Engenharia de Software Aula 04

Métodos Ágeis

Eduardo Figueiredo
http://www.dcc.ufmg.br/~figueiredo
dcc603@gmail.com

14 Março 2012

Agenda da Aula

- Processos que consideram requisitos voláteis
- Métodos ágeis de desenvolvimento de software
  - Programação Extrema (XP)

Lidando com Mudanças

Prototipagem
Entrega Incremental
Modelo Espiral

Lidando com Mudanças

- **Prototipagem**
- Entrega Incremental
- Modelo Espiral

Prototipagem

- É geralmente usada junto com outro modelo de processo
- Planeja e modela rapidamente um protótipo
  - Mais comum na definição de interfaces com os usuários (telas)
- Começa com os requisitos menos compreendidos
  - Objetivo: entender os requisitos

Prototipagem

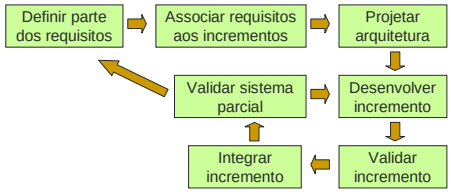
- O protótipo deveria ser descartado
  - E o sistema re-implementado usando outro modelo (exemplo, Cascata)
- Principal vantagem
  - Auxilia o engenheiro de software e o cliente a entenderem melhor o que deve ser construído

Lidando com Mudanças

- Prototipagem
- **Entrega Incremental**
- Modelo Espiral

Entrega Incremental

- Combina elementos do modelo cascata aplicados de maneira iterativa



Vantagens da Entrega Incremental

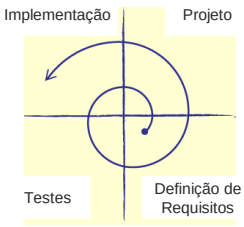
- Os clientes não precisam esperar a entrega final do sistema
  - Eles podem usar o sistema parcial
- Serviços de mais alta prioridade podem ser entregues primeiro
- O risco de falha global do projeto é menor que o Modelo Cascata

Lidando com Mudanças

- Prototipagem
- Entrega Incremental
- **Modelo Espiral**

Modelo Espiral

- Combina elementos dos modelos Incremental e Prototipagem
  - E sequência adotada do Modelo Cascata
- Software é desenvolvido em versões
  - Prototipagem nas primeiras versões
  - Incremental nas últimas versões



Processos de Software

- Modelo Cascata
- Desenvolvimento Incremental
- Baseado em Reuso
- Prototipagem
- Entrega Incremental
- Modelo Espiral

Cap. 2 do livro Sommerville, 9ª Edição

Métodos Ágeis

Por que processos ágeis?

- As regras de negócios mudam rapidamente
  - O software tem que ser adaptado para as novas regras
- Desenvolvimento e entrega rápida são importantes em mercados competitivos
  - A entrega rápida pode ser tão (ou mais) desejável que a qualidade

Processos Tradicionais vs. Ágeis

- Processos tradicionais são baseados em especificação detalhada dos requisitos, projeto e testes
- Métodos ágeis têm por objetivo criar software útil rapidamente
  - Não se preocupam com a documentação completa em todas as fases

De onde vem os métodos ágeis?

- Década de 80
  - A visão é de processos rigorosos para desenvolvimento de software
  - Objetivo: produzir sistemas grandes, complexos e de vida longa
- Década de 90
  - Métodos ágeis ganham força
  - Objetivo: se concentrar no software e não no projeto e documentação

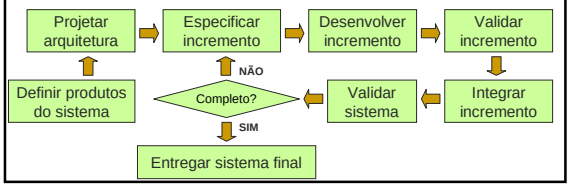
Manifesto Ágil

- Passamos a valorizar:
  - **Indivíduos e interações** mais que processos e ferramentas
  - **Software em funcionamento** mais que documentação abrangente
  - **Colaboração com o cliente** mais que negociação de contratos
  - **Responder a mudanças** mais que seguir um plano

http://agilemanifesto.org/

Modelo Geral dos Métodos Ágeis

- Processos ágeis são geralmente iterativos
  - Seguem uma série de incrementos
  - Cada incremento inclui uma nova funcionalidade ao sistema



Características Gerais

- Atividades de especificação, projeto e implementação são feitas em paralelo
  - Especificação não é detalhada
  - Documentação de projeto é mínima ou gerada automaticamente
- A interface do sistema é criada rapidamente
  - Antes das funcionalidades serem implementadas

Incrementos

- O sistema é desenvolvido por uma série de incrementos
- Usuários finais (ou representante do cliente) participam da especificação e validação de cada incremento

Vantagens

- Entrega acelerada de partes dos serviços ao cliente
  - Partes mais importantes podem ser entregues primeiro
- Engajamento dos usuários finais
  - Maior chance dos usuários ficarem satisfeitos com o produto

Problemas

- Gerenciamento
  - Falta documentação para o gerente
- Fechar o contrato com o cliente
  - Não há especificação completa do sistema
- Validar o sistema
  - A equipe de verificação e validação não tem a especificação
- Modificações contínuas podem corromper a estrutura do sistema

Manifesto Ágil - Princípios (3 de 12)

Nossa maior prioridade é satisfazer o cliente através da entrega contínua e adiantada de software com valor agregado.

Pessoas de negócio e desenvolvedores devem trabalhar diariamente em conjunto por todo o projeto.

O método mais eficiente e eficaz de transmitir informações para e entre uma equipe de desenvolvimento é através de conversa face a face.

http://agilemanifesto.org/

Princípios de Métodos Ágeis

- Envolvimento do cliente
- Entrega Incremental
- Pessoas, não processos
- Aceite as mudanças
- Manter a simplicidade
- Uso de metáforas

Cliente está engajado?

- Envolvimento do cliente
  - É atrativo
- Mas,
  - Depende de um cliente disponível
  - O cliente deve ser capaz de transmitir seu conhecimento a equipe de desenvolvimento

Problemas de Comunicação

- Foco na pessoa, não no processo
  - Valorização da equipe
- Mas,
  - Os membros da equipe devem ter perfil adequado para trabalho intenso e colaborativo
  - Exige muita comunicação

Mudanças x Simplicidade

- Priorizar mudanças é complicado
  - O que implementar primeiro?
  - Quando tem vários *stakeholders*, quem deve ser priorizado?
- Manter a simplicidade requer trabalho (intelectual) extra
  - Como lidar com a pressão do cronograma de entregas

Quando **evitar** métodos ágeis

- Sistemas grandes e complexos
  - O sistema deve ser gerenciável
- Quando as equipes trabalham em locais distribuídos
  - A comunicação é mais difícil
- Em sistemas críticos
  - Não pode haver falhas
  - Os requisitos devem ser completamente especificados

Programação Extrema (XP)

Programação Extrema (XP)

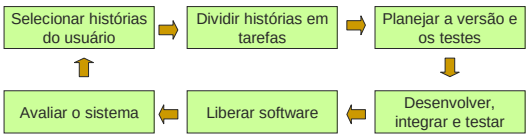
- Proposta a partir de boas práticas de desenvolvimento iterativo
- Propõe o envolvimento do cliente ao extremo
  - O cliente (ou seu representante) deve estar disponível durante todo o desenvolvimento
- Programadores trabalham em pares

Dos requisitos aos testes

- Os requisitos são escritos como cenários (estórias do usuário)
  - Estas estórias são implementadas diretamente por um conjunto de tarefas
- Para cada tarefa, é desenvolvido um conjunto de testes
  - Testes são feitos antes da implementação

Modelo de Processo XP

- O espaço de tempo entre releases é curto



Práticas de XP

- Planejamento incremental
- Pequenas releases
- Projeto simples
- Desenvolvimento dirigido por testes
- Refatoração
  - Reestruturação constante de código

Práticas de XP

- Programação por pares
- Propriedade coletiva do código
  - Qualquer um pode modificar qualquer coisa (não há ilha de conhecimento)
- Integração contínua
- Ritmo sustentável
- Cliente sempre disponível

Pequenas Releases

- Novas versões do sistema podem ser compiladas várias vezes por dia
  - Testes unitários automatizados devem ser executados após cada compilação
- Incrementos são entregues ao cliente a cada duas semanas

Refatorações

- XP prega que modelar o sistema para mudanças futuras é um esforço inútil
- Refatorações são constantemente aplicadas para permitir adaptações

Testes em XP

- XP enfatiza mais as atividades de teste que outros métodos ágeis
- Práticas
  - Desenvolver primeiro os testes
  - Codificação incremental a partir dos testes
  - Envolvimento do usuário na escrita e validação dos testes
  - Uso de ferramentas para testes automatizados
- Testes de aceitação também são incrementais

Desvantagens de Testar Primeiro

- Programadores preferem programar do que testar
  - Teste podem ser mal feitos ou incompletos
- Alguns testes são difíceis de escrever
  - Pode ser tão difícil quando implementar
- É difícil avaliar a abrangência dos testes

Programação por Pares (PP)

- Dois programadores sentam juntos na frente de um mesmo computador
- Os pares nem sempre são os mesmos
  - A alocação dinâmica dos pares é sugerido, pois favorece a propriedade coletiva do código

Vantagens de PP

- Responsabilidade comum
  - As acertos e falhas são de responsabilidade de toda a equipe
- Processo informal de revisão
  - Enquanto um programa, o parceiro revisa informalmente o código
- Favorece a melhoria da qualidade
  - Os parceiros discutem oportunidades para refatorações

Resumo

- Filosofia dos métodos ágeis
  - Desenvolvimento iterativo
  - Envolvimento do cliente
  - Evitar documentação desnecessária
- Exemplos de métodos ágeis
  - Programação Extrema (XP)

Bibliografia Principal

- Ian Sommerville. Engenharia de Software, 9ª Edição. Pearson Education, 2011.
  - Seção 3.1 a 3.4
- A. Koscianski e M. Soares. Qualidade de Software, 2ª Edição. Novatec, 2006.
  - Seção 10.3 Metodologias Ágeis

Bibliografia Adicional (XP)

- Kent Beck and Cynthia Andres. **Extreme Programming Explained: Embrace Change**, 2nd Edition. Addison-Wesley Professional, 2004.

Próxima Aula

- Laboratório 2011 e 2012 (*vou confirmar pelo website*)
  - Lab 2011 (Turma A): nomes começando com A até H
  - Lab 2012 (Turma B): nomes começando com I até Z
- Tema: Gerência de Projetos de Software