

Gerenciamento de Configuração

Objetivos

- Explicar a importância do gerenciamento de configuração de software (CM)
- Descrever as atividades fundamentais de gerenciamento de configuração: gerenciamento de mudanças, gerenciamento de versões e construções de sistemas

Gerenciamento de configuração

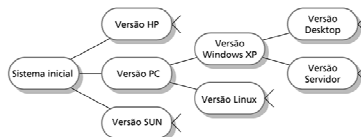
- Novas versões de sistemas de software são criados quando eles:
 - Mudam para máquinas/OS diferentes;
 - Oferecem funcionalidade diferente;
 - São configurados para requisitos de usuários particulares.
- O gerenciamento de configuração exerce controle sobre os artefatos produzidos pelo desenvolvimento de software:
 - Mudança de sistema é uma atividade de equipe;
 - O CM tem por objetivo controlar os custos e o esforço envolvidos na realização das mudanças em um sistema.

Gerenciamento de configuração

- Envolve o desenvolvimento e a aplicação de procedimentos e padrões para gerenciar um produto de software em evolução.
- O CM pode ser visto como parte de um processo mais geral de gerenciamento do projeto.
- Quando liberados para o CM, os sistemas de software são chamados, algumas vezes, de baselines, visto que são um ponto de partida para desenvolvimento posterior.

Famílias de sistemas

Figura 29.1
Famílias de sistemas.



Padrões de CM

- O CM deve ser sempre baseado em um conjunto de padrões que são aplicados dentro de uma organização.
- Os padrões devem definir como os itens de configuração são identificados, como as mudanças são controladas e como as novas versões são gerenciadas.
- Os padrões podem ser baseados em padrões de CM externos (por exemplo, o padrão IEEE para CM).

Construção freqüente do sistema

- É mais fácil encontrar problemas que surgem das interações de componentes no início do processo.
 - Em especial quando usa-se incrementos pequenos e *builds* frequentes
- Isso encoraja o uso de testes automatizados – os desenvolvedores estão sob pressão para não 'quebrar a construção'.
- O processo de gerenciamento de mudanças precisa alcançar equilíbrio:
 - Burocracia vs. Rastreabilidade

Planejamento de gerenciamento de configuração

- Todos os produtos do processo de software podem ser gerenciados:
 - Especificações;
 - Projetos;
 - Programas;
 - Dados de teste;
 - Manuais de usuário.
- Milhares de artefatos separados podem ser gerados para um sistema grande e complexo de software.
- É necessário definir quais estão sujeitos ao gerenciamento de configuração

Gerenciamento de mudanças

- Sistemas de software estão sujeitos a solicitações contínuas de mudanças:
 - De usuários;
 - De desenvolvedores;
 - De forças de mercado.
- O gerenciamento de mudanças está relacionado à manutenção da rastreabilidade dessas mudanças, de modo que:
 - Reparos realmente corrijam falhas
 - Novas falhas introduzidas por reparos possam ser identificadas rapidamente
 - Seja fácil descobrir quais mudanças foram implementadas e quando

Formulário de solicitação de mudança

Figura 29.4

Formulário de solicitação de mudança parcialmente preenchido.

Formulário de Solicitação de Mudança	
Projeto: Prototipo/Parâmetros PCL	Número: 2502
Solicitante da mudança: I. Sommerville	Data: 1/12/02
Mudança solicitada: Quando um componente é selecionado da estrutura, apresentar o nome do arquivo onde ele está armazenado.	
Analista da mudança: G. Dean	Data da análise: 10/12/02
Componentes afetados: Display-Icon.Select, Display-Icon.Display	
Componentes associados: FileTable	
Avaliação da mudança: Relativamente simples de implementar se uma tabela de nome de arquivo estiver disponível. Requer o projeto e a implementação de um campo na tela. Não é requerida nenhuma mudança dos componentes associados.	
Prioridade da mudança: Baixa	
Implementação da mudança: Isforça estimado: 0,5 dia	
Data para o CCB: 15/12/02	Data de decisão do CCB: 1/2/03
Decisão do CCB: Aceita a mudança. Mudança a ser implementada no Release 2.1	Data da mudança:
Implementador da mudança:	Decisão do GQ:
Data de submissão ao GQ:	
Data da submissão ao CM:	
Comentários	

Formulário de solicitação de mudança

Figura 29.4

Formulário de solicitação de mudança parcialmente preenchido.

Formulário de Solicitação de Mudança	
Projeto: Prototipo/Parâmetros PCL	Número: 2502
Solicitante da mudança: I. Sommerville	Data: 1/12/02
Mudança solicitada: Quando um componente é selecionado da estrutura, apresentar o nome do arquivo onde ele está armazenado.	
Analista da mudança: G. Dean	Data da análise: 10/12/02
Componentes afetados: Display-Icon.Select, Display-Icon.Display	
Componentes associados: FileTable	
Avaliação da mudança: Relativamente simples de implementar se uma tabela de nome de arquivo estiver disponível. Requer o projeto e a implementação de um campo na tela. Não é requerida nenhuma mudança dos componentes associados.	
Prioridade da mudança: Baixa	
Implementação da mudança: Isforça estimado: 0,5 dia	
Data para o CCB: 15/12/02	Data de decisão do CCB: 1/2/03
Decisão do CCB: Aceita a mudança. Mudança a ser implementada no Release 2.1	Data da mudança:
Implementador da mudança:	Decisão do GQ:
Data de submissão ao GQ:	
Data da submissão ao CM:	
Comentários	

O formulário já dá uma boa idéia sobre como o processo de gerenciamento de mudanças funciona.

Um Exemplo Prático de Formulário

Bugzilla Bug 52094 byatt should give ben \$50 - Mozilla Firefox

Query: page Enter new bug

Bug: 52094 alias: Hardware: PC OS: Windows 2000 Version: Trunk Status: VERIFIED Resolution: WORKING Assigned To: byatt@mozilla.org (David Hyatt) QA Contact: jgm@netscape.com URL: http://www.zachipton.com/ben Summary: byatt should give ben \$50 Status: Whiteboard: Keywords: helpwanted, meta, modem, nsonty, pp, testcase

Reporter: ben@netscape.com (Ben Goodger) CC: adam@gmp.org, andreas@fedoraproject.org, bharat@vfp.net, blake@netscape.com, brian@mozilla.org

Flags: (Help) blocking-1.4x, blocking-1.5a Requester:

Attachment	Type	Created	Flags	Actions
1	text/html	2003-07-14 12:14		

Acompanhamento de mudanças

- O maior problema no gerenciamento de mudanças é o acompanhamento do status da mudança.
- Ferramentas de gerenciamento de mudanças fornecem meios para se acompanhar a situação de cada solicitação de mudança
 - Automaticamente enviam solicitações de mudança para as pessoas certas no tempo certo.
- São integrados a sistemas de e-mail, permitindo a distribuição eletrônica da solicitação de mudança.
 - Mesmo assim, é comum que muitas solicitações de mudanças sejam sumariamente ignoradas

Comitê de controle de mudanças

- As mudanças podem ser revisadas por um grupo externo, que decide se elas são ou não adequadas em termos de custo, tempo e risco, a partir de um ponto de vista estratégico ou organizacional ao invés de um ponto de vista técnico.
- O grupo deve ser independente do responsável de projeto pelo sistema. Esse grupo é, algumas vezes, chamado de comitê de controle de mudanças (CCB).
- O CCB pode conter representantes do cliente e do pessoal fornecedor.

Procedência histórica

- É um registro das mudanças realizadas em um documento ou um componente de código.
- Deve registrar, em linhas gerais, a mudança feita, a lógica da mudança, quem fez a mudança e quando foi implementada.
- Pode ser incluída como um comentário no código.
 - Se um estilo de cabeçalho padrão é usado para a procedência histórica, as ferramentas podem processar isso automaticamente.



Informação de cabeçalho de componente

Figura 29.5

Informação de cabeçalho de componente.

```
// Projeto BANKSEC (S1 608/7)
//
// BANKSEC.TOOLS\AUTH\RBA\USER_ROLE
//
// Objeto: currentRole
// Autor: N. Perwez
// Data de criação: 10 de novembro de 2002
//
// (c) Lancaster University 2002
//
// Historico de modificação
// Versão  Implementador  Data      Mudança      Razão
// 0       J. Jones      1/12/2002  Adicionar cabeçalho Submetido ao CM
// 1       N. Perwez     9/4/2003  Novo campo   Solicit. de mud. 107/02
```

Algumas Ferramentas de Gerenciamento de Mudanças

- Bugzilla 
- IBM Rational ClearCase
- Mantis 
- Também é possível usar um Wiki com esse fim

Gerenciamento de versões e releases

- Elaborar um esquema de identificação para versões de sistema.
- Planejar quando uma nova versão de sistema será produzida.
- Assegurar que procedimentos e ferramentas de gerenciamento das versões sejam adequadamente aplicados.
- Planejar e distribuir releases da nova versão do sistema.

Versões/variantes/releases

- Versão É uma instância de um sistema que é funcionalmente distinta, de alguma maneira, de outras instâncias de um sistema.
- Variante Uma versão de um sistema que tem apenas pequenas diferenças com relação a outras instâncias (normalmente devido a diferenças no hardware/software alvo)
 - Ex.: O Office para MacOS é uma variante do Office para Windows.
- Release É uma instância de um sistema distribuída para os usuários fora da equipe de desenvolvimento.

Identificação de versões

- Os procedimentos para identificação de versões devem definir uma maneira não ambígua de identificação de versões de componentes.
- Existem três técnicas básicas para identificação de componentes:
 - Numeração de versões;
 - Identificação baseada em atributos;

Numeração de versões

- É um esquema simples de numeração usa uma derivação linear
 - V1, V1.1, V1.2, V2.1, V2.2 etc.
- A estrutura de derivação real é uma árvore ou uma rede, e não uma seqüência.
- Os nomes não são significativos.
- Um esquema de hierarquia de atribuição de nomes conduz a poucos erros na identificação de versões.

Um Exemplo: Números de Versões no Linux



- A.B.C[D]
 - A – versão do *kernel* (apenas duas mudanças: em 1994 e em 1996)
 - B – revisão importante do *kernel*
 - C – mudanças menores: novos *drivers* e novas funcionalidades individuais
 - D – atualizações de segurança e correções de *bugs*
- Versão atual: 2.6.24.3

Identificação baseada em atributos

- Os atributos podem ser associados a uma versão com a combinação de atributos que a identificam.
 - Exemplos de atributos são Data, Criador, Linguagem de Programação, Cliente, Status, etc.
- É mais flexível do que um esquema explícito de atribuição de nomes para recuperação de versões; contudo, pode causar problemas com a unicidade
- Na prática, uma versão também necessita de um nome que facilite a referência.
 - Atributos, nesse aspecto, têm limitações

Consultas baseadas em atributos

- Uma importante vantagem da identificação baseada em atributos, é que ela pode apoiar consultas tais que você pode encontrar 'a mais recente versão em Java', etc.
- A consulta seleciona uma versão dependendo dos valores de atributos
 - Ex.: (linguagem =Java, plataforma = XP, data = Jan 2003).

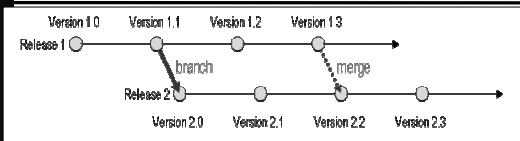
Branching e Merging

- Um elemento fundamental do gerenciamento de configuração
 - O livro não fala sobre!
- Compromisso entre produtividade e risco
- Branching:** Consiste em usar diferentes “ramos” de desenvolvimento para aumentar o paralelismo
 - Cada ramo é chamada de *branch*
 - Código não é compartilhado entre *branches*
- Merging:** a combinação de uma desses ramos com o ramo principal
 - Diferenças entre os *branches* combinados precisam ser resolvidas

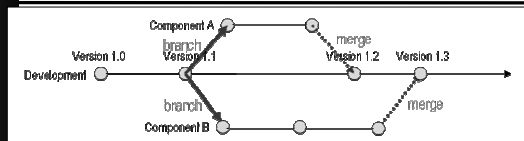
Algumas Razões para se Criar um Branch

- Implementar uma solicitação de mudança
- Implementar uma funcionalidade pontual
- Paralelizar o desenvolvimento dos componentes do sistema
 - Também aplicável ao desenvolvimento paralelo de diferentes versões do sistema
 - Atribuição de tarefas a diferentes partes da equipe de desenvolvimento

Branch-per-Release e Code-Promotion-Branches



Branch-per-Component e Branch-per-Technology



Anti-Padrões de Branching e Merging

- Merge-Paranoia*
- Merge-Mania*
- Big-Bang-Merge*
- Branch-Mania*
- Cascading Branches*
- Vejam “A Branching & Merging Primer”, de Chris Birmele
 - Parte do material desta aula foi tirada desse tutorial
 - Já está no grupo da disciplina

Funcionalidades de um Sistema de Controle de Versões

- Manutenção de um repositório de itens de configuração
 - Com suporte ao *checkin* e ao *checkout* distribuídos
- Criação e manutenção de múltiplas versões
 - Armazenamento de informações sobre cada versão
- Criação e *merging* de *branches*
- Capacidade de realizar consultas sobre versões dos sistemas, com base em seus atributos

Releases de sistema

- Um *release* não é apenas um conjunto de programas executáveis.
- Pode incluir também:
 - Arquivos de configuração definindo como o release é configurado para uma instalação particular;
 - Arquivos de dados necessários para a operação do sistema;
 - Um programa de instalação para instalar o sistema no hardware alvo;
 - Documentação eletrônica e em papel;
 - Empacotamento e publicidade associada.

Criação de releases

- A criação de releases envolve a coleta de todos os arquivos e a documentação necessária para criar um release de sistema.
- Descrições de configuração têm de ser escritas para hardware diferente e scripts de instalação têm de ser escritos.
- O release específico deve ser documentado para registrar exatamente quais arquivos foram usados para criá-lo. Isso permite que ele seja recriado, se necessário.

Construção (*build*) de sistemas

- É o processo de compilação e ligação de componentes de software em um sistema executável.
 - Pode incluir a execução de testes
- Sistemas diferentes são construídos a partir de combinações diferentes de componentes.
- Esse processo é, atualmente, sempre apoiado por ferramentas automatizadas que são dirigidas por 'scripts de construção'.

Construção de sistemas

Figura 29.7
Construção de sistemas.

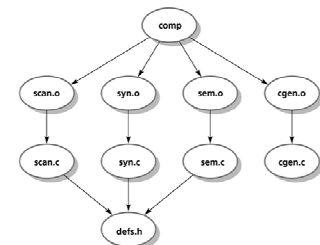


Construção de sistemas



- A construção de um sistema grande é computacionalmente dispendiosa e pode levar várias horas.
- Centenas de arquivos podem estar envolvidos.
- As ferramentas de construção de sistemas podem fornecer:
 - Uma linguagem de especificação de dependência e um interpretador associado;
 - Seleção de ferramentas e apoio à instanciação;
 - Compilação distribuída;
 - Gerenciamento de objetos derivados.


Dependências entre componentes

Figura 29.9
Dependências entre componentes.



Algumas Ferramentas de Controle de Versões e Geração de *Builds*

- CVS (+ WinCVS)
- SVN 
- Git 
- IBM Rational ClearCase

- Ant 
- GNU Make 