O Uso de Extreme Programming em uma Organização CMM Nível 2

Renata Endriss Carneiro Campelo[1]
Fábio Gomes Silva[2]
Hermano Perrelli de Moura[3]

[1,3] Universidade Federal de Pernambuco
Centro de Informática
{recc, hermano}@cin.ufpe.br

[1,2] Centro de Estudos e Sistemas Avançados do Recife
{renata.endriss, fgs}@cesar.org.br

Resumo

Recentemente a comunidade de software vem se deparando com um grupo de novas metodologias de desenvolvimento de software, classificadas como "Metodologias Ágeis", que possui como principais características valores contrastantes com modelos de qualidade reconhecidos mundialmente. Os valores são: Indivíduos e iterações sobre processos e ferramentas; Software funcionando sobre documentação compreensiva; Colaboração do cliente sobre negociação de contrato; Resposta à mudança sobre seguir um plano.

Este artigo relata o uso em um projeto da metodologia ágil Extreme Programming (XP) numa organização alinhada aos processos definidos pelo nível 2 do Capability Maturity Model (CMM), além de ressaltar os benefícios e problemas encontrados pelo uso destas duas abordagens.
PALAVRAS-CHAVE: Capability Maturity Model, CMM, Extreme Programming, XP.

Abstract

A new software development group has becoming popular inside the software comunity, it's the "Agile Methodologies" group, wich has as principal features, values that act against the quality models recognized around the world. These values are: Individuals and iteractions over processes and tools; Working software over comprehensive documentation; Customer collaboration over contract negotiation; Responding to change over following a plan.

This article describes the use of the agile methodology Extreme Programming (XP) in an organization implementing the level 2 of the Capability Maturity Model (CMM), describing the benefits and problems originated due to the use of these approaches.
KEY-WORDS: Capability Maturity Model, CMM, Extreme Programming, XP.

1. Introdução

Recentemente a comunidade de software vem se deparando com um grupo de novas metodologias de desenvolvimento de software, classificadas como "Metodologias Ágeis" [1], o qual define os seguintes valores para o desenvolvimento de software [5]:

1. **Indivíduos e iterações** sobre processos e ferramentas.
2. **Software funcionando** sobre documentação compreensiva.
3. **Colaboração do cliente** sobre negociação de contrato.
4. **Resposta à mudança** sobre seguir um plano.

Além dos valores, as metodologias ágeis definem princípios, sendo importante ressaltar [5]: Satisfazer o cliente através de entrega contínua e rápida de software de valor;

Mudanças em requisitos devem ser bem vindas, mesmo que em momentos tardios do desenvolvimento; Cliente e desenvolvedores devem trabalhar juntos diariamente no projeto; Deve-se utilizar a comunicação face-a-face como o método mais eficiente e efetivo de fluxo de informações; Deve-se utilizar como medida principal de progresso do projeto o software funcionando; Deve-se assumir simplicidade para todos os aspectos do projeto.

Algumas das metodologias que fazem parte deste grupo são: Extreme Programming (XP) [2] e SCRUM [4], sendo XP a mais conhecida e utilizada.

Em paralelo a disseminação das metodologias ágeis, os investimentos em qualidade de software vêm aumentando a cada ano. A última pesquisa conduzida pelo Ministério de Ciência e Tecnologia sobre o setor de software brasileiro, indica um crescimento no uso de modelos de qualidade como o Capability Maturity Model (CMM) [6].

Mark Paulk, um dos elaboradores do CMM, analisou como XP se comporta em relação ao CMM [7]. Como conclusão, afirma que XP é um processo documentado e disciplinado e que atende a vários processos definidos pelo CMM e que, mesmo havendo diferenças entre eles, ambos podem conviver em um mesmo ambiente: "XP possui boas práticas de engenharia que podem funcionar bem com o CMM e outros métodos altamente estruturados. A chave é considerar, cuidadosamente, as práticas de XP e implementá-las no ambiente correto". Além disto: "As práticas de XP podem ser compatíveis com as práticas do CMM (metas ou áreas chave de processo), mesmo que elas não contemplem completamente o modelo".

Este artigo relata a experiência de usar XP em um projeto de uma organização alinhada ao nível 2 do CMM, analisando dificuldades encontradas, adaptações necessárias e impacto sobre o projeto.

2. Extreme Programming

XP é uma metodologia de desenvolvimento baseada em valores, princípios e práticas. Os quatro valores definidos por XP são [3]:

- Comunicação: a comunicação da equipe deve ser eficaz.
- Simplicidade: a solução mais simples deve ser a selecionada para resolver o problema.
- *Feedback*: o trabalho da equipe deve ser orientado ao *feedback* do cliente e do software funcionando.
- Coragem: as pessoas assumam responsabilidade sobre o trabalho a ser realizado.

Para guiar escolhas, sustentar os valores e torná-los mais concretos, XP define princípios. Alguns deles são: *feedback* rápido, assumir simplicidade, mudança incremental, aceitar mudanças e qualidade do trabalho [3].

Para concretizar os princípios, XP define doze práticas: planejamento do jogo, *releases* pequenos, uso de metáforas para o projeto do software, design simples, testes automáticos, *refactoring*, programação em pares, propriedade coletiva do código, integração contínua, 40 horas semanais de trabalho, cliente no local, padrão de codificação [3]. XP sugere que as práticas sejam adotadas da forma mais fiel possível, embora afirme que ajustes devam ser feitos de acordo com a realidade das organizações, desde que os valores sejam respeitados [8].

XP não é adequada a todos tipos de projeto. É aconselhável que seja utilizada em equipes pequenas, com aproximadamente dez desenvolvedores, onde o ambiente de desenvolvimento facilite testes e integração [3].

3. O Nível 2 do CMM

O nível 2 do CMM foca em questões de gerenciamento, objetivando a repetição de casos passados de sucesso. No nível 2, são definidas metas e práticas para atingi-las para as seguintes áreas chave de processos (KPA) [6]:

- Gerenciamento de requisitos: objetiva estabelecer um entendimento comum dos requisitos entre o cliente e o projeto.
- Planejamento de projetos: o propósito é estabelecer para o desenvolvimento do software e acompanhamento do projeto.
- Acompanhamento de projetos: o objetivo desta KPA é prover visibilidade sobre o progresso para que ações corretivas possam ser tomadas quando necessário.
- Gerenciamento de subcontrato: esta KPA visa selecionar um subcontratado qualificado e gerenciá-lo eficazmente.
- Garantia da qualidade: prover a alta gerência com visibilidade adequada sobre os processos utilizados e artefatos construídos.
- Gerenciamento de configuração: visa estabelecer e manter a integridade dos artefatos construídos.

4. Usando XP numa Organização CMM 2

4.1 A Organização

A organização em que se deu o experimento é o C.E.S.A.R – Centro de Estudos e Sistemas Avançados do Recife [13], o qual possui um processo de desenvolvimento de software padrão, chamado ProSCes, elaborado para estar completamente alinhado ao nível 2 do CMM. Grande parte deste processo é utilizada por projetos do C.E.S.A.R, classificados como nível 2 do CMM em avaliação formal realizada em junho de 2003.

4.2 O Projeto

O projeto fruto deste relato foi firmado sobre um contrato de 36 meses e envolve atualmente 19 pessoas. Ele engloba desenvolvimento de novas funcionalidades e manutenção, utilizando a linguagem de programação Java. O cliente é um órgão público do governo federal.

O projeto enfrenta algumas dificuldades devido a características próprias. Primeiramente, o projeto enfrenta a falta de um gestor do projeto no lado cliente, o que acarreta no não estabelecimento de um ponto focal, com responsabilidade e autoridade para tomar decisões em nome do cliente. O problema gerado por este fato é que as decisões não podem ser tomadas na velocidade necessária, mudanças são difíceis de serem negociadas e compromissos difíceis de serem estabelecidos. Esta situação foi agravada em 2003 devido à troca de presidentes, o que provocou mudanças significativas nas camadas gerenciais do cliente, que interagiam com o projeto.

Além disto, os usuários do sistema são representantes de diferentes áreas da empresa contratante, que possuem interesses divergentes. Estes usuários, responsáveis por definir requisitos e prioridades, e validar os produtos gerados pelo projeto, entram em conflitos que muitas vezes, comprometem o progresso do projeto.

Outro fator que torna o gerenciamento do projeto não trivial se deve ao fato de o projeto ser de desenvolvimento e manutenção ao mesmo tempo. Por um lado, os usuários cobram que todos os defeitos sejam corrigidos com a máxima urgência, por outro, o Patrocinador do projeto cobra que as novas funcionalidades sejam entregues nas datas acordadas.

A decisão de utilizar XP no projeto se deu objetivando aumentar a satisfação do cliente, a produtividade, comprometimento e motivação da equipe, aumentar a eficácia das estimativas, além de diminuir os grandes riscos do projeto e facilitar o planejamento. Os aspectos de XP que mais contribuíram para apoiar a decisão de utilizá-lo visando o alcance destes objetivos foram os *releases* curtos de software de valor, as iterações curtas de tamanho fixo, as reuniões diárias e as estimativas realizadas pelos próprios desenvolvedores.

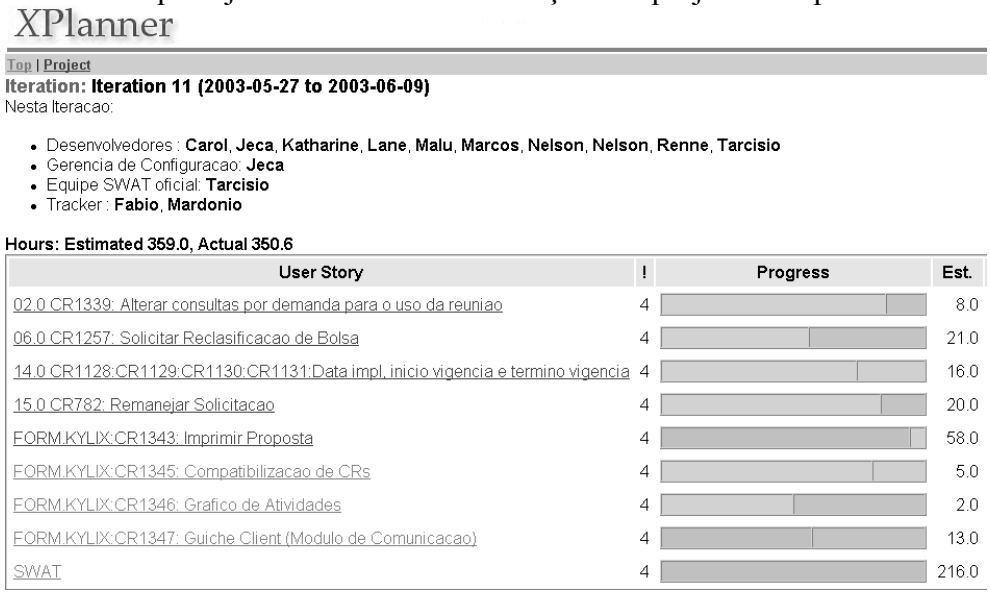
4.3 O Processo do Projeto

O processo adotado pelo projeto mescla práticas do nível 2 do CMM e de XP. Este processo foi definido a partir das práticas de XP adotadas pelo projeto e um diagnóstico de aderência ao nível 2 do CMM realizado pelo grupo de garantia da qualidade do C.E.S.A.R. A seguir são descritas as características da aplicação do processo ao projeto.

4.3.1 Uso de XP no projeto

Práticas Adotadas: O projeto não implementou a risca as doze práticas de XP. Dentre as práticas adotadas estão:

- Jogo do planejamento – O planejamento detalhado do projeto segue esta prática de XP: há um planejamento macro dos *releases*, os quais são detalhados em termos de histórias antes de iniciar. Cada release é composto de iterações de duas semanas, as quais também são detalhadas apenas antes de iniciar, em termos de tarefas para realizar as histórias. Histórias e tarefas são selecionadas para os *releases* e iterações de acordo com a prioridade do cliente e o esforço necessário para realizá-las é estimado pela equipe de desenvolvimento e registradas na ferramenta XPlanner [12]. A figura Figura 1 exibe o planejamento de uma das iterações do projeto no Xplanner.


XPlanner

Top | Project

Iteration: Iteration 11 (2003-05-27 to 2003-06-09)

Nesta Iteracao:

- Desenvolvedores : **Carol**, **Jeca**, **Katharine**, **Lane**, **Malu**, **Marcos**, **Nelson**, **Nelson**, **Renne**, **Tarcisio**
- Gerencia de Configuracao: **Jeca**
- Equipe SWAT oficial: **Tarcisio**
- Tracker : **Fabio**, **Mardonio**

Hours: Estimated 359.0, Actual 350.6

User Story	!	Progress	Est.
02.0 CR1339: Alterar consultas por demanda para o uso da reuniao	4		8.0
06.0 CR1257: Solicitar Reclasificacao de Bolsa	4		21.0
14.0 CR1128:CR1129:CR1130:CR1131:Data impl, inicio vigencia e termino vigencia	4		16.0
15.0 CR782: Remanejar Solicitacao	4		20.0
FORM.KYLIX:CR1343: Imprimir Proposta	4		58.0
FORM.KYLIX:CR1345: Compatibilizacao de CRs	4		5.0
FORM.KYLIX:CR1346: Grafico de Atividades	4		2.0
FORM.KYLIX:CR1347: Guiche Client (Modulo de Comunicacao)	4		13.0
SWAT	4		216.0

Figura 1 - O planejamento do jogo no XPlanner

- *Releases* pequenos – A entrega do software foi planejada para ocorrer incrementalmente, sendo cada pedaço do sistema entregue em um prazo médio de um mês e meio.
- *Refactoring* – Mesmo tendo uma arquitetura estabilizada para todo o projeto, ao receberem tarefas de desenvolvimento, os desenvolvedores analisam se é possível melhorar a forma como o software foi implementado. Recentemente uma grande reestruturação na arquitetura do software foi realizada para aumentar a performance do sistema.
- Padrão de codificação – O projeto segue o padrão de codificação definido pelo C.E.S.A.R para desenvolvimento em Java.
- Propriedade coletiva – Todos da equipe podem alterar qualquer arquivo que compõe o software.
- 40 horas por semana – O C.E.S.A.R possui um banco de horas para que as horas trabalhadas a mais possam ser compensadas com descanso, conforme acordado com o gerente do projeto e política da organização. Mesmo com o banco de horas, a equipe trabalha usualmente por volta de 40 horas por semana.
- Programação em pares – Esta prática não é adotada como sugere XP: toda a equipe trabalhar todo o tempo em pares. No projeto, a programação em pares é adotada para novatos e pessoas pouco experientes na tecnologia adotada, os quais devem trabalhar com um membro mais experiente da equipe até que estejam aptos a realizar suas atividades plenamente. Atualmente, a possibilidade de estender esta prática para outras situações está sendo avaliada.
- Integração contínua – Diariamente o *build* do software é gerado automaticamente pela ferramenta *Ant* [9], adotada pelo projeto. Quando acionada, a ferramenta gera o *buid* do sistema, um arquivo informando os arquivos alterados e as requisições de mudança (CRs) fechadas na ferramenta de controle de mudança do projeto (o Bugzilla [10]).

Práticas não Adotadas: dentre as práticas de XP não adotadas pelo projeto, estão:

- Testes automáticos – Apesar de acreditar nos benefícios dos testes automáticos, esta prática ainda não foi adotada no projeto. Esta decisão se deu pelo fato de a adoção de XP ter ocorrido quando grande parte do código já estava implementada. Para obter todos os benefícios dos testes automáticos, todo o código deveria possuir testes, inclusive este código anteriormente produzido. Implementar todos estes testes traria um *overhead* muito grande para a equipe.
- Cliente no local – O cliente não se localiza no mesmo estado em que o projeto foi desenvolvido. Para representar o cliente dentro do projeto e para que o trabalho de entendimento dos requisitos e validação dos artefatos gerados fosse efetivo, parte da equipe passou a trabalhar no local do cliente. Esta equipe é responsável por especificar os requisitos, validá-los e conduzir testes de aceitação.
- Metáforas – O projeto utiliza o paradigma de programação orientação a objetos, o que torna o projeto de software perto da realidade do cliente. Devido a isto, o uso de metáforas não foi adotado no projeto.
- Projeto simples – Pelo fato de a equipe e o software a ser construído serem relativamente grandes, optou-se, no início do projeto, por definir uma arquitetura estável, que facilitasse a inserção de novas funcionalidades, manutenção, componentização e independência entre as regras de negócio e dados.

4.3.2 Aderência de XP ao CMM

O diagnóstico realizado pelo grupo de garantia da qualidade do C.E.S.A.R para verificar conformidade ao nível 2 do CMM apontou alguns problemas no processo utilizado pelo projeto, ou seja, XP.

As KPAs gerenciamento de requisitos, planejamento de projetos de software, acompanhamento de projetos de software e gerenciamento de configuração de software possuíam várias práticas contempladas pelo processo do projeto, no entanto, algumas práticas não eram abordadas ou eram contempladas de forma diferente. O processo não abordava a KPA garantia da qualidade. A KPA gerência de subcontratação não foi implementada no projeto, pois o projeto não subcontrata.

- Aderência a KPA Planejamento de Projetos

O projeto implementava esta KPA através dos planos de *release* e iteração, estimativas de esforço para as histórias e tarefas e elaboração do cronograma de *releases* e iterações.

Algumas práticas desta KPA não implementadas eram: identificação e avaliação de riscos, estimativa de recursos críticos computacionais, planejamento das instalações e ferramentas de apoio, participação da alta gerência para firmar compromissos com grupos externos à organização. Além disto, os planos elaborados não contemplavam o conteúdo de um plano conforme sugerido pelo CMM.

- Aderência a KPA Acompanhamento de Projetos

O acompanhamento do projeto era bastante forte devido aos *releases* e iterações curtas e às reuniões diárias com toda a equipe. Esta KPA era implementada com a atualização dos planos de *release* e iteração de acordo com mudanças no projeto, reuniões diárias para a comunicação das mudanças aos grupos afetados e acompanhamento das atividades técnicas. Além disto, o projeto possuía acompanhamento das estimativas e do cronograma de atividades, onde a cada reunião de iteração se verificava o que foi cumprido na iteração passada e se planeja as atividades para a próxima iteração.

Aspectos desta KPA não contemplados eram: acompanhamento de riscos, de recursos críticos computacionais, participação da alta gerência nas mudanças de compromissos com grupos externos, revisões em marcos.

- Aderência a KPA Gerenciamento de Requisitos

Os requisitos (ou histórias, pela nomenclatura de XP) eram identificados e documentadas no Plano de *Release*. Nas iterações, as histórias eram detalhadas em cartões, um para cada história. Mudanças eram controladas para os *releases* e iterações: o cliente poderia inserir, excluir ou alterar histórias de uma iteração e/ou release, desde que a mudança não excedesse a capacidade (produtividade estimada) da equipe.

As práticas desta KPA não contempladas eram: requisitos não técnicos, não funcionais e critérios de aceitação não eram documentados, não havia como identificar a versão e as mudanças realizadas sobre os requisitos documentados.

- Aderência a KPA Gerenciamento de Configuração de Software

Devido à natureza de desenvolvimento de novas funcionalidades e manutenção, o projeto enfrentava vários problemas no controle de versão do sistema. Através de *releases* freqüentes e integração contínua, XP apóia o estabelecimento de *baselines* e controle de mudanças a eles, no entanto, não havia um plano de gerência de configuração do projeto que identificasse estas *baselines*, que estabelecesse o procedimento de mudanças, as versões, o procedimento para realização de *release*, auditorias nas *baselines*.

4.3.3 Adaptações ao processo do projeto

- Adaptações para atender a KPA Planejamento de Projetos

Um plano do projeto foi elaborado contemplando o conteúdo de um plano XP, de um plano CMM e necessidades do C.E.S.A.R. Desta forma, o plano do projeto passou a definir: propósito, escopo, objetivos, descrição do cliente, organograma do projeto (com papéis da equipe e do cliente), responsabilidades, procedimentos, padrões e métodos, artefatos a serem entregues, instalações e ferramentas, treinamentos, planejamento de *releases*, referencia a ferramenta XPlanner (que define o planejamento das próximas iterações), estimativas e planilha de riscos.

Recursos críticos computacionais não foram identificados para o projeto. Uma planilha de riscos foi adotada. Nesta planilha os riscos são identificados, priorizados e planos de contingência e mitigação são definidos. O plano do projeto foi submetido à análise e aprovação dos clientes e alta gerência do C.E.S.A.R, para que os compromissos do projeto fossem acordados e firmados.

- Adaptações para atender a KPA Acompanhamento de Projetos

Sempre que o projeto sofre mudanças que impactam prazo, esforço e custo planejados, o plano precisa ser alterado e submetido a uma nova revisão pelo cliente e alta gerência. Além disto, um relatório mensal de progresso do projeto é enviado para a alta gerência do C.E.S.A.R e para o cliente. Este relatório contém: dependências, riscos, atividades planejadas e realizadas, métricas. Desta forma, a alta gerência participa das mudanças aos compromissos com grupos externos e os riscos são acompanhados e reportados. Revisões formais foram agendadas para serem conduzidas com representantes do cliente a cada *release*.

- Adaptações para atender a KPA Gerenciamento de Requisitos

Os requisitos não técnicos e critérios de aceitação foram documentados no plano do projeto. Além disto, foi criado um artefato, chamado, Documento de Requisitos que descreve os requisitos funcionais e não funcionais macros do projeto. Para alterar os requisitos deste documento, uma requisição de mudança deve ser registrada na ferramenta de controle de mudanças adotada, avaliada e aprovada pelo gerente do projeto.

Como citado anteriormente, o documento de requisitos possui os requisitos de alto nível do projeto, assim como o plano de *releases* de XP. O detalhamento dos requisitos é tratado através de histórias e tarefas, como sugere XP. A flexibilidade para realizar alterações não foi diminuída, apenas há um controle maior através das solicitações de mudanças formais para alterar o escopo macro do projeto, visando um maior controle sobre as mudanças de custo, prazo e esforço. As alterações das histórias nas iterações são realizadas conforme sugere XP: podem e devem ser realizadas de acordo com necessidades do cliente e/ou técnicas, desde que a capacidade da equipe seja respeitada.

- Adaptações para atender a KPA Gerenciamento da Configuração

Um plano de gerência da configuração foi elaborado para o projeto descrevendo as *baselines*, itens de configuração, CCB (comitê que avalia as mudanças).

Para não tornar custoso o processo de gerência de configuração dentro do projeto, foram adotadas ferramentas *freeware* para controle de versão (CVS [11]) e de mudanças (Bugzilla). Além disto, o gerente de configuração do projeto é um membro da equipe de desenvolvimento e dedica uma média de quatro horas diárias a esta atividade.

- Adaptações para atender a KPA Garantia da Qualidade

O C.E.S.A.R possui um grupo de garantia da qualidade o qual é alocado nos projetos em tempo parcial. Um engenheiro de qualidade, participante deste grupo, dedica seis horas semanais ao projeto para realizar atividades como: auditorias nos artefatos e processos, acompanhamento dos desvios, reportagem à equipe, ao gerente do projeto e à alta gerência do C.E.S.A.R. Estas atividades seguem um plano de garantia da qualidade, inserido dentro do plano do projeto, elaborado pelo engenheiro de qualidade do projeto.

4.3.4 Impacto do processo

O processo definido, com práticas do CMM e de XP, ocasionou em vários benefícios ao projeto. O gerenciamento passou a ser mais eficaz e visível através da identificação e acompanhamento de riscos, envolvimento da alta gerência, plano contemplando todos os principais aspectos do projeto e revisões formais. O projeto passou a ter um histórico das mudanças aos requisitos através das solicitações de mudança registradas na ferramenta de controle de mudanças do projeto, o que contribuiu também para melhor gerenciar as mudanças e os impactos decorrentes dela. Os problemas de gerenciamento de configuração foram consideravelmente diminuídos e o processo se tornou mais visível para toda a equipe.

O formalismo adicionado ao projeto se deu essencialmente no nível gerencial e em pouco impactou o trabalho da equipe como um todo. O projeto não perdeu a dinâmica proposta por XP: as iterações curtas, *releases* freqüentes, reuniões diárias, integração contínua, programação em pares contribui para a comunicação e motivação da equipe. As mudanças às iterações não sofreram nenhum impacto. Mudanças aos *releases*, considerados escopo do projeto, passaram a ser feitas através de solicitação de mudanças, tratadas nas reuniões de planejamento das iterações.

5. Conclusão

O nível 2 do CMM foca nas questões de gerenciamento e XP na produção de software. As duas abordagens são complementares e utilizá-las simultaneamente trouxe benefícios para o projeto. XP trouxe para o projeto motivação da equipe de desenvolvimento, estimativas mais confiáveis, maior controle sobre o progresso do projeto através das iterações curtas, maior satisfação do cliente através dos *releases* freqüentes. O CMM trouxe um gerenciamento mais eficaz. O experimento relatado demonstrou que é possível e harmonioso implementar as práticas do nível 2 do CMM e de XP, apesar de não ser possível afirmar que o nível 2 tenha sido completamente atingido, pois o projeto não passou por uma avaliação formal.

Referências

[1] J.A. Highsmith, Agile Software Development Ecosystems, Addison-Wesley, 2002
[2] K. Beck et al., Planning Extreme Programming, Addison-Wesley, 2000
[3] K. Beck, Exteme Programming Explained: Embrace change, Addison-Wesley, 1999
[4] M. Beedle, Agile Software Development with Scrum, Prentice Hall, 2001
[5] M. Fowler et al., "The Agile Manifesto", Software Development Magazine, August 2001
[6] M.C. Paulk et al., The Capability Maturity Model: Guidelines for Improving the Software Process, Addison-Wesley, 1995
[7] M.C. Paulk, "Extreme Programming from a CMM Perspective", IEEE, 2001
[8] R. Jeffries et al, Extreme Programming Installed, Addison-Wesley, 2000
[9] Web site da ferramenta Ant, http://ant.apache.org/ (último acesso em 19/05/2003)
[10] Web site da ferramenta Bugzilla , http://bugzilla.mozilla.org/ (último acesso em 19/05/2003)
[11] Web site da ferramenta CVS, http://www.cvshome.org/ (último acesso em 19/05/2003)
[12] Web site da ferramenta XPlanner, http://www.xplanner.org/ (último acesso em 19/05/2003)
[13] Web site do C.E.S.A.R – Centro de Estudos e Sistemas Avançados do Recife