem sido um problema para as organizações. Os produtos de software, principalmente os que suportam os processos de negócio críticos

para a organização, precisam continuar o seu ciclo de vida, mesmo que os desenvolvedores sejam substituídos. A definição de um processo de software deve estabelecer e formalizar informações sobre: atividades e papéis; os artefatos de entrada e saída que devem ser criados ou mantidos em cada atividade; os procedimentos e ferramentas utilizados; e o modelo de ciclo de vida utilizado (Fuggetta 2000).

Grandes fornecedores de produtos tipo Enterprise Resource Planning (ERP) possuem processos de software voltados para a implantação de suas soluções, como o *Oracle Unified Method* (OUM) e o *Accelerated SAP* (ASAP). Conforme Esteves e Jorge (2001), o objetivo destes métodos é reduzir o tempo dos projetos de implantação. Após a fase de implantação, este processo tem se mostrado pouco eficiente para as manutenções evolutivas e para o desenvolvimento de novos módulos.

Este relato tem por objetivo apresentar a experiência obtida na definição e customização de um processo de software, baseado em práticas ágeis, que tem sido executado em uma empresa. Estas melhorias iniciaram em 2008 com a implantação em uma equipe (célula ágil). Com a percepção dos ganhos e satisfação dos clientes, este processo ágil foi propagado para as demais equipes da organização.

O trabalho está assim organizado: a seção 2 descreve os métodos ágeis; a seção 3 o processo de software elaborado; a seção 4 relata sua implantação, detalhando os pontos de ação e os resultados obtidos; e, por fim, a seção 5 apresenta as considerações.

2. Métodos ágeis para planejamento e desenvolvimento

O manifesto ágil expressa princípios e valores que norteiam o trabalho de uma equipe ágil. O objetivo primordial ágil é a capacidade de responder às mudanças de requisitos as quais muitas vezes reduzem a robustez do processo, por isso é importante buscar o equilíbrio para satisfazer a necessidade da organização (Rosenberg *et al.* 2005).

No processo aqui apresentado foram utilizados, de forma combinada, três processos ágeis: Scrum, *eXtreme Programing* (XP) e *Open Unified Process* (OpenUP).

2.1. Scrum

O Scrum é um framework ágil voltado ao planejamento e organização que pode ser empregado em diferentes processos e técnicas. Os papéis do Scrum, segundo Schwaber (2007), são definidos como: Equipe de desenvolvimento, composta por um grupo de desenvolvedores multifuncionais e capazes de se auto-organizar; *Product owner*, responsável por liderar a gestão do projeto e iterações, buscando maximizar o valor e controlar os riscos; e, *Scrum Master***,** responsável pela aderência do processo de software, tratar os obstáculos para que a equipe realize o planejado e não permitir priorizações em favor da produtividade que comprometam a qualidade.

2.2 eXtreme Programming

O eXtreme Programming (XP) é o método ágil mais popular (Grigoriev e Yevtushenko 2003), que caracteriza-se por estabelecer 13 práticas que definem como se deve trabalhar. Conforme Jeffries (2012): *"**Toda a equipe**, clientes e desenvolvedores devem trabalhar juntos; usando **planejamento do jogo** e definindo **pequenas liberações** de software que passam por **testes do cliente**. Para cada funcionalidade busca-se um **projeto simples**; a equipe é organizada em **pares** de desenvolvedores os quais realizam o **desenvolvimento dirigido por testes;** para manter a simplicidade e limpeza do*

*produto utiliza-se o **refactoring.** As **metáforas** são incentivadas para manter a união da equipe e ajudar a memorizar assuntos. A **integração contínua** auxilia na manutenção da qualidade e reduz o tempo de montagem final do produto. Deve ser incentivada a **propriedade coletiva** do código, e manter um **padrão de codificação**. Deve-se manter um **ritmo sustentável** para manter as práticas e qualidade do produto."*

Embora o XP pregue o seguimento de todas suas práticas, é possível iniciar com um subconjunto de atividades. Auer e Miller (2001) denominaram 6 práticas essências, a saber: Pequenas liberações e iterações; Desenvolvimento dirigido por testes (TDD); Re-fatoração; Integração contínua; Programação em pares; e Planejamento do jogo.

2.3 OpenUP

O OpenUP nasceu de uma iniciativa da IBM para propagar parte do *Rational Unified Process* (RUP) de forma livre. Os princípios do manifesto ágil guiam este processo, buscando a colaboração entre os indivíduos da equipe. Assim como o RUP, o OpenUP segue o ciclo de vida iterativo e incremental, sendo suas fases também organizadas em iniciação, elaboração, construção e transição. Entretanto o OpenUP possui um conjunto simplificado e reduzido de atividades, papéis, tarefas, produtos de trabalho e guias.

O esforço em um projeto OpenUP é organizado em micro-incrementos que representam pequenas unidades de trabalho, constante e medidos em horas ou dias, o que fornece um *feedback* rápido que possibilita decisões durante cada iteração. Dentro do planejamento da iteração, na tarefa de Gerenciar a Iteração, o OpenUP sugere a utilização de reuniões diárias no guia de orientação para colaboração da equipe, prática conhecida no SCRUM como as reuniões diárias de acompanhamento (OpenUP 2011).

3. Processo desenhado

Inicialmente foram analisados os problemas sob a ótica do cliente, onde se identificou a necessidade de reduzir o tempo das entregas. Visando aumentar a produtividade, calculou-se o PCE (*Process Cycle Efficiency*) para os projetos do último ano, obtendo 14% para o ciclo total. O PCE é calculado com a divisão do tempo de valor agregado pelo tempo total decorrido (George et al. 2004).

Visando eliminar os gargalos e otimizar o processo, elaborou-se um novo modelo dividido em duas fases: Concepção (destinada à identificação dos ativos, análise do processo e requisitos) e Desenvolvimento de Software. O início do processo é a chegada de demandas por soluções de negócio, que são a entrada para a Fase de Concepção. Em seguida, vem a Fase de Desenvolvimento, a qual é seguida pelos processos de gestão de liberação, comunicação aos usuários, serviço de gestão de acessos, treinamento, e a transição para a equipe de suporte e monitoração.

Os papéis e responsabilidades dos envolvidos na Fase de Concepção são:

- **Líder de Processo de Negócio (Product Owner):** realiza a pontuação do valor do negócio, prioriza as atividades de sua equipe, leva os assuntos para aprovação de investimentos junto ao Grupo de Integração de Processo (GIP) e à comissão do respectivo processo de negócio;
- **Analistas de Processo:** identificam e analisam a aderência dos ativos de software existentes, podendo resultar na configuração e liberação de acesso aos usuários; mapeiam o processo de negócio e suas possíveis melhorias, utilizando a ferramenta Aris; quando necessário o desenvolvimento, elaboram a

especificação de requisitos, os critérios de aceitação e realizam a validação com o usuário chave, utilizando *Enterprise Architect* e *Microsoft Word*;

- **Usuários Chave:** responsável pela execução do processo de negócio, mas também valida os requisitos e esclarece dúvidas referentes ao processo de negócio atual e análise de mudanças. Muitas vezes apoia o processo de implantação, treinamento e mudanças organizacionais;
- **Gerente de Projeto:** responsável pelo método de trabalho durante a concepção, mantem o plano dos projetos, apoia a avaliação de capacidade e alocação de recursos, atualiza o plano de mitigação de riscos, problemas e comunica os gestores de TI. Também possui formação de *Scrum Master;*
- **Arquiteto de Software:** o arquiteto de software é envolvido para definir a arquitetura de demandas com tamanho maior que 400 horas. Nestes casos, o arquiteto deve elaborar e documentar a arquitetura, planejando, opcionalmente, uma *sprint* de desenvolvimento para validar a arquitetura. Para demandas menores utiliza-se de um guia de solução técnica e padrões da organização para orientação da equipe de desenvolvimento. O arquiteto também apoia a equipe de infraestrutura no dimensionamento, análise de impacto e custos, onde pode ser necessário realizar simulações, através de cenários de testes de carga e stress;
- **Analista de Sistemas:** o analista de sistemas de desenvolvimento é envolvido na Fase de Concepção para realizar a estimativa (pontos de função ou tabela padrão de estimativas para extensão do ERP). Ao envolver a equipe responsável pela execução na estimativa observa-se um maior comprometimento com a solução e com os prazos, podendo, ainda, antecipar problemas técnicos.

A Fase de Concepção é apresentada nas Figuras 1 (Planejamento da Concepção) e 2 (Execução da Concepção).

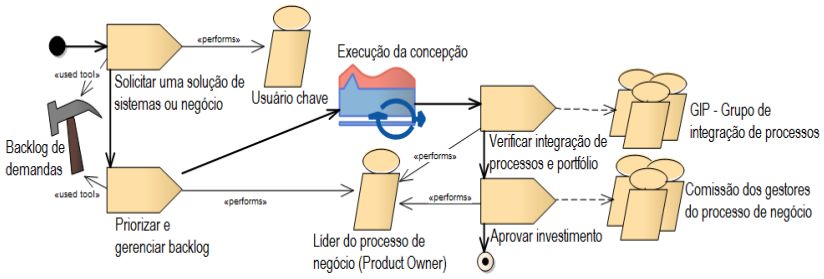

Figura 1 – Fluxo de Planejamento da Fase de Concepção.

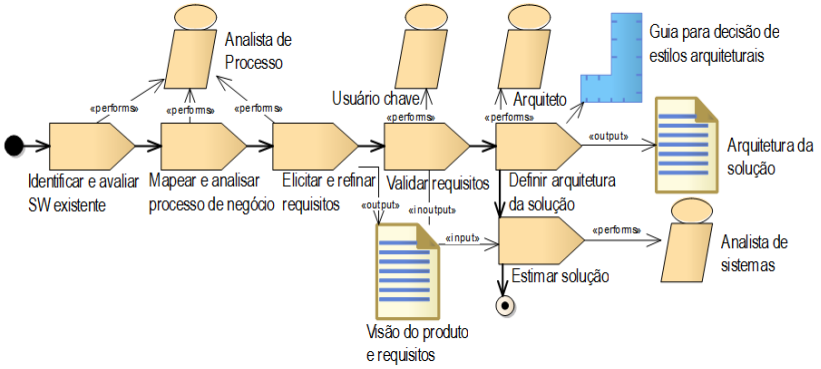

Figura 2 – Fluxo de Execução da Fase de Concepção.

Após a execução da Fase de Concepção, o líder do processo aprova os investimentos junto ao GIP e à sua comissão. Estas comissões estão organizadas em oito grupos, onde cada grupo é responsável por processos de negócios chave da organização. Estas comissões são formadas por gerentes relacionados ao processo de negócio das diversas empresas do grupo. Mediante a aprovação, a visão e os requisitos entram no *backlog* da equipe de desenvolvimento do processo de negócio correspondente. Os papéis envolvidos na Fase de Desenvolvimento são:

- ***Product owner*:** papel geralmente representado pelo líder do processo de negócio que representa o cliente na priorização do *backlog* e tarefas;
- **Equipe de desenvolvimento:** atualmente todas as equipes de desenvolvimento possuem dois papéis distintos: o Analista de Sistemas responsável pela especificação de caso de uso, elaboração dos cenários de testes e definição de domínio de dados (classe e/ou tabelas); e o Programador, que detalha o diagrama de domínio, realiza o desenvolvimento dos testes unitários automatizados, e a construção do software. Em algumas células pequenas, uma mesma pessoa realiza ambos os papéis, mas esta condição pode apresentar riscos principalmente na atividade de testes;
- ***Scrum Master*:** responsável pelo planejamento da alocação dos recursos de desenvolvimento, treinamentos, orientação do processo, e tratativas dos problemas para realização das *sprints*.

A Fase de Desenvolvimento se encontra ilustrada nas Figuras 3 e 4.

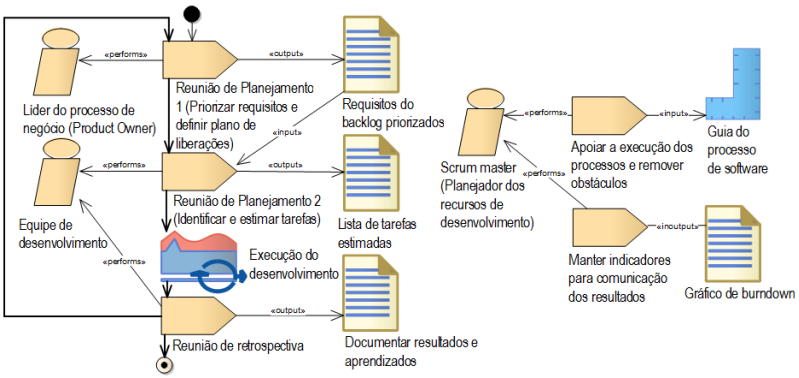

Figura 3 – Fluxo de planejamento da Fase de Desenvolvimento.

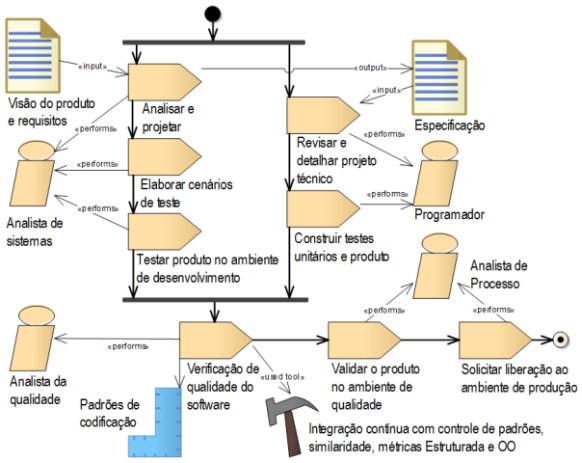

Figura 4 – Fluxo de execução da Fase de Desenvolvimento.

4. Implantação do processo

O cenário de implantação deste processo foi a WEG, uma multinacional brasileira que possui mais de 23.800 colaboradores, sendo uma das maiores fabricantes de equipamentos elétricos do mundo. A WEG atua nas áreas de comando e proteção, variação de velocidade, automação de processos industriais, geração e distribuição de energia e tintas e vernizes industriais (WEG 2011).

O processo de desenvolvimento aqui apresentado destina-se ao desenvolvimento de sistemas administrativos, no qual atuam cerca de 70 profissionais. Após implantação de um novo ERP, diversos artefatos são gerados para revisão da arquitetura de processos e sistemas. A manutenção desta documentação envolve uma grande carga de trabalho, cerca de 300 profissionais. Após a implantação, é natural reduzir as equipes, onde a robustez do processo de implantação do ERP passa a ser um desperdício.

O processo apresentado foi aplicado em um projeto piloto de 120 pontos de função, com uma equipe de 2 analistas, 3 programadores, 1 arquiteto e 1 scrum master. No início, o processo foi implantado somente na equipe de desenvolvimento, responsável pela análise, construção e testes, e depois foi adaptado para suportar a equipe de processo, responsável pela concepção. A posteriori foram realizados ajustes de capacidade e dedicou-se uma equipe para o processo de negócio de vendas e pós-vendas. Após 3 meses de trabalho desta equipe, decidiu-se expandir o processo, criando outras equipes para cada processo de negócio. Atualmente são 7 equipes. A equipe precursora está em sua *sprint* 27, em pouco mais de 2 anos de execução do processo.

No processo anterior, as atividades relacionadas ao desenvolvimento apresentavam um PCE em torno de 40%. Após a implantação, o PCE da fase de desenvolvimento ficou em 77%. Já o PCE total (para ambas as fases) era de 14% e passou para cerca de 30%. Para alcançar estes resultados foram realizadas as ações descritas a seguir.

4.1 Agilidade na comunicação com controles de mudanças

Ao executar o processo, pôde-se perceber que grupos de usuários não podem exercer o papel do *product owner,* pois isto pode causar desperdícios nas reuniões da equipe. O *product owner* deve ser uma pessoa responsável e capacitada para realização de suas atividades e com envolvimento integral no projeto. Outro ponto importante é entregar valor o quanto antes para o cliente. Para isso, deve-se guiar a equipe com o propósito de evitar planejamentos de *sprints* longas no início do projeto, entregando pequenos pedaços do produto que possam trazer retorno ao negócio o mais breve possível.

Os métodos ágeis aceitam a natureza imprevisível do software, priorizando o diálogo e a negociação, buscando realizar mudanças de requisitos de forma ágil. Mas para isso, como evidenciado por Catunda et al. (2011), é fundamental o apoio de ferramentas para que a gestão de requisitos e mudanças ágeis não prejudiquem os processos de software. A adoção da prática *timebox* tem evitado esforços com replanejamentos, com produto entregue e implantado ao final de cada *sprint*. O *timebox* é uma prática que busca incitar o cumprimento dos objetivos, através de prazos bem definidos para as iterações, tarefas, reuniões de retrospectiva, planejamento e diárias. Dentro de cada iteração na fase desenvolvimento, as práticas de *timebox* e gestão mediante a troca de requisitos (ou casos de uso) tem atendido o controle de mudanças. A troca de requisitos é uma prática que aceita a mudança com o cumprimento do

timebox, onde para entrar uma nova solicitação ou mudança é preciso retirar outra(s) de mesmo tamanho. Esta prática favorece o cumprimento dos prazos, evita replanejamentos e entrega os itens de maior valor agregado antes em produção.

O *backlog* precisa ser cuidadosamente gerenciado, pois este já foi orçado e aprovado em comissão, porém a data firme da entrega é gerada somente ao planejar a *sprint* na fase de desenvolvimento. As interrupções devido à correção de bugs ou à elaboração de orçamentos tem afetado a capacidade de trabalho da equipe, o que leva a planejamentos que usam em média 80% da sua capacidade. Para cada liberação utiliza-se o recurso de *baseline* e registro de liberação vinculado aos objetos e código fonte (recursos de change *request* SAP e *tags)*.

4.2 Autonomia e treinamento

Para acelerar o desenvolvimento de pequenas demandas, sem o desperdício de tempo no aguardo de aprovações por comissões, foi definida uma alçada onde as demandas seriam aprovadas diretamente pelo Líder de processo de negócio. Definiu-se o procedimento de sempre registrar estas pequenas demandas (menor que 100 horas) e enviar para prestação de contas na comissão e evitar o mau uso deste mecanismo.

Um importante passo para implantação das práticas ágeis de planejamento foi a capacitação de 3 colaboradores com o *Certified Scrum Master* e a definição clara da responsabilidade destes no processo de software. Outros fatores foram: o treinamento de novos colaboradores, a reciclagem periódica e as auditorias para verificar o cumprimento das práticas. O *Scrum Master* tem o papel de presar pela execução do processo e suas práticas, mas mantendo o objetivo maior da *sprint*. A reunião de retrospectiva é uma importante prática que captura as oportunidades para melhoria contínua dos processos de software. No ambiente corporativo as lições aprendidas devem ser compartilhadas entre as diversas equipes. Esta ligação com as demais equipes tem sido realizada pelos *Scrum Master, e, também*, através de registros em um repositório comum também para comunicação dos gestores.

4.3 Investimentos no controle da qualidade do produto

As práticas essenciais do XP têm sido aplicadas entre as equipes, porém algumas em menor escala (um terço da equipe), tais como a Programação em Pares e o Planejamento do Jogo. Tem se investido na formação das equipes e na propriedade coletiva do software. Foram elaborados guias de desenvolvimento e implantadas ferramentas para verificação de padronização de código através de métricas para medir tanto o desenvolvimento estruturado como orientadas a objetos (nas plataformas ABAP, Java e Microsoft .NET). Investiu-se em ferramentas para congelar e carregar dados para os cenários de testes, assim como nas ferramentas para automatização de testes unitários e funcionais e ferramentas de integração contínua.

4.4 Reuso de processos e sistemas

A gerência de processos e priorização das necessidades era realizada pela aprovação na comissão dos gestores do processo de negócio, entretanto, como existem oito comissões específicas, a aprovação tinha uma visão não integrada. Para resolver este problema, criou-se o Grupo de Integração de Processos (GIP) para reduzir o número de demandas de pequeno valor e reduzir o retrabalho. Este grupo avalia a integração entre os processos de negócio, arquitetura de processos e sistemas, analisa os relacionados e dependências entre as necessidades e os investimentos.

5. Conclusões

Processos bem definidos são essenciais para sustentar o desenvolvimento, garantindo a qualidade dos produtos das organizações. As práticas dos métodos ágeis são facilmente combináveis aos métodos clássicos e trazem ganhos principalmente na comunicação e comprometimento da equipe. A experiência do Scrum master é essencial para fazer a diferença na condução da equipe, zelando pelo processo e qualidade do produto. Além disto, o Scrum aprimora a comunicação das equipes e pode ser combinado com os papéis clássicos do OpenUP ou RUP, sem colocá-los em risco. Os próximos passos são: implantação de um projeto de melhoria baseado em CMMI ou MPS-BR; evolução da gestão de portfólio de projetos e demandas, gestão das necessidades de negócio com seus requisitos e mudanças.

Agradecimentos

Os autores agradecem a Católica de Santa Catarina pelo auxílio Financeiro, a WEG e aos colegas André Castoldi, Luiz R. Ronchi e Tatiana Baruffi.

6. Referências Bibliográficas

Auer, K.; Miller, R. (2001), Extreme Programming Applied: Playing to Win. Addison-Wesley Professional.

Catunda, E., Nascimento, C., Cerdeiral, C., et al. (2011). Implementação do Nível F do MR-MPS com Práticas Ágeis do Scrum em uma Fábrica de Software. In: X Simpósio Brasileiro de Qualidade de Software, Curitiba-PR.

George, M. L., Maxey, J.; Rowlands, D. T.; Upton, M (2004). The Lean Six Sigma Pocket Toolbook: A Quick Reference Guide to Nearly 100 Tools for Improving Process Quality, Speed, and Complexity. McGraw-Hill, 288p.

Grigoriev, P. A., Yevtushenko, S. A. (2003), Elements of an Agile Discovery Environment. Springer, Berlin.

Esteves, J., Jorge, J. (2001), Análise Comparativa de Metodologias de Implementação de SAP. In: Conference of Associação Portuguesa de Sistemas de Informação – APSI, Évora, Portugal.

Fuggetta, A. (2000), Software Process: A Roadmap. In: International Conference on Software engineering, Limerick, Ireland. New York: ACM, 2000, p. 25-34.

Jeffries, R (2011), What is Extreme Programming? Disponível em: http://xprogramming.com/what-is-extreme-programming/. Acessado: 24/01/2012.

Kruchten, P. (2003), The Rational Unified Process: An Introduction. Reading, MA: Addison-Wesley Longman, Inc, 2003. 336 p.

OpenUP (2011), Open Unified Process. Disponível em: http://epf.eclipse.org/wikis/openup/. Acessado 24/01/2012.

Rosenberg, D., Collins-Cope, M., Stephens, M. (2005). Agile Development with the ICONIX Process: People, Process and Pragmatism. Apress.

Schwaber, K., (2007), The Enterprise and Scrum. Redmond - WA: Microsoft Press.

WEG, 2011. “History - The Founders”, Disponível em: http://www.weg.net/us/About-WEG/History/The-Founders. Acessado em janeiro 2012.