Controle de Versão com CVS

Danilo Toshiaki Sato
www.dtsato.com

Treinamento ALESP – SPL

Agenda

1. Introdução
2. Conceitos Importantes
3. Operações mais comuns - Eclipse
4. Processo de Liberação – SPL
5. Conclusão
6. Workshop

Introdução

- CVS = *Concurrent Versions System*
- Principais funções:
  1. Guardar histórico de mudanças:
     - Habilidade de voltar à versão do dia anterior
  2. Colaboração:
     - CVS não faz *lock* de arquivos
     - Permite que mais de um desenvolvedor trabalhe no mesmo arquivo

Introdução

- Colaboração simultânea é possível através do modelo **copiar-alterar-juntar**:
  - **Copiar**: Os desenvolvedores pedem uma cópia local para o servidor
  - **Alterar**: O desenvolvedor A altera sua cópia local e envia para o servidor
  - **Juntar** (*merge*): O desenvolvedor B envia sua alteração para o servidor e é informado sobre o conflito

Conceitos Importantes

- **Revisão (*revision*):**
  - Um número que representa uma "foto" do arquivo no servidor
  - Cada commit de arquivo, altera seu número de revisão
  - Exemplos de número de revisão: `1.3`, `1.3.2.2`
  - Um número de revisão sempre tem um número par de inteiros separados por `'.'`
  - Revisão inicial padrão de cada arquivo é `1.1`
  - Números de revisão com mais de um `'.'` representam branches

Conceitos Importantes

- **Repositório:**
  - A cópia mestre onde o servidor do CVS mantêm todo o histórico de revisões
  - Cada projeto tem exatamente um repositório
- **Cópia Local (*working copy*):**
  - A cópia onde você faz suas alterações nos arquivos do projeto
  - Podem exitir diversas cópias locais para o mesmo arquivo do projeto
  - Geralmente cada desenvolvedor tem sua cópia local

Conceitos Importantes

- ***Check out*:**
  - O ato de pedir uma cópia local para o repositório
  - Sua cópia local irá refletir o estado do arquivo no momento do *check out*
- ***Commit*:**
  - O ato de enviar suas alterações na cópia local para o repositório
  - Também conhecido como *check in*
  - Para que outros desenvolvedores enxerguem suas alterações, você deve publicá-las através de um *commit*

Conceitos Importantes

- **Mensagem de Log:**
  - Um comentário que descreve cada alteração
  - Anexado à revisão no momento do commit
  - Importante para comunicar o que foi alterado para outros desenvolvedores
- **Atualização (*update*):**
  - O ato de trazer alterações feitas no repositório para sua cópia local
  - Operação complementar ao *commit*

Conceitos Importantes

- **Conflito:**
  - A situação quando dois desenvolvedores tentam "*commitar*" mudanças feitas numa mesma região do arquivo
  - O CVS avisa sobre o conflito, mas o desenvolvedor deve resolvê-lo
- ***Tag*:**
  - Os números de revisão de cada arquivo são independentes
  - Uma *tag* é um nome simbólico para uma determinada revisão de um ou mais arquivos

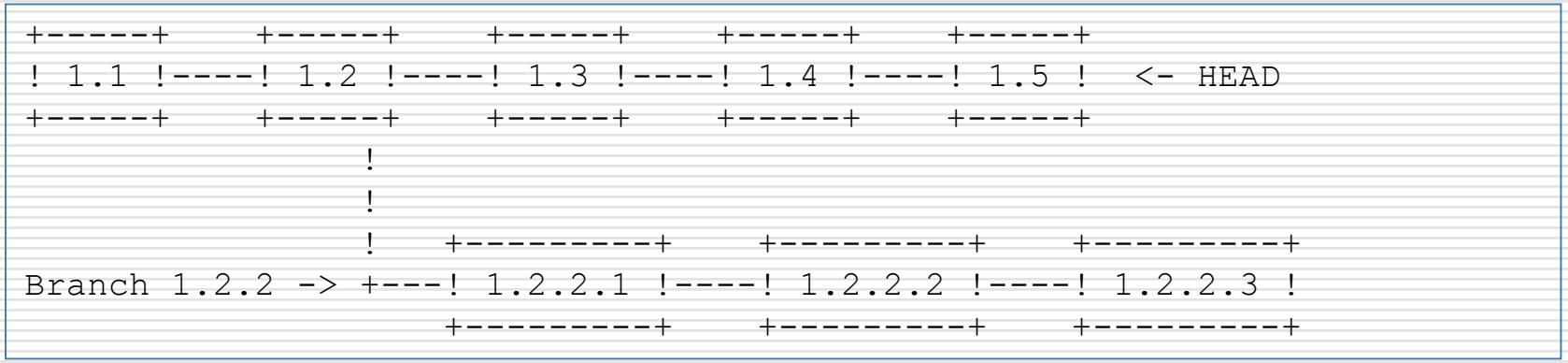
Conceitos Importantes

```
arquivo1 arquivo2 arquivo3 arquivo4 arquivo5
1.1      1.1      1.1      1.1      1.1
1.2      1.2      1.2      1.2
1.3      1.3      1.3      1.3
1.4               1.4      1.4
                  1.5
                  1.6
```

```
arquivo1 arquivo2 arquivo3 arquivo4 arquivo5
                  1.1
                  1.2
         1.1      1.3
1.1      1.2      1.4      1.1
1.2*-----1.3*-----1.5*-----1.2*-----1.1* <-- Tag (Ex: "Bug_21")
1.3               1.6      1.3
1.4                        1.4
```

Conceitos Importantes

- **Branch:**
  - É um recurso que permite isolar mudanças numa linha de desenvolvimento paralela
  - Mudanças feitas num branch não são imediatamente refletidas da linha principal (HEAD) ou em outros branches

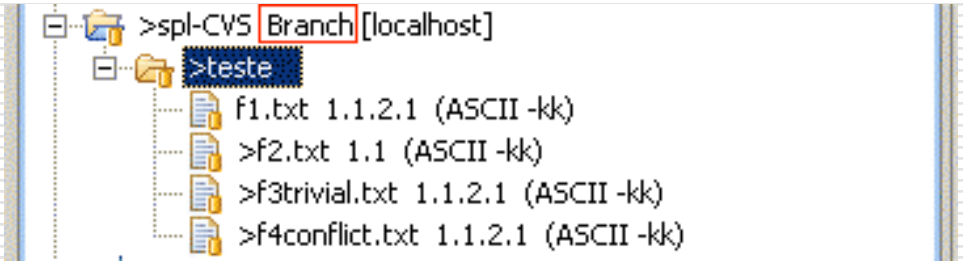


Operações mais Comuns

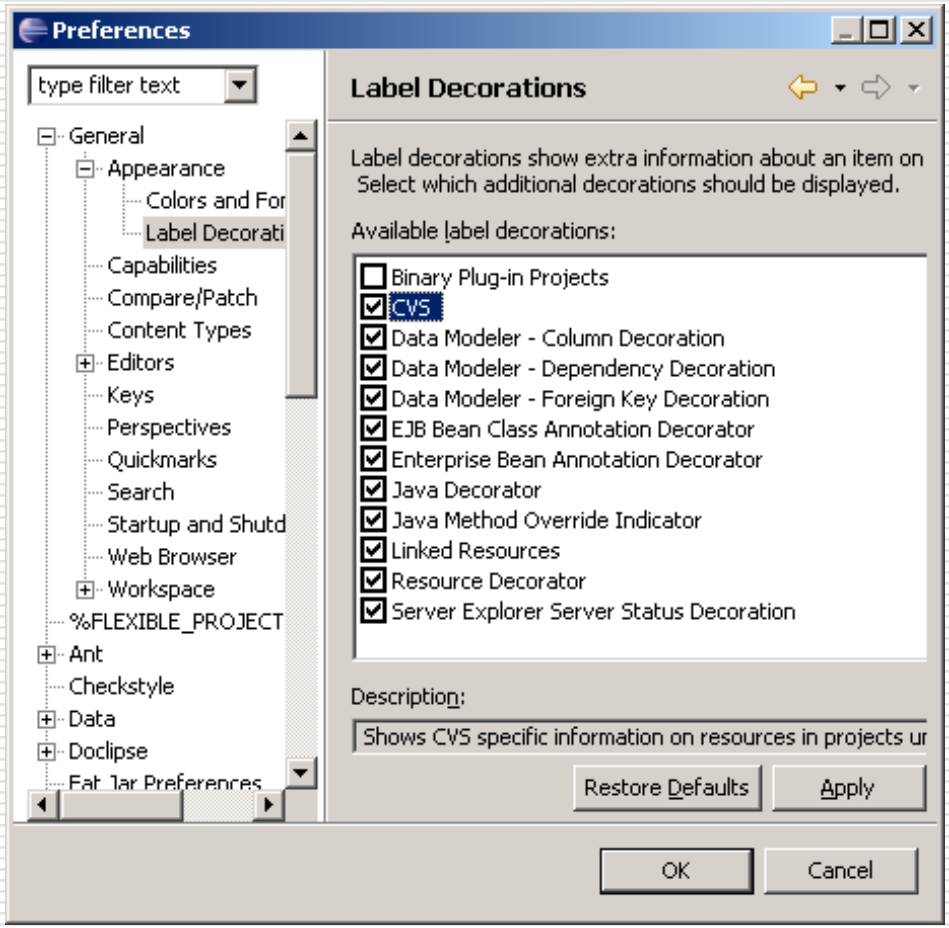
- Eclipse:
  - *Tag* = Versão
  - Versão ≠ Revisão
  - Perspectiva do Repositório CVS:
    - Gerenciamento dos repositórios e projetos
    - Gerenciamento de *Branches* e Versões (*Tags*)
    - Histórico de Revisões
    - Árvore de Versões (*Tags*)

Operações mais Comuns

- Configuração de *Labels*:
  - Permite a identificação da *tag/branch* associada à cópia local
  - Abra *"Windows > Preferences"*, navegue até *"General > Appearance > Label Decorations"* e escolha a opção *"CVS"*



Operações mais Comuns



Operações mais Comuns

- Sincronizar a cópia local:
  - Operação para verificar as diferenças entre a cópia local e o repositório
  - Permite:
    - Atualizar (*update*) a cópia local com alterações feitas no repositório ()
    - *Commit* de arquivos alterados ()
    - *Merge* para resolver conflitos ()
- Basta clicar com o botão direito no recurso desejado (projeto, arquivo ou pasta) e escolher *"Team > Sincronyze with Repository"*

Operações mais Comuns

- Atualização (*update*):
  - *"Team > Update"*
- *Commit*:
  - *"Team > Commit..."*
  - Entrar com a mensagem de log do *commit*
- Resolução de conflitos (*merge*):
  - Na visão de sincronização, resolver o conflito na cópia local
  - Clicar com o botão direito no arquivo e escolher *"Mark as merged"*

Operações mais Comuns

- Criar uma *tag*:
  - *Tag* = Versão
  - Clicar com o botão direito no recurso desejado
  - *"Team > Tag as Version..."*
  - Preencher o nome da *tag* e clicar em *"OK"*
  - É possível mover uma *tag* existente para a revisão atual
  - É possível criar uma *tag* diretamente no repositório, através da perspectiva CVS

Operações mais Comuns

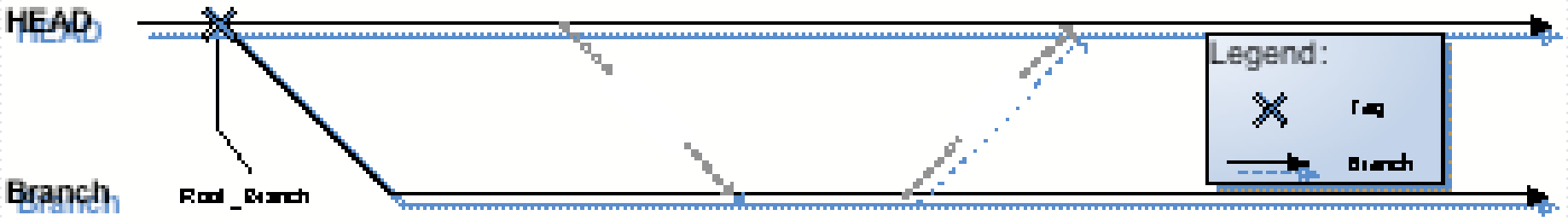
- Substituir a cópia local com uma versão (*tag*)/*branch* diferente:
  - Clicar com o botão direito no recurso desejado
  - *"Replace With > Another Branch or Version..."*
  - Escolha a *tag/branch* desejada e clique em *"OK"*
  - O label com o nome da *tag/branch* deve aparecer ao lado do nome do projeto
  - Ao fazer o *check out* de uma *tag*, você não pode alterar seus arquivos e fazer *commit*:
    - *Tags* estão associadas à números de revisão existentes

Operações mais Comuns

- Criar um *branch*:
  - Atualize sua cópia local com o *branch* que servirá de base (HEAD) para o novo *branch*
  - Clique com o botão direito no recurso desejado
  - *"Team > Branch..."*
  - Entre com o nome do *branch*
  - Deixe o checkbox *"Start working in the branch"* marcado caso queira atualizar sua cópia local com os arquivos do novo *branch*
  - O campo *"Version Name"* é automaticamente populado, representando o nome de uma *tag* que marca o início do *branch* (útil na hora do *merge*)

Operações mais Comuns

- *Merge* entre *branches*:
  - Ocorre em duas direções:
    - Do *branch* para o *sub-branch* (***rebase***)
    - Do *sub-branch* para o *branch*
  - Recomenda-se fazer um *rebase* antes de fazer o *merge* para o *branch* principal
  - Do ponto de vista da operação no CVS, só muda o alvo do *merge*



Operações mais Comuns

- *Merge* entre *branches*:
  - Atualize sua cópia local com o *branch* que servirá como alvo do *merge:*
    - *Rebase*: alvo = *sub-branch*
    - *Merge*: alvo = HEAD
  - Clique com o botão direito no recurso desejado
  - *"Team > Merge..."*
  - Escolha o *branch* desejado (***end tag***) e o ponto inicial comum (geralmente o *root tag* do *branch*)
  - A visão de sincronização será exibida
  - Faça o *merge* na sua cópia local
  - Faça o *commit* da cópia local já atualizada

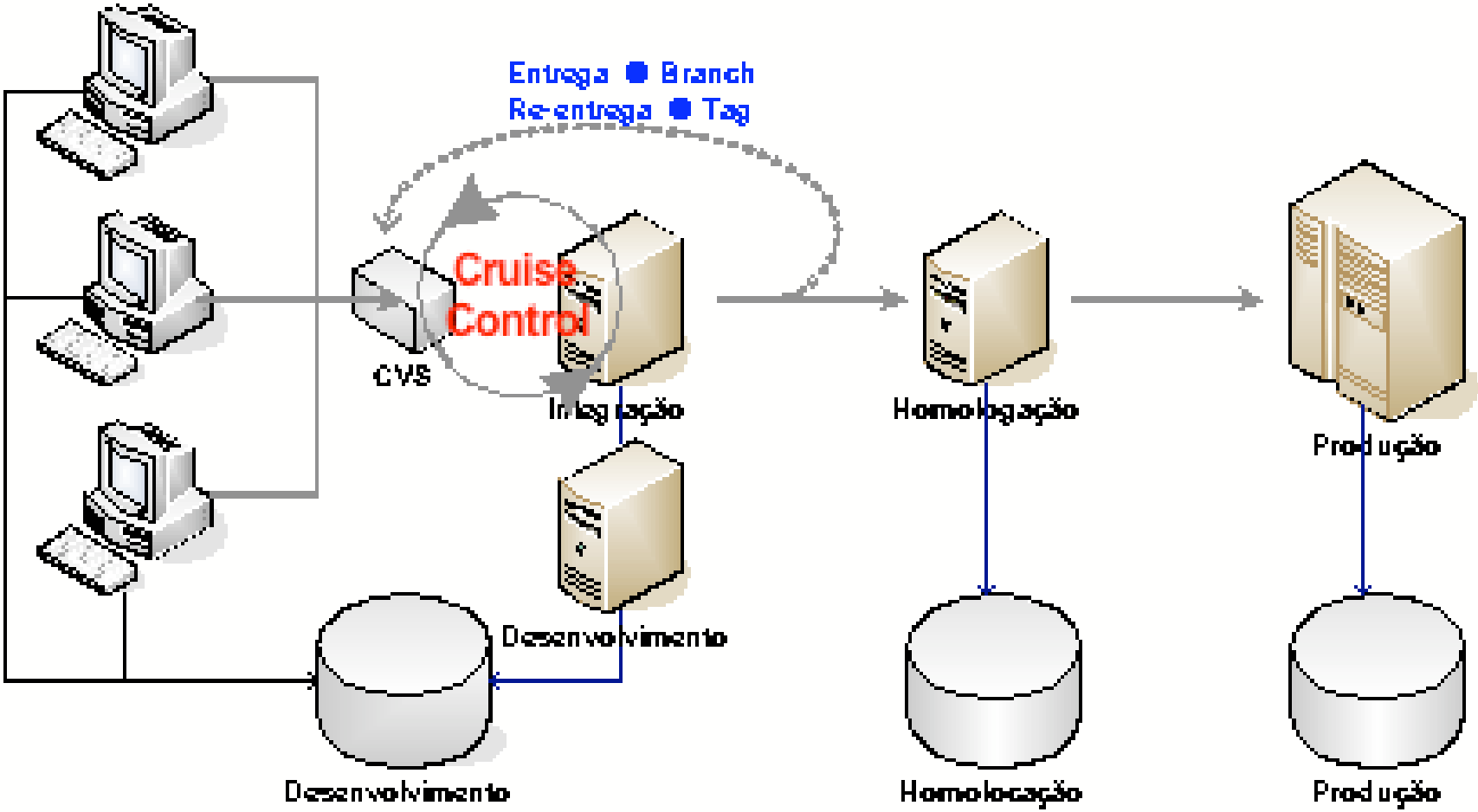
Processo de Liberação - SPL

- Separação hierárquica em 4 ambientes:
  1. Computador Local (BD: Desenvolvimento)
  2. Ambiente de Integração (BD: Desenvolvimento)
  3. Ambiente de Homologação (BD: Homologação)
  4. Ambiente de Produção (BD: Produção)
- Código não pode "pular" nenhum ambiente
- Integração entre 1 e 2 será feita pelo CVS e pelo Cruise Control
- Entregas entre 2 e 3 (*releases*) correspondem a um *branch* no CVS (Ex: *"Release1"*)

Processo de Liberação - SPL

- Homologação → Produção não acarreta mudança no código nem no CVS
- Novas funcionalidades são desenvolvidas sempre no HEAD
- Releases passam a ser numeradas:
  - *"Versão_1"*, *"Versão_2"*, *"Versão_2_1"*
- Erros em homologação/produção:
  - São corrigidos no *branch* e devem passar pelo ambiente de integração
  - Na re-entrega para homologação, uma nova *tag* deve ser criada no *branch* do *release*
  - Após a re-entrega, deve ser feito um *merge* do *branch* para o HEAD para que as atualizações sejam refletidas na próxima entrega

Processo de Liberação - SPL



Processo de Liberação - SPL

- Vantagens:
  - Processo controlado de entregas
  - Possibilidade de voltar para qualquer versão entregue no passado
  - Separação entre código instável (desenvolvimento de novas funcionalidades) e código entregue
  - Solução de bugs não impacta o desenvolvimento de novas funcionalidades

Conclusão

- Visão geral dos principais conceitos e operações do CVS
- Realização das operações no ambiente Eclipse
- Proposta para um novo processo de liberação no SPL
- Surgimento de um novo papel: *"Release Manager"*

Referências

- Branching with Eclipse and CVS: http://www.eclipse.org/articles/Article-CVS-branching/eclipse_branch.html
- Branches, merging and tags: http://plone.org/documentation/tutorial/cvs/branches-and-tags
- CVS official site (for those braves enough who don't want to use Eclipse CVS client): http://www.nongnu.org/cvs
- CVS Wikied Manual: http://ximbiot.com/cvs/wiki/index.php?title=CVS--Concurrent_Versions_System_v1.12.12.1

Dúvidas ou Sugestões?