XP
eXtreme Programming, uma metodologia ágil para desenvolvimento de software.

Equipe WEB Cercomp
web@cercomp.ufg.br

Introdução

- Criada por Kent Baeck em 1996 durante o projeto Daimler-Chrysler.
- O sucesso de XP advém da intensa satisfação do cliente. *Cliente satisfeito é o melhor indicativo de sucesso de um projeto*.
- Esta metodologia foi criada para produzir o software que o cliente precisa seguindo as especificações à risca.
- XP encoraja os desenvolvedores a atender as requisições de mudanças dos requisitos do software, no momento em que isto acontece.

Princípios de XP

- Existem alguns princípios que traduzem os princípios da metodologia XP e que devem ser rigorosamente seguidos, são eles:
  - Simplicidade;
  - Comunicação;
  - Feedback;
  - Coragem.

Princípios de XP

- Simplicidade:
  - Optar sempre pela solução mais simples possível (KISS);
  - Evitar soluções genéricas, pois elas podem provocar uma grande perda de tempo;
  - A solução deve responder simplesmente um requisito do usuário;
    - Algumas funcionalidades podem nunca vir a ser utilizadas.

Princípios de XP

- Comunicação:
  - A maneira de haver compreendimento entre a equipe de desenvolvimento e os usuários;
  - "*A comunicação é chave para o sucesso*".

Princípios de XP

- Feedback:
  - Deve ser o mais cedo possível, para saber se a equipe de desenvolvimento está no caminho correto;
  - Concreto, proveniente dos resultados dos códigos;
  - Constante, pois o ciclo de desenvolvimento precisa ser curto.

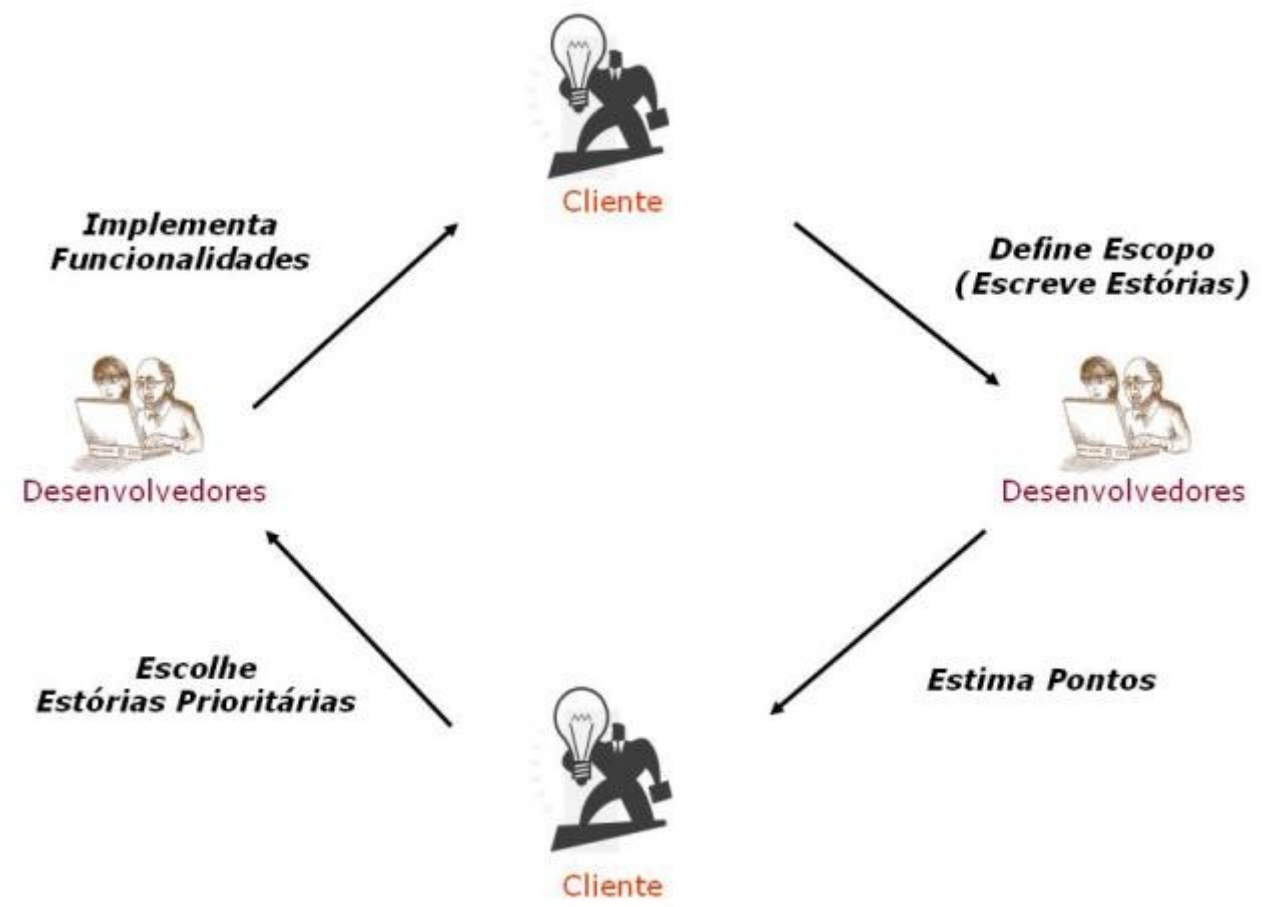
Princípios de XP

- Coragem:
  - Deixar o cliente sempre a par da situação atual do projeto;
  - Encarar abertamente as mudanças necessárias;
  - Aprender com os erros;
  - Acreditar nos Feedback's, não na teoria;
  - “Jogar para ganhar e não apenas para ter uma desculpa”.
  - Fazer o que precisar ser feito:
    - Jogar fora os códigos ruins.

Iniciando um projeto com XP, “Histórias dos usuários”

- Uma história é um texto ou parágrafo escrito pelo usuário afim de definir suas necessidades;
- O objetivo da “História” não é definir o escopo global do sistema, mas sim, estimar a complexidade de cada parte do sistema para poder estimar prazos;
- As “histórias” mais importantes e/ou mais difíceis tem prioridade;
- Os demais aspectos do sistema são tratados diretamente com o cliente no início do desenvolvimento;
- “É importante sempre haver um usuário disponível para trabalhar lado a lado com a equipe”.

Iniciando um projeto com XP, “Histórias dos usuários”



Planejamento do Release - Planning Game

- Cada release consiste de um número de iterações e cada iteração tem um conjunto de histórias implementadas. Em cada iteração devemos:
  - Planejar, para sempre fazer primeiro aquilo que é mais importante;
  - Programar, para garantir a finalização do software;
  - Testes, para saber se o programa está funcionando de acordo com a especificação;
  - Refatore, para garantir a qualidade dos seus códigos;
  - Escutar as dicas da equipe.

Pequenas Versões - Small Releases

- Liberação de pequenas versões funcionais do projeto (software com as funcionalidades que já foram implementadas até o momento);
- Pequenas versões auxiliam muito no processo de aceitação por parte do cliente, que já pode testar uma parte do sistema que está comprando.

Metáfora - Metaphor

- Facilita a comunicação com o cliente, entendendo a realidade dele;

  Ex: O conceito de rápido para um cliente de um sistema jurídico é diferente do mesmo conceito para um programador experiente em controlar comunicação em sistemas em tempo real.

- É preciso traduzir as palavras do cliente para o significado que ele espera dentro do projeto.

Propriedade coletiva do código

- POO causa uma necessidade de alteração em muitas partes do código;
- Coordenar as modificações leva tempo e gera resistências no autor do código;
- Em XP não se coordena, altera-se o que precisa ser alterado e TODOS são responsáveis por TODO o código;
- Padrões de codificação evitam PROBLEMÁTICAS.

Programação em Pares

- Em XP programação é feita em pares;
- Os pares podem mudar com relativa rapidez;
- Permite uma revisão de códigos constante;
- Favorece a comunicação e o aprendizado;
- É necessário estabelecer um padrão de codificação;
- Pode reduzir o tempo de desenvolvimento;

Desenvolvimento Orientado a Testes

- “Se você tem apenas a funcionalidade seu software está incompleto”.
- Testes significam uma “redundância lógica” que você adiciona ao código;
- Testes permitem que você refatore sem medo de quebrar o código;
- Escrevendo testes antes da funcionalidade, você clareia dúvidas sobre o que o software deve fazer;
- “Testes Unitários” - Testar métodos ou classes separadamente;
- “Testes de aceitação” - Construídos pelo cliente para aceitar um determinado requisito do sistema.

Reuniões em pé

- Também conhecidas por “stand-up meeting”.
- Reunião diária da equipe de desenvolvimento. Ela é breve e geralmente realizada pela manhã;
- Tem por objetivo compartilhar informações sobre o projetor e priorizar atividades específicas;
- Se possível, é bom que o cliente participe de tais reuniões.
- “*Em qualquer diálogo presencial, existem pelo menos duas coisas que são compartilhadas de maneira bastante rica: informações e emoções*”.

Visão geral


Extreme Programming
Planning/Feedback Loops
Release Plan
months
Iteration Plan
weeks
Acceptance Test
days
Stand Up Meeting
one day
Pair Negotiation
hours
Unit Test
minutes
Pair Programming
seconds
Code
© J. Donovan Wells

Problemas em XP

- Considerar que os testes são partes normais do processos de desenvolvimento do sistema;
- KISS (Keep It Stupid Simple);
- Admitir que você não sabe como fazer;
- Colaborar;
- Vencer resistência nas pessoas;
- Equipes grandes;
- Situações em que você não pode mudar o código sem pedir a permissão para o seu superior.

Referências

- Wikipedia – Programação Extrema
  - http://pt.wikipedia.org/wiki/Programa%C3%A7%C3%A3o_extrema
- Redes UNB – Metodologias Ágeis
  - www.redes.unb.br/material/ESOO/Metodologias%20Ã geis.pdf