XP - Extreme Programming

Resumo

Este trabalho tem por objetivo descrever a metodologia de desenvolvimento de software conhecida como XP (extreme programming), que visa prover uma metodologia que permita o desenvolvimento rápido e eficiente de sofwares de todos os portes e tem um número significativos de seguidores.

Autor: Roberto de Holanda Christoph

1. Introdução a XP.....3
2. Quando usar XP.....4
3. A metodologia.....5
3.1 - Planejamento.....5
3.2 - Design.....8
3.3 - Codificação.....9
3.4 - Testes.....12
4. Quando não usar XP.....13
5. Conclusão.....14
Bibliografia.....16

1 – Introdução a XP

Em 1996, Kent Beck começou um projeto em DaimlerChrysler usando novos conceitos em desenvolvimento de software, conceitos esses criados por ele no começo da década de 90 com ajuda de Ward Cunningham (pessoa com quem havia trabalhado). A esses novos conceitos deu-se o nome de metodologia de Extreme Programming (XP).

XP é basicamente um conjunto de regras, valores e princípios que irão permitir o rápido desenvolvimento de softwares sem comprometer sua qualidade, satisfazendo todos os requisitos do cliente de maneira satisfatória e em curto prazo. É conhecida como uma metodologia “leve”(Lightweight methodology) ou seja, não envolve um conjunto muito grande de regras e princípios a serem seguidos rigidamente, sendo portanto uma metodologia fácil de seguir.

A metodologia tem grande ênfase em trabalho de equipe, são considerados membro de uma equipe tanto os programadores quanto gerentes e principalmente o cliente, sendo que este terá sempre uma participação bem ativa junto a equipe de desenvolvimento.

A comunicação entre todos os membros da equipe é um fator essencial da metodologia XP, é através dela que será possível criar um design simples e eficiente para o projeto, possibilitando assim que este seja entregue ao cliente o mais cedo possível para que caso necessário sejam implementadas as mudanças nos requisitos solicitados.

Outro fator importante são os testes, já que estes são definidos antes da implementação em si e será através deles que será feita a avaliação de sucesso ou não de uma determinada parte do projeto.

2 – Quando usar XP.

A seguir estão algumas situações em que o uso da metodologia XP seria Recomendado:

- Mudança constante dos requisitos.

O melhor lugar para se usar XP é quando se encontra um projeto em que é esperado uma grande variação nos requisitos durante o seu período de desenvolvimento, já que esta metodologia é usada para a rápida adaptação a essas mudanças. Tais mudanças podem ser necessária devido a uma infinidade de fatores, como por exemplo indecisão por parte do cliente ou mesmo uma necessidade de reavaliação dos requisitos do projeto devido a fatores externos.

- Equipe de desenvolvimento pequena.

O tamanho ideal de uma equipe de desenvolvimento para esta metodologia não deve ser muito grande, sendo que o tamanho ideal de uma equipe deve ser de 2 a 12 programadores, pelo menos um gerente e o cliente com o qual estes irão interagir. Não é aconselhável manter uma equipe de programadores maior do que 12 membros, já que seu custo-benefício não será alto devido a metodologia em si funciona melhor com grupos menores.

É necessário ressaltar que a equipe escolhida para o projeto deve ser receptiva ao trabalho em grupo, já que outra marcante característica desta metodologia é o trabalho em conjunto em que todos os membros estarão profundamente envolvidos, será também necessário uma grande interação com o cliente e reuniões periódicas(em alguns casos até diárias) entre os desenvolvedores, gerentes e o cliente.

- Tempo disponível pequeno.

Outro fator que deve ser considerado é o tempo disponível, já que uma das mais chamativas características da metodologia é a de produzir um software de qualidade em curtos espaços de tempo.

- Necessidade de alta produtividade.

Os índices de produtividade atingidos com a metodologia são significativamente altos se comparados as metodologias tradicionais do meio corporativo.

3 – A metodologia.

A seguir uma descrição da metodologia XP em si, focando em um passo por vez como deve ser a sua implementação.

3.1 - Planejamento

Nessa fase é necessário a cooperação de toda a equipe, existirá uma interação muito grande entre os gerentes, clientes e programadores.

Primeiramente é necessário a formulação das “User Stories”, e para tanto o cliente deve descrever cada função que ele deseje ver no produto final. A descrição deve ser feita em linguagem natural pelo cliente e não deve ser muito longa, sendo o ideal aproximadamente 3 linhas escritas em um cartão. A descrição deve evitar ao máximo o uso de termos técnicos ou qualquer descrição de implementação ou hardware. Não é necessário entrar em muito detalhes ao se fazer as “User Sories”, já que cada uma será examinada mais detalhadamente em passos futuros, por enquanto só é necessário que se tenha uma vaga idéia do que será feito para se poder fazer um estimativa inicial (e ainda bem bruta) sobre o tempo de cada parte do projeto.

Após a elaboração das “User Stories”, é necessário que o cliente proponha testes para cada um destes. Esses devem ser simplesmente o que o cliente espera que o sistema responda devido a uma certa entrada. Esses testes serão importantíssimos, pois nenhuma “User Story” pode ser considerada completa sem ter passado em todos os testes propostos. Após isso é necessário um tempo para que todas elas sejam examinadas pela equipe e consequentemente sejam divididas em tarefas.

Tarefas são uma reformulação dos cartões, já que estes foram escritos em linguagem natural pelo cliente e estas são escirtas pelos programadores em linguagem de desenvolvimento.

Consequentemente as tarefas devem ser distribuídas pelos membros da equipe(não é necessário distribuir todas as tarefas, as que sobrarem serão divididas mais tarde) e a cada uma deve ser associado um tempo de desenvolvimento. É importante que esse tempo seja estimado pela pessoa que vai desenvolver a tarefa, já que isso varia muito de programador para programador, deve ser também levado em conta ao se fazer a estimativa se este terá dedicação integral a tarefa ou não. Não deve ser esquecido de se contabilizar no tempo final o tempo que será usado para a realização dos testes.

Cada tarefa não deve ultrapassar o tempo de desenvolvimento de 3 dias (dias ideais, ou seja, dias em que o programador se dedicará unicamente a tarefa). Caso o tempo estimado seja menor que um dia, a tarefa deve ser concatenada com outra, e se for acima de 3 dias deve ser dividida em duas ou mais tarefas.

Devem ser feitas reuniões periódicas no decorrer do projeto, essas reuniões devem ocorrer sempre em um período de 1 a 3 semanas e serão o coração do projeto. Nestas reuniões serão discutidos os progressos de cada tarefa e se necessário serão feitas reformulações nas divisões das tarefas bem como nos requisitos, a cada reunião serão feitas também as distribuições de novas tarefas para os programadores. O programador não deve nunca tentar adiantar o seu trabalho implementando “User Stories” que não lhe foram

atribuídos, estas serão implementadas no seu devido tempo quando forem prioridade.

Outro passo importante é a reunião onde serão definidas as datas de lançamento das versões do produto. Nesta reunião devem ser definidos quais “User Stories” devem ser implementadas em cada lançamento e também a data destes lançamentos. Novamente a participação do cliente é fundamental nesta fase, pois será ele quem irá decidir quais funções devem ser priorizadas nos lançamentos. O ideal é que hajam diversas pequenas versões do produto no decorrer da fase de desenvolvimento, para que possam ser examinadas logo pelo cliente. É sempre aconselhável começar por partes mais críticas do projeto, já que caso existam correções a serem feitas, estas ocorram ainda no começo. A velocidade do desenvolvimento deve ser acompanhada de perto pelos gerentes, pois caso exista algum disparate do planejado, deve-se fazer uma nova reunião para se reformular as datas de lançamento das versões, o que é perfeitamente normal no decorrer do projeto já que variações na velocidade de desenvolvimento são esperados em um caso real. O principal problema ao se estimar o tempo de um projeto é sem dúvida a sua estimativa inicial, mas logo no início do projeto(normalmente a partir da segunda reunião) já é possível se fazer uma estimativa muito melhor do que a primeira.

A metodologia XP também tem um foco muito grande na produtividade, e para tanto é necessário que se tenham certas considerações ao se planejar a divisão das tarefas. Tente evitar ao máximo deixar uma determinada parte do projeto ao encargo de apenas uma pessoa, pois isso causará um monopólio de conhecimento nesta área e caso essa pessoa deixe a equipe ou sofra alguma fatalidade, o projeto poderá sofrer um grande atraso. O ideal é ter sempre pelo menos duas pessoas trabalhando em cada área do projeto, e até programando juntos no mesmo computador(pair programming), claro que isso nem sempre é possível devido ao número de pessoas da equipe. Essa divisão de pessoal aumentará em muito a flexibilidade da equipe e ajudará na não existência de um grupo de pessoas sobrecarregada de tarefas enquanto outras não tem o que fazer.

Deve ser feito uma reunião informal todos os dias, onde serão discutidas as dificuldades encontradas. Essa reunião não deve ser muito longa e pode até mesmo ser feita em frente a um computador de um programador da equipe. O objetivo desta reunião é evitar que sejam feitas diversas outras durante o dia com um número reduzido de pessoas em cada uma delas, aumentando assim a produtividade.

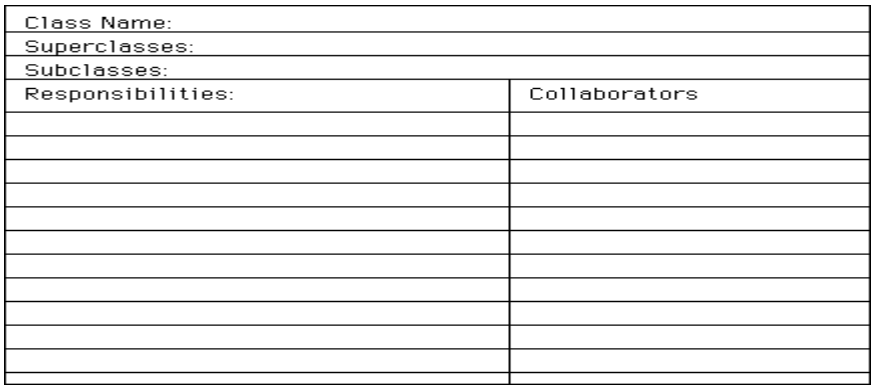
É muito importante nesta fase de planejamento não ter medo de mudanças nos requisitos ou de reavaliação de prazos, já que isso é perfeitamente normal de ocorrer durante qualquer projeto. Deve-se ter em mente que é importante a discussão antes de cada uma dessas mudanças, para se ter certeza de sua eficácia e da sua aceitação por parte da equipe.

3.2 - Design

A palavra chave para um bom design na metodologia XP é simplicidade. Sempre tente fazer tudo do jeito mais simples possível, um código simples é sempre muito mais fácil de interagir que um mais complexo, portanto sempre que achar uma oportunidade, simplifique ao máximo que puder. Tente evitar de implementar funcionalidades extras no início do projeto, mesmo parecendo que estas serão úteis no futuro, já que na verdade a maioria dessas funções nunca serão usadas no produto final e apenas representarão uma perda de produtividade, bem como o tamanho e complexidade do design .

Quanto as normas de programação(nomes de classes, objetos, etc), a metodologia não faz uso de nenhuma em particular, mas exige que os programadores da equipe façam uso de uma que seja a mais intuitiva possível. Com isso será possível que os programadores possam trocar, checar e consertar códigos de outros companheiros de forma mais segura e rápida. Uma outra forma de fazer a equipe colaborar com o design do projeto é fazer uso de “CRC Cards” (Use Class, Responsibilities, and Collaboration Cards). Cada cartão é usado para se representar um objeto, e sua classe deve ser escrita em cima dele, responsabilidades ficam na parte esquerda de baixo e as

classes colaborativas são escritas a direita de cada responsabilidade. A equipe deve discutir a disponibilidade dos cartões CRC e como devem ser as relações entre cada um. O problema com o uso de cartões CRC, é a falta de documentação escrita que esse método acarreta, uma solução para isso seria a reprodução escrita da disposição final dos cartões junto com as observações dos programadores.



CRC Cards

Tenha muito cuidado ao tentar reutilizar um design de um projeto antigo. A metodologia XP não aconselha tal prática, já que muitas vezes reutilizar um design significa se utilizar de várias partes consideradas caixas pretas, o que acabará aumentado a complexidade do design. Elimine sempre qualquer função que não tenha uma funcionalidade explícita e tente remover toda e qualquer redundância, lembre-se que o objetivo principal é ter um design o mais simples possível.

3.3 - Codificação

Como vimos anteriormente, a participação do cliente é uma peça fundamental na elaboração do projeto, será ele quem definirá as “User Stories”, assim como também definirá o que considera mais importante para ser colocado em cada nova versão do produto. Com isso em mente, é necessário que este esteja sempre disponível durante a fase de codificação, pois os programadores

deverão sempre recorrer a ele em caso de dúvida ou ambiguidade já que a transformação de uma “User Story” em uma tarefa nem sempre é trivial (muitos detalhes são omitidos em uma “User Story” para atender ao requisito de simplicidade) . O ideal é que este cliente passe a trabalhar no mesmo ambiente da equipe de desenvolvimento, para isso ocorrer deve haver um acordo com a empresa contratante para a liberação de pelo menos uma pessoa para representar esse papel. É importantíssimo também que esta pessoa não seja apenas um funcionário comum, e sim alguém que tenha conhecimento do projeto como um todo e esteja apto a responder com segurança a qualquer dúvida da equipe. Caberá a essa pessoa também avaliar quando uma versão está realmente pronta, verificando a validade dos testes e se realmente tudo está de acordo com o que fora planejado.

Em relação as normas de codificação, estas seguem as mesmas diretrizes ditas anteriormente em relação aos nomes de classes e objetos, ou seja, a metodologia XP não faz uso de nenhum conjunto de normas específicas, mas é necessário que o conjunto escolhido seja eficaz para gerar um código bem organizado e que possa ser entendido sem maiores problemas pelos outros programadores envolvidos no projeto. É mandatório que toda a equipe seja familiar com esse conjunto de normas, seja para poder entender um código alheio com mais facilidade ou para gerar o código com o intuito de que este possa ser entendido desta forma.

Como também foi dito anteriormente, os testes e suas respostas esperadas devem estar definidos antes do início da codificação de qualquer tarefa, deve-se começar a codificação tendo em mente que o código escrito deve ser o mais simples possível para resolver um destes testes, assim que o mencionado teste estiver ok, deve-se realizar o mesmo processo para um próximo teste e assim sucessivamente até que não hajam mais testes a serem realizados. Não implemente nada que não seja estritamente necessário, as funcionalidades adicionais devem ser deixadas de lado já que a maioria delas nunca é usada, gerando apenas mais código e provocando o aumento da complexidade.

O ideal é que todo código seja gerado usando “Pair Programming”, ou seja, duas pessoas programando no mesmo computador. A idéia pode parecer esquisita a primeira vista, e é provável que ocorram vários problemas no início se a equipe não estiver acostumada a trabalhar desta forma. Os benefícios trazidos pelo “Pair Programming” são a geração de um código de mais alto nível e uma melhor distribuição de conhecimentos. O primeiro benefício provém do fato de que enquanto um programador codifica, o outro tenta corrigir qualquer eventual erro e sugerir mudanças no código para o aumento de sua qualidade, já o segundo benefíco provém do fato desta metologia diminuir as já citadas “ilhas de conhecimento”, já que ambos os progamadores estarão a par do código e em caso de relocação, saída ou doença de um dos integrantes da equipe, ainda poderemos contar com o seu par.

Um grande problema que atinge a maioria do projetos é a integração de tarefas, pois mesmo que todas estas passem nos seus respectivos testes, isso não quer dizer de maneira alguma que funcionarão ao serem integradas para formar uma versão do projeto. Não se deve nunca deixar a integração de um projeto para a última hora, pois os problemas resultantes provavelmente irão acarretar em um atraso significativo, um jeito de se evitar isso pode ser nomeando uma equipe de integração para que esta fique responsável por essa parte, mas essa solução tem os seus problemas já que talvez não existam membros suficientes na equipe para a sua formação. Uma solução melhor seria a criação de um repositório onde cada par de programadores teria sua vez de depositar código e fazer os testes que lhe são cabidos, esta solução é simples e eficiente, deve-se apenas ter cuidado para duas ou mais equipes não depositarem código ao mesmo tempo enquanto estes são realizados. É importante que o código do repositório seja sempre o mais atualizado possível, por isso as equipe devem sempre depositar seus códigos mais recentes na primeira oportunidade que tiverem de acessar o repositório (quando outra equipe não o estiver usando), o ideal é que o código seja atualizado algumas vezes ao dia por cada par de programadores.

Um fato que é relevante de ser mencionado é que esta metodologia não aconselha de maneira alguma que sejam utilizadas horas extras para a finalização de alguma versão. Essas horas extras podem parecer inofensivas, mas elas cansam a equipe de desenvolvimento e isso acaba por gerar código de baixa qualidade assim como baixa produtividade nos dias seguintes. A melhor solução é se fazer uma reunião e reformular a data de entrega da versão.

3.4 - Testes

Os testes são um dos pontos chaves da metodologia XP, serão eles que determinarão se uma tarefa está completa ou não, mas para isso é necessário que estes testes tenham a cobertura mais ampla possível, por isso todas as classes da tarefa devem ser testadas, só pode ser deixado de lado o que for extremamente simples.

Todos os testes realizados e seus resultados obtidos devem ser arquivados juntamente com o código no repositório do projeto, para que todos possam ter livre acesso a eles. Outro fato que deve ser levado em conta, é a adoção de alguma ferramenta que automatize os testes, já que com esta seria possível que todos os testes fossem rodados novamente sempre que possível, verificando assim se existem erros em seções ateriores de código devido a uma mudança no código mais recente.

Quando um “bug” for encontrado e corrigido, deve-se fazer um novo teste que tenha a finalidade de verificar se ele realmente foi resolvido, este teste deve ser automatizado(no caso de uso de uma ferramenta para fazer os testes) juntamente com os demais e repetido sempre que possível.

4 – Quando não usar XP

Mesmo com todos os benefícios trazidos com o uso da metodologia XP, existem algumas situações em que ela não deve ser usada, situações essas como:

- Equipe muito grande - Quando a equipe de desenvolvimento é maior que 12 pessoas, muito provavelmente a metodologia não poderá ser usada com todo o seu potencial. Isso decorre do fato que é praticamente impossível manter o nível de comunicação desejável entre os membros, as reuniões se tornarão improdutivas, assim como o cliente não poderá dar a devida atenção a cada programador e dificilmente todos estarão a par de tudo o que ocorre com o projeto.

- Falta de perfil da equipe - Essa é uma questão delicada, mas que deve ser abordada. Certos profissionais não gostam de trabalhar em equipe e quando forçados a isso, geram problemas. Deve-se certificar-se de que a equipe tem o perfil necessário para se adotar a metodologia XP.

- Sistemas com um tempo de resposta muito alto – Caso o sistema a ser desenvolvido sofra deste problema, isto é, demore várias horas para compilar, ou necessita de uma resposta que demore um dia para processar, XP não deve ser a metodologia indicada. Para garantir qualidade é necessário que os testes sejam contínuos(se possível automatizados), e se para cada um destes testes for necessário esperar uma fatia considerável de tempo, teremos uma grande barreira que pode tornar o projeto inviável.

- Instalações físicas apropriadas – Esse item pode parecer irrelevante, mas não o é. Caso não exista espaço físico suficiente para o “pair programming” ou para a interação entre o cliente e os programadores(uma sala muito fechada ou um ambiente sensível ao barulho), isso pode acarretar grandes problemas no uso da metodologia, podendo tornar seu uso até inviável.

- Quando a empresa faz muito uso de horas extras - Algumas empresas fazem o intenso uso de verdadeiras “maratonas” de programação para cumprir prazos. O uso destas horas extras não é recomendável e impedirá o uso apropriado da metodologia, já que não é possível se usar XP a contento com uma equipe desgastada.

5 – Conclusão

É importante observar que a metodologia XP não é uma metodologia rígida de maneira alguma, logo após o término de um projeto, junte a equipe que participou do processo e discuta sobre o que foi observado ao longo dele, quais as vantagens apresentadas pela metodologia, se foi sentido um aumento de produção, se o ambiente de trabalho foi agradável, enfim pergunte o que cada um achou de trabalhar desta maneira e após todos falarem, discutam maneiras de melhorar ainda mais o processo eliminando o que foi desnecessário e priorizando as práticas que obteram maior sucesso. É necessário um cuidado extra neste ponto, pois se forem retirados muitos passos, o que se estará fazendo definitivamente não será XP, e seus benefícios serão menores do que o planejado(já imaginou um projeto XP sem “Pair Programming”, “User Stories” e reuniões com o cliente?).

A metodologia XP possui diversos pontos fortes que já foram discutidos anteriormente, destes é importante salientar o fato de se trabalhar com diversas versões do produto que vão crescendo até formar o todo, isto fará com que a equipe veja o produto tomando forma, o que aumentará a sua motivação, assim como os testes contínuos contribuirão para a certeza de que o projeto está se movendo para o lado certo.

O eficiência desta metodologia já foi verificada em vários experimentos práticos, e é algo que deve ser testado por uma equipe que tenha o perfil de gostar de trabalhar em conjunto, mas alguns itens ainda permanecem em aberto e devem ser alvo de estudos futuros, como exemplo podemos citar:

- Qual o tamanho ideal para uma equipe praticar XP onde encontraremos a eficiência máxima para esta metodologia?
- É realmente impossível praticar XP com uma equipe muito grande? O quão grande esta equipe seria então já que já houveram casos de 50 pessoas trabalhando conjuntamente com ela e obtendo sucesso?
- Quais os domínios em que XP se encaixa melhor? E pior?

Bibliografia

[KB] Kent Beck – Extreme Programming Explained – 1999

[WW] William C. Wake – Extreme Programming Explored – 2000

[GI] Extreme Programming a Gentle Introdution – http://www.extremeprogramming.org/ - 05/2002

[XR] XP programming.com na Extreme Programming Resource - http://www.xprogramming.com/ - 05/2002

[JB] John Brewer and Jera Design – Extreme Programming FAQ - http://www.jera.com/techinfo/xpfaq.html – 05/2002

[OO] OO Tips Extreme Programming - http://ootips.org/xp.html – 05/2002