**Engenharia de Software** Prof. Thales Damasceno

**3o Bloco Engenharia de Software**

**Capítulo 5 Prática: Uma visão genérica**

**3o Bloco Engenharia de Software**

**Introdução**

A prática é uma visão de conceitos, princípios, métodos e ferramentas da qual um engenheiro de software faz uso diariamente.

**3o Bloco Engenharia de Software**

**Por que é importante?**

A prática fornece os detalhes de que você vai precisar para seguir caminho. Ela lhe diz onde estão as pontes, os bloqueios e as encruzilhadas, Ela a ajuda a entender os conceitos e princípios que precisam ser compreendidos e seguidos para dirigir com segurança e rapidez.

**3o Bloco Engenharia de Software**

Três elementos da prática aplicam-se independentemente do modelo de processo escolhido:

▪ **Conceitos**

▪ **Princípios**

▪ **Métodos**

**3o Bloco Engenharia de Software**

**A essência da prática**

Entenda o problema Planeje uma solução Execute o plano Examine o resultado quanto à precisão

**3o Bloco Engenharia de Software**

**Princípios Centrais (Princípios de Hooker)**

1o A razão por que tudo existe

2o Mantenha a coisa simples

3o Mantenha a visão

4o O que você produz outros vão “comer”

5o Esteja aberto para o futuro

6o Planeje com antecedência o reuso

7o Pense

**3o Bloco Engenharia de Software**

**Práticas de Comunicação**

\* Escute \* Prepare-se antes de se comunicar \* Alguém deve facilitar a atividade \* Comunicação face-a-face é melhor \* Conserve-se focado \* Faça documentações \* Busque colaboração \* Se algo não está claro, desenhe a figura \* Prossiga

**3o Bloco Engenharia de Software**

**Práticas de Comunicação DOCUMENTAÇÃO**

**3o Bloco Engenharia de Software**

**Práticas de Planejamento**

▪ Entenda o escopo do projeto

▪ Envolva o cliente na atividade de planejamento

▪ Reconheça que o planejamento é iterativo

▪ Estime com base no que é sabido

▪ Considere risco à medida que você define o plano

**3o Bloco Engenharia de Software**

**Práticas de Planejamento**

▪ Seja realista

▪ Ajuste a granularidade à medida que você define o plano

▪ Defina como você pretende garantir a qualidade

▪ Descreva como você pretende acomodar as modificações

▪ Acompanhe o plano com frequência e faça ajuste quando necessário.

**Prática de Modelagem**

Criamos modelos para obter um melhor entendimento da entidade real a ser construída.

**3o Bloco Engenharia de Software**

**3o Bloco Engenharia de Software**

**Prática de Modelagem**

No entanto, quando a entidade é um software, nosso

modelo precisa assumir uma forma diferente. No trabalho de engenharia de software duas classes de modelos são criadas:

**- Modelo de Análise - Modelos de Projeto**

**3o Bloco Engenharia de Software**

**Prática de Modelagem**

Modelo de Análise – Representa o requisitos do cliente mostrando os software em três domínios diferentes:

**1. O domínio de informação 2. O domínio funcional 3. O domínio comportamental**

**3o Bloco Engenharia de Software**

**Prática de Modelagem**

Modelos de Projeto – Representam características de Software que ajudam os profissionais a construí-lo efetivamente:

**1. A arquitetura 2. A interface do usuário 3. Detalhes em nível de componentes**

**Prática de Modelagem**

**Princípios da Modelagem de Análise**

1. O domínio de informação de um problema precisa ser repre-

sentado e entendido.

**3o Bloco Engenharia de Software**

**3o Bloco Engenharia de Software**

**Prática de Modelagem**

**Princípios da Modelagem de Análise**

**2. As funções a serem desenvolvidas pelo software devem ser definidas.** As funções podem ser descritas em vários níveis de abstração que vão desde uma declaração geral de objetivo ate uma descrição detalhada**.**

**3o Bloco Engenharia de Software**

**Prática de Modelagem**

**Princípios da Modelagem de Análise**

**3. O comportamento do software (como consequência de eventos externos) precisa ser representado.** O comportamento do software é guiado por suas interações com o ambiente externo.

**3o Bloco Engenharia de Software**

**Prática de Modelagem**

**Princípios da Modelagem de Análise**

**4. Os modelos que mostram informação, função e comportamento devem ser particionados de um modo que revele detalhes em forma de camadas (hierarquia)** Problemas complexos sãos difíceis de serem resolvidos como um todo. Por isso, usamos uma estratégia de dividir e conquistar. “Particionamento”.

**Prática de Modelagem**

**Princípios da Modelagem de Análise**

**5. Princípio 05: A tarefa de análise deve ir da informação essencial até os detalhes de implementação**

**3o Bloco Engenharia de Software**

**3o Bloco Engenharia de Software**

**Prática de Modelagem**

**Princípios da Modelagem de Projeto**

**1. O projeto deve estar relacionado com o modelo de análise.** O modelo de análise descreve o domínio de informação do problema, funções visíveis, o comportamento do sistema e um conjunto de classes. O modelo de projeto traduz essa informação em uma arquitetura.

**3o Bloco Engenharia de Software**

**Prática de Modelagem**

**Princípios da Modelagem de Projeto**

**2. Sempre considere a arquitetura do sistema a ser construído.** A arquitetura do software é o esqueleto do sistema a ser construído. Ela afeta as interfaces, estruturas de dados, fluxo de controle e comportamento do programa, o modo pelo qual o teste pode ser conduzido, a manutenção do sistema e muito mais.

**3o Bloco Engenharia de Software**

**Prática de Modelagem**

**Princípios da Modelagem de Projeto**

**3. O projeto dos dados é tão importante quanto o projeto de funções de processamento.** Um projeto de dados bem estruturado ajuda a simplificar o fluxo do programa, torna o projeto e implementação dos componentes de software mais fáceis e deixa o processamento global mais eficiente.

**3o Bloco Engenharia de Software**

**Prática de Modelagem**

**Princípios da Modelagem de Projeto**

**4. As interfaces precisam ser projetadas com cuidado.** Uma interface bem projetada torna a integração mais fácil e ajuda o testador.

**3o Bloco Engenharia de Software**

**Prática de Modelagem**

**Princípios da Modelagem de Projeto**

**5. O projeto de interface do usuário de estar sintonizado com as necessidades do usuário final.** A interface do usuário é a manifestação visível do software. Um projeto de interface pobre conduz, muitas vezes, a percepção de que o software é ruim.

**3o Bloco Engenharia de Software**

**Prática de Modelagem**

**3o Bloco Engenharia de Software**

**Prática de Modelagem**

**Princípios da Modelagem de Projeto**

**6. O projeto em nível de componente deve ser funcionalmente independente.** A independência funcional é uma medida de “objetividade” de um componente de software.

**3o Bloco Engenharia de Software**

**Prática de Modelagem**

**Princípios da Modelagem de Projeto**

**7. Os componentes devem ser fracamente acoplados uns aos outros e ao ambiente externo.** A medida que um nível de acoplamento aumenta, a probabilidade de propagação de erros também aumenta. Assim, o acoplamento de componentes deve ser mantido tão baixo quanto for razoável.

**3o Bloco Engenharia de Software**

**Prática de Modelagem**

**Princípios da Modelagem de Projeto**

**8. Modelos de projeto devem ser facilmente compreensíveis.** O objetivos do projeto é comunicar a informação para os profissionais que vão gerar o código, para aqueles que vão testar o software, e para os outros. Se o projeto for difícil de entender, ele não servirá como meio de comunicação efetivo.

**3o Bloco Engenharia de Software**

**Prática de Modelagem**

**Princípios da Modelagem de Projeto**

**9. O projeto deve ser desenvolvido iterativamente.** A cada iteração o projetista deve lutar por maior simplicidade.

**3o Bloco Engenharia de Software**

**Prática de Construção**

No trabalho moderno de engenharia de software, a codificação é a parte onde o código-fonte é gerado e testado para descobrir erros.

**Princípio 01: Todos os testes devem estar relacionados aos requisitos do cliente** – O objetivo do teste de software é descobrir erros.

**3o Bloco Engenharia de Software**

**Prática de Construção**

**Princípio 02: Os testes devem ser planejados com antecedência** – Todos os testes podem ser planejados e projetados antes que qualquer código tenha sido gerado, ou seja, o planejamento de testes.

**3o Bloco Engenharia de Software**

**Prática de Construção**

**Princípio 03: O princípio de Pareto se aplica ao teste de software** – Colocado simplesmente, o princípio de Pareto implica que 80% de todos os erros descobertos durante o teste estarão, provavelmente, relacionados a 20% de todos os componentes do programa.

**3o Bloco Engenharia de Software**

**Prática de Construção**

**Princípio 04: O teste deve começar “no varejo” e progredir até o teste “no atacado”** – Os primeiros testes planejados e executados concentram-se nos componentes individuais. À medida que o teste progride, o foco se desloca numa tentativa de encontrar erros em conjuntos integrados de componentes e, finalmente, em todo o sistema.

**3o Bloco Engenharia de Software**

**Prática de Construção**

**Princípio 05: Testes exaustivos não são possíveis -** é impossível executar todas as combinações de caminhos durante o teste.

**3o Bloco Engenharia de Software**

**Prática de Implantação**

**A atividade de implantação engloba três ações:**

**entrega suporte**

**feedback**

**3o Bloco Engenharia de Software**

**Prática de Implantação**

**Princípio 01: As expectativas do cliente quanto ao software devem ser geridas** – O engenheiro de software deve ser cuidadoso com o envio de mensagens conflitantes ao cliente

(por exemplo, prometer mais do que você pode realmente entregar).

**3o Bloco Engenharia de Software**

**Prática de Implantação**

**Princípio 02: Um pacote completo de entrega deve ser montado e testado.**

**3o Bloco Engenharia de Software**

**Prática de Implantação**

**Princípio 03: Um regime de suporte deve ser estabelecido antes de o software ser entregue**

**3o Bloco Engenharia de Software**

**Prática de Implantação**

**Princípio 04: Materiais institucionais adequados devem ser fornecidos aos usuários finais**

**3o Bloco Engenharia de Software**

**Prática de Implantação**

**Princípio 05: Software defeituoso deve ser corrigido primeiro e, depois entregue** – Pressionados pelo tempo, algumas organizações de software entregam incrementos de baixa qualidade com um aviso ao cliente de que os defeitos “serão corrigidos na versão seguinte”

**3o Bloco Engenharia de Software**

**Referências bibliográficas**

▪ LEITE, Jair C**. Engenharia de Software [*homepage* na Internet].** Natal-RN: UFRN; 3 de Junho de 2007. Acesso em 20 de Janeiro de 2013. Disponível em:

http://engenhariadesoftware.blogspot.com.br/2007/06/modelos- de-software.html

▪ PRESSMAN, Roger S. Engenharia de Software, Sexta Edição. Editora MCGrawHill: Porto Alegre, 2010.

**3o Bloco Engenharia de Software**

**Desenvolvido por:**

Jacinto Vasconcelos Albuquerque

Mikael Araújo

Roberta Poliana Fonseca Ribeiro