
Metodologias Ágeis e Extreme Programming(XP) Uma Introdução

José Paulo Papo

Analista de TI na área de produtos auto-atendimento da TECBAN S/A
Professor da FIAP.

E-mails: j_paulop@yahoo.com

jose.papo@tecban.com.br

eXtreme Programming

Supere o Medo



Klaus Wuestefeld e Humberto Soares

Gerentes de Desenvolvimento

klaus@objective.com.br

humberto@objective.com.br



OBJECTIVE SOLUTIONS

“DESENVOLVER E IMPLANTAR SOLUÇÕES
PERSONALIZADAS DE SOFTWARE
COM NOSSOS PARCEIROS E CLIENTES”

Serviços de Desenvolvimento e Consultoria

- Projetos Especiais
- Processo e Ferramentas de Desenvolvimento

- Faturamento.....R\$4.5 milhões (2001)

- Escritórios.....São Paulo e Curitiba

- +50 Consultores e Desenvolvedores

Principais Clientes



Por que processo?
Para que metodologia?

Qual o melhor processo?

Montando a “Mochila”

- Equilibrando Utilidade, Peso e Confiabilidade
 - Menos Ítens
 - Ítens Mais Leves
 - Genéricos: Ítens com múltiplas funções
 - Menos Disciplina / Mais Diversão
 - Sinergia: O conjunto de itens se reforça
-

Montando a “Mochila”

- Quem tem interesse em aumentar o peso da sua mochila?
 - Cliente?
 - Gerente?
-

Objetivos

- Introduzir as Metodologias Ágeis e seu Manifesto
 - Introduzir Extreme Programming, seus princípios e práticas
 - Comparar e relacionar XP e RUP
 - Comparar e relacionar XP e CMM
 - O que buscamos: Aguçar a curiosidade e o interesse de todos os participantes em analisar e se aprofundar nas metodologias ágeis, em XP e outros processos de desenvolvimento, com vistas a melhorar o desenvolvimento de software de sua organização e, conseqüentemente, do Brasil.
-

Riscos e mitos do desenvolvimento de software

- Riscos

- Atrasos no cronograma.
- Cancelamento de projetos.
- Entropia de sistemas em produção.
- Taxa de defeitos.
- Incompreensão dos requisitos de negócio e sistemas.
- Mudanças constantes.
- “Scope Creep”(invasão de requisitos).
- Alta taxa de evasão de desenvolvedores.

- Mitos

- Podemos coletar todos os requisitos de uma só vez.
 - Podemos antecipar todas as mudanças.
 - Podemos controlar completamente todo o projeto de software.
 - Custo de mudança é, por natureza, maior conforme o projeto avança.
-

Metodologias Ágeis: As motivações

- Primórdios do desenvolvimento de software: “code and fix”.
 - A primeira alternativa: imposição de um processo disciplinado e detalhado, com o intuito de tornar o desenvolvimento de software mais predizível e eficiente. Inspirado em outras disciplinas da área de engenharia.
 - A crítica mais freqüente é que são burocráticas, não populares por imporem padrões rígidos e não serem tão bem sucedidas como deveriam. Ficaram conhecidas como metodologias “heavyweight” ou monumentais (termo de Jim Highsmith).
 - Como reação às metodologias “heavyweight”, surgiu um novo grupo de metodologias nos últimos anos, que ficaram conhecidas como metodologias “lightweight”, mas agora receberam o termo ágeis.
-

Metodologias Ágeis e Heavyweight: As diferenças

- O apelo inicial foi a reação ao aspecto burocrático das suas antecessoras e seu compromisso entre nenhum processo e muito processo.
 - Explicam que o desenvolvimento de software é inerentemente complexo e se assemelha a sistemas não lineares e caóticos.
 - São mais adaptativas que preditivas.
 - São mais orientadas a pessoas que orientadas a processos.
-

Manifesto para o desenvolvimento de software Ágil

Nós estamos descobrindo melhores maneiras de desenvolver software criando-o e ajudando outros a criá-lo. Através deste trabalho passamos a valorizar:

- **Indivíduos e interações** sobre processos e ferramentas.
- **Software funcional** sobre documentação extensa.
- **Colaboração do cliente** sobre negociação contratual.
- **Responder a mudanças** sobre seguir um plano.

Assim, enquanto há valor nos itens à direita, nós damos mais valor aos itens à esquerda

Benefícios da adoção

- Custos mais baixos ou não alterados
 - Melhoria da produtividade
 - Melhoria da qualidade do software produzido
 - Melhoria da satisfação dos clientes e para o negócio
-

O que é a XP?

- XP é uma metodologia “lightweight”(ou ágil) para equipes pequenas e médias, desenvolvendo software a partir de requisitos vagos e em rápida e constante mudança.
 - XP descreve uma maneira de desenvolver software que combina melhores práticas existentes na área há muito tempo.
 - Na XP essas práticas se complementam e controlam uma à outra. A diferença da XP é que ela pega essas práticas do senso comum e as utiliza a níveis extremos.
 - “XP é o mais importante movimento de nosso campo hoje. Eu vejo que ele será tão essencial para a geração presente quanto o SEI e o CMM foram para a anterior.” - Tom DeMarco
-

Os quatro valores da XP

- Comunicação
 - Simplicidade
 - Feedback
 - Coragem
-

Carta de direitos do cliente

- Você tem o direito a um plano, para saber o que será realizado, quando e a que custo.
 - Você tem o direito de receber o maior valor possível de cada semana de um desenvolvedor.
 - Você tem o direito de ver o progresso de um sistema sendo executado, que prove seu funcionamento ao passar por testes repetidos que você mesmo especifica.
 - Você tem o direito de mudar de idéia, substituir funcionalidade e mudar suas prioridades sem pagar custos exorbitantes.
 - Você tem o direito de ser informado de mudanças de cronograma, em tempo de escolher como reduzir o escopo para restaurar a data original. Você pode cancelar o projeto a qualquer tempo e ainda assim ter um sistema funcionando refletindo o investimento feito até o momento.
-

Carta de direitos do desenvolvedor

- Você tem o direito de saber o que é necessário, com claras declarações de prioridade.
 - Você tem o direito de produzir trabalho de qualidade o tempo todo.
 - Você tem o direito de pedir e receber ajuda de seus pares, superiores e clientes.
 - Você tem o direito de fazer e atualizar suas próprias estimativas.
 - Você tem o direito de aceitar as suas responsabilidades, ao invés de tê-las impostas a você de cima para baixo.
-

As doze práticas da XP

- Processo de Planejamento (“Planning Game”)
 - Releases Curtos
 - Metáfora
 - Projeto(Design) Simples
 - Testes
 - Refactoring
 - Programação em Pares
 - Propriedade Coletiva do Código
 - Integração Contínua
 - Semana de 40 Horas
 - On-Site Customer (Cliente sempre presente)
 - Padrões de Codificação
-

On-Site Customer (Cliente Sempre Presente)

- Um cliente deve estar sempre presente, ajudando a equipe a responder questões, resolver disputas e colocar pequenas prioridades em tarefas.
 - Pode-se dizer que um cliente é muito caro para se disponibilizar para a equipe de desenvolvimento.
 - Gerentes devem dizer o que tem mais valor: o software estar pronto mais cedo ou o trabalho de uma ou duas pessoas.
 - Segundo o Standish CHAOS Report de 1998 os principais fatores de sucesso para um projeto são o envolvimento do usuário, suporte dos executivos e objetivos de negócio claros (50% de importância, em conjunto).
-

Planning Game - Motivação

- Desenvolvimento de software é um diálogo constante entre pessoal de negócio e pessoal técnico.
 - Pessoas de negócio decidem sobre: escopo, prioridade, composição de releases e datas de lançamento.
 - As pessoas de negócio não podem tomar essas decisões no vácuo. Elas precisam do pessoal técnico para decidir sobre: estimativas, conseqüências técnicas, processo de desenvolvimento do produto, cronograma detalhado.
-

Planning Game – Como funciona?

- A XP trabalha com dois níveis de planejamento.
 - Planejamento do release: Cliente propõe “user stories”(estórias) e os desenvolvedores avaliam sua dificuldade através de semanas ideais.
 - Planejamento de iteração: cada iteração irá durar de uma a três semanas. Cada release pode ter várias iterações. Cada uma delas irá implementar algumas estórias definidas para o release.
 - Cada iteração é feita como uma “timebox”. Caso não se consiga implementar todas as funcionalidades requeridas, as que sobraem são colocadas na próxima iteração. Nunca se atrasa um “milestone”(marco).
 - O cliente pode mudar suas prioridades de requisitos a cada iteração que passa. Como elas são curtas(média de duas semanas) isso garante que mudanças são incorporadas tranquilamente.
 - Garante que o cliente tenha o maior ROI(retorno sobre o investimento) em cada iteração de desenvolvimento.
-

Releases Curtos

- Cada release deve ser tão curto quanto possível, contendo os requisitos de negócio de maior valor para o cliente.
 - O release deve fazer sentido como um todo. Não deve existir um release com metade de um requisito(o que não faz sentido).
 - Releases Curtos promovem o desenvolvimento iterativo e incremental.
 - Se iterações curtas são boas(como reconhecido por Frederick Brooks), elas serão feitas bem pequenas - alguns dias, semanas ou um mês por vez é melhor que seis meses ou um ano com antecedência.
-

Metáfora

- A metáfora é um meio de ajudar a todos (clientes e desenvolvedores) a compreender melhor o objetivo e propósito do produto sendo criado.
 - Pode ser considerado como a arquitetura de um sistema, vista como uma análise de domínio. Desse modo, pode-se construir entidades de software com alta coesão e baixo acoplamento entre si.
 - Se arquitetura é importante, todos irão trabalhar definindo-a e refinando-a o tempo todo.
-

Práticas de XP

- Pair Programming
 - Test Driven Development
 - Simple Design
 - Refactoring
-

Propriedade Coletiva do Código

- Módulos não são “propriedade” de nenhum desenvolvedor.
 - Todo desenvolvedor da equipe tem o direito de checar um módulo e modificá-lo.
 - O código é propriedade da equipe. Dessa maneira, todos os desenvolvedores se familiarizam com todo o código.
-

Integração Contínua

- Código é integrado e testado depois de algumas horas (no máximo no final de cada dia).
 - No final de um episódio de desenvolvimento, o código é integrado e todos os casos de teste devem rodar a 100%.
 - Essa técnica já existe há tempos e foi conhecida como “daily builds and smoke tests”, largamente utilizada na Microsoft.
 - Se integração de código é importante, ela será feita várias vezes ao dia.
-

Padrões de Codificação

- Cada equipe deve possuir padrões de codificação que serão usados por todos.
 - Idealmente, serão decididos por consenso.
 - Cada um dos padrões deve ter o claro objetivo de ajudar a melhorar a comunicação da equipe.
 - Todos devem concordar em utilizá-los.
-

Semana de 40 Horas (Passo Sustentável)

XP na perspectiva da RUP

- A RUP (e o Unified Process-UP) é um framework de processo, enquanto a XP é um processo em si, já definido e com limites claros.
 - A RUP precisa ser customizada.
 - Quatro dessas customizações apareceram que possuem grande semelhança com a XP e buscam agilizar o RUP.
 - Agile UP, de Craig Larman
 - Processo dX, Robert Martin
 - Devido ao grande sucesso da XP entre os desenvolvedores, a nova versão da RUP(2002) já vem com uma customização para projetos pequenos que possui características de metodologias ágeis e da XP.
 - A Rational também já disponibilizou um plugin de XP para o RUP, em seu portal para clientes RDN(Rational Developer Network).
 - Duas diferenças importantes entre o Agile UP e a XP: A UP recomenda a criação incremental de casos de uso e requisitos não-funcionais. A UP recomenda mais tempo gasto com diagramação e modelagem visual, e possui atividades mais detalhadas sobre isso. Os principais proponentes da XP recomendam pouco tempo gasto nessas atividades e com um nível de formalismo menor.
 - Resumindo, alguns autores consideram a XP(apesar das diferenças) como uma espécie de instância customizada da RUP.
-

XP na perspectiva do CMM

- As contradições vistas entre XP e o CMM são mitos.
 - O CMM é um menu e não diz como desenvolver software.
 - O CMM não impõe uma determinada metodologia ou ciclo de vida.
 - XP não é um processo indisciplinado. Pelo contrário, ele é “explicitamente definido, gerenciado, medido, controlado e efetivo” na definição de Paulk.
 - Como resultado, excluindo as KPAs de Gerenciamento de Subcontratação e SQA, XP devidamente implementado em uma organização pode atingir o CMM nível 2 e, com um pouco mais de trabalho, o nível 3.
-

XP na perspectiva do CMM

- “XP é um outro exemplo de um bom processo de software(ou filosofia) que, pelo menos dentro de um contexto apropriado, pode satisfazer muitos objetivos do CMM nível 2 e 3”.
- Segundo Mark Paulk, um dos “pais” do CMM, a seguinte tabela se aplica:

Table 3. Satisfaction of Software CMM Key Process Areas by XP.

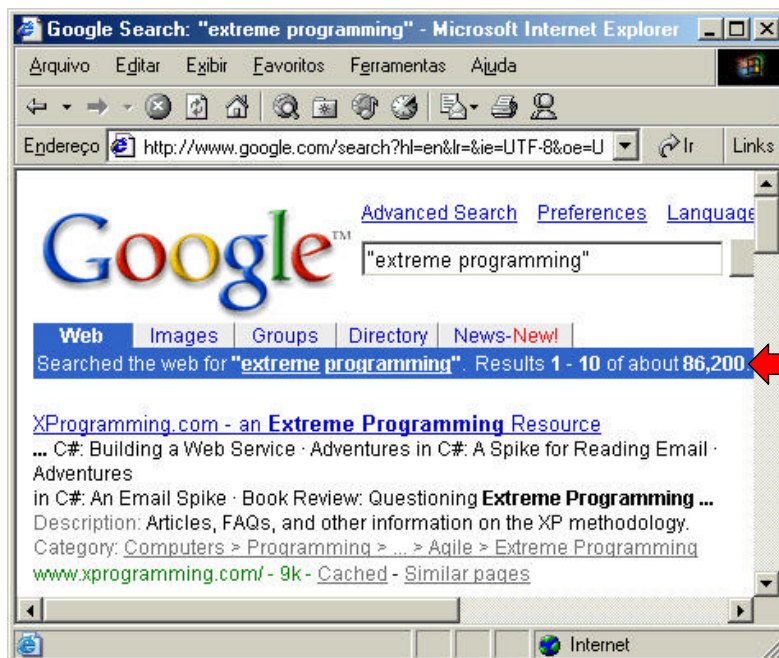
Level 2 KPA	Satisfaction	Level 3 KPA	Satisfaction	High Maturity KPA	Satisfaction
RM	√√	OPF	√	QPM	--
SPP	√√	OPD	√	SQM	--
SPTO	√√	TP	--		
SSM	--	ISM	--	DP	√
SQA	√	SPE	√√	TCM	--
SCM	√	IC	√√	PCM	--
		PR	√√		

√

partially addressed in XP

√√

largely addressed in XP (perhaps by inference)
(in the appropriate environment)



Google Search: "unified process" - Microsoft Internet Explorer

Arquivo Editar Exibir Favoritos Ferramentas Ajuda

Google™ [Advanced Search](#) [Preferences](#) [Language Tools](#) [Search Tips](#)

"unified process"

[Web](#) [Images](#) [Groups](#) [Directory](#) [News-News!](#)

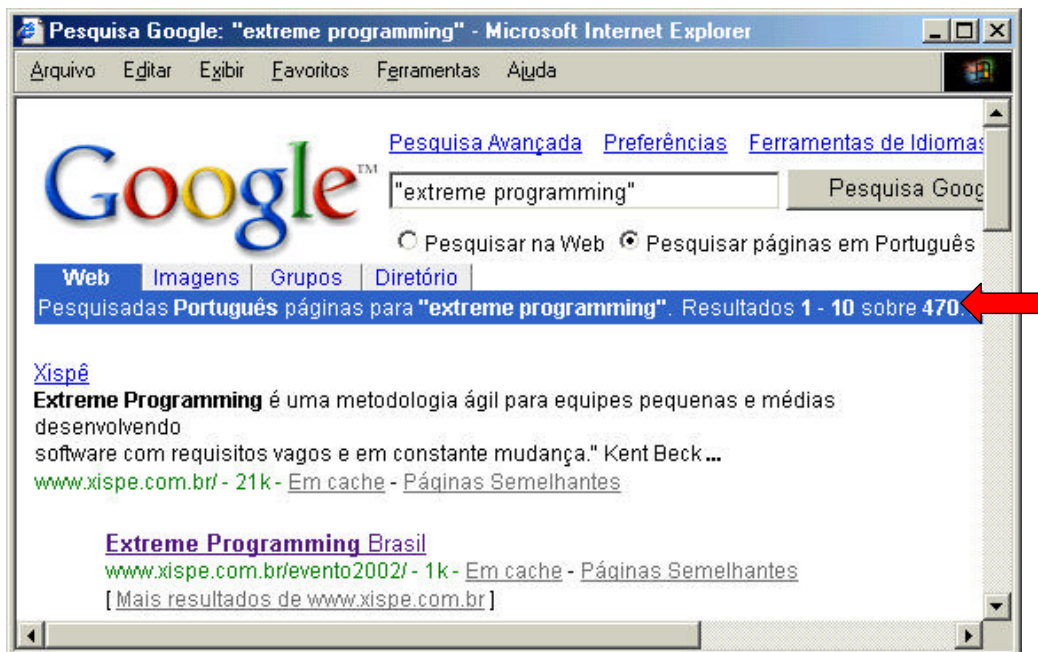
Searched the web for "unified process". Results 1 - 10 of about 39,900. Search took 0.29 seconds.

[Process Improvement and Software Development with Rational ...](#)
... category. Rational **Unified Process**®. New! Enhancements to the Rational **Unified Process** Platform for Developers and Project Managers. New! ...
www.rational.com/products/rup/index.jsp - 37k - [Cached](#) - [Similar pages](#)

[Rational Software](#)
Rational has the leading solution to the problems of e-development and other software projects, with best practices, services, and the premier tools for ...
Description: Helps organizations develop and deploy software for the internet and other platforms through tools, ...
Category: [Computers](#) > [Software](#) > [Retailers](#) > [Sales and Support](#)
www.rational.com/ - 38k - [Cached](#) - [Similar pages](#) - [Stock quotes: RATL](#)
[[More results from www.rational.com](#)]

[Ronin International -- Enterprise Unified Process \(EUP\) ...](#)
... Whitepaper download: Enterprise **Unified Process** (EUP) (~650k, last updated March 25, 2002). ... Books About Enhancing the RUP and the Enterprise **Unified Process**: ...
www.ronin-intl.com/publications/unifiedProcess.html - 11k - [Cached](#) - [Similar pages](#)





O Futuro da XP e das Metodologias Ágeis

- As metodologias ágeis e a XP estão tendo um alto grau de visibilidade por parte da comunidade de software. Muito disso ocorre por seu sucesso em diversos projetos ao redor do mundo.
 - Geoffrey Moore definiu o ciclo de vida de adoção de uma tecnologia(em seu livro “Cruzando o Abismo”) progredindo dos entusiastas de tecnologia aos visionários para os pragmáticos, depois os conservadores e chegando nos céticos.
 - A XP e as metodologias ágeis ainda continuam no domínio dos entusiastas e visionários, isto é, ainda não penetraram no grande mercado, dominado pelos pragmáticos. Quanto mais projetos forem bem sucedidos e mais estudos acadêmicos surgirem na comunidade de desenvolvimento de software, maiores as probabilidades de sucesso das metodologias ágeis.
 - Mas ainda está cedo para dizer se elas se tornarão o equivalente a uma nova tecnologia de ruptura(como descrito em “O Dilema da Inovação” de Clayton Christensen).
-

Conclusões

-
- Acredito, como defendido por Alistair Cockburn, que cada tipo de organização e/ou projeto demanda uma metodologia e processos de desenvolvimento diferentes. Parafraseando Frederick Brooks: “Não existe bala de prata”.
 - O vital para o sucesso de qualquer iniciativa para atingir a meta da Melhoria do Processo de Software é que ela tenha apoio da Alta Gerência e realmente seja vista como fundamental e urgente para a organização (seguindo o processo de oito etapas de Kotter).
 - Ainda assim, a controvérsia continuará por um certo tempo, pois o paradigma das metodologias ágeis cria radicalizações de ambos os lados e que, como vimos, não passam de mitos.
 - Lembrando Martin Fowler: “Qual a diferença entre um metodologista e um terrorista? Resposta: Você pode negociar com um terrorista.”
-

Bibliografia e Maiores Informações

- Astels, Miller, Novak(2002) Extreme Programming:Guia Prático Ed Campus
 - Beck, K. (2000). Extreme Programming Explained: Embrace Change.
 - Beck e Fowler(2001). Planning Extreme Programming.
 - Beck, K. (2002). Test-Driven Development by Example
 - Brooks, Frederick(1995) The Mythical Man-Month 20th Anniversary Edition.
 - Cantor, Murray(2002) Software Leadership.
 - Cockburn, A. (2000). Writing Effective Use Cases.
 - Cockburn, A. (2001). Agile software development.
 - Cusumano, M. e Yoffie, D.(1998). Competing on Internet Time.
 - Fowler, Martin e Scott, Kendal(2002) UML Essencial 2a. edição
 - Highsmith, J. A. (2002). Agile Software Development Ecosystems.
 - Larman, Craig(2002). Applying UML and Patterns, 2nd Edition.
 - McBreen, Pete(2002). Software Craftmanship.
 - McBreen, Pete(2002). Questioning Extreme Programming
 - McCarthy, Jim(1995). Dynamics of Software Development.
 - Palmer e Felsing (2002). A practical guide to Feature-Driven Development.
 - Schwaber, K. e Beedle (2002). Agile Software Development with SCRUM.
 - Smith, P. e Reinertsen, D.(1997). Desenvolvendo produtos na metade do tempo.
-

Bibliografia e Maiores Informações(2)

- Agile Alliance: www.agilealliance.org
 - Agile Modeling: www.agilemodeling.org
 - XP: www.xispe.com.br; www.xpers.com.br;
www.extremeprogramming.org; www.xprogramming.com;
 - Scrum: www.controlchaos.com
 - Crystal: www.crystallmethodologies.org
 - A Comparison of RUP and XP:
www.rational.com/products/whitepapers/423.jsp
 - Using RUP for Small Projects: Expanding upon XP:
www.rational.com/products/whitepapers/tp183.jsp
 - XP from a CMM Perspective: www.sei.cmu.edu/papers/xp-cmm-paper.pdf
 - The New Methodology:
www.martinfowler.com/articles/newMethodology.html
-

Dúvidas, sugestões, opiniões

“Sua idéia precisa ser original apenas quanto à sua adaptação ao problema em que você está trabalhando no momento” – Thomas Edison

“Nada é mais simples do que a grandeza; na verdade, ser simples é ser grande” – Ralph Waldo Emerson

“Pessoas são mais importantes que qualquer processo.” – Grady Booch

“Le mieux est l’ennemi du bien(O melhor é inimigo do bom)” – Voltaire

“O nosso é um mundo onde as pessoas não sabem o que querem e estão dispostas a ir ao inferno para conseguí-lo.” – Don Marquis

“Previsão é muito difícil, especialmente se for sobre o futuro.” – Anônimo

“Hoje, a era do processo pesado terminou.” - Tom DeMarco

“Não são as espécies mais fortes que sobrevivem, nem as mais inteligentes, mas as mais aptas à mudanças.” – Charles Darwin

“De todos os projetos iniciados na Terra, falham aqueles com grandes promessas.” – William Shakespeare
