RELPREV

Tutorial básico - Subversion

Versão 1.0

Histórico da Revisão

Data	Versão	Descrição	Autor
10/10/2013	1.0	Criação do documento	Natan Pimenta

Índice Analítico

Sumário

Tutorial básico - Subversion

1. O que é Subversion ?

Subversion (http://subversion.tigris.org) é um sistema de controle de versões *open source*. Gerencia arquivos e diretórios ao longo do tempo, acompanhando as alterações, quem as fez e quando foram feitas, entre outras. É provável que qualquer que seja o cenário onde exista uma coleção de arquivos e pessoas que os alteram ao longo do tempo, teremos a necessidade de serviços oferecidos por sistemas de controle de versões. Subversion é um robusto exemplo de tais sistemas.

2. Conceitos

Repositório: Um diretório contendo arquivos gerenciados pelo Subversion é um repositório. O repositório armazena os dados gerenciados pelo Subversion em um servidor (máquina que tem a missão de executar os serviços do Subversion). É no repositório que versões são armazenadas e podem ser recuperadas, assim como o histórico de alterações.

Cópia de trabalho: Usuários não têm acesso direto às informações mantidas em um repositório. (Há algumas exceções.) Para alterá-las é preciso a existência de uma cópia de trabalho. É na cópia de trabalho que reside uma cópia de todo ou parte do conteúdo do repositório que se deseja trabalhar. Em um determinado instante, podem existir zero ou mais cópias de trabalho oriundas de um determinado repositório. Também convém ressaltar que, em geral, o repositório se encontra em máquina diferente daquelas das cópias de trabalho (empregadas por desenvolvedores). Modificações não são feitas diretamente no repositório, mas em cópias de trabalho e, posteriormente, enviadas para o repositório (o meio compartilhado). Em conseqüência, quando se perde acidentalmente uma cópia de trabalho, perde-se apenas as alterações nela efetuadas e não transferidas para o repositório. O repositório só é alterado quando explicitamente é requisitada a sua atualização. Quando uma alteração é feita em uma cópia de trabalho e enviada para o repositório, então esta alteração torna-se visível a todos aqueles que têm acesso ao repositório. O histórico de alterações é guardado no repositório e pode ser posteriormente consultado.

Revisão: Também é conhecida em inglês por changeset. Mudanças são efetuadas em cópias de trabalho e, quando um conjunto destas deve ser persistido no repositório, as alterações correspondentes no repositório dão origem a uma nova revisão ou changeset. Noutras palavras, ao longo do tempo, alterações em cópias de trabalho que são persistidas dão origem a revisões. Nem toda mudança em uma ou mais cópias de trabalho dá origem a uma revisão. O responsável por uma cópia de trabalho pode simplesmente não efetivar as alterações e, inclusive, simplesmente remover toda a cópia de trabalho com alterações. Uma revisão é gerada apenas quando mudanças em uma cópia de trabalho são enviadas para o repositório, tornando-as persistentes. Esta operação recebe o nome de **commit**. Cada revisão possui um número único, acrescido de uma unidade a cada mudança efetuada no repositório. O número de uma revisão é suficiente para identificar todo o conteúdo do repositório imediatamente após a mudança que deu

origem ao número em questão. Desta forma, com o número apropriado, pode-se criar uma cópia de trabalho gerada em algum tempo no passado e sobre a qual várias alterações já se sucederam. Repositórios (ou árvores de diretórios) sob a tutela do Subversion não oferecem apenas a versão mais recente dos dados ali depositados, mas o conteúdo em um dia específico, em uma revisão específica. Você também pode saber quem efetuou as mudanças, a data e quais foram estas.

Tronco (trunk): Já vimos que o repositório armazena e gerencia artefatos ao longo do tempo. É comum denominarmos o conjunto destes artefatos no repositório de tronco (trunk).

Ramo (branch): Em alguns cenários é relevante ter uma "cópia" do tronco, também gerenciada pelo Subversion, de forma que mudanças feitas no tronco afetem apenas o tronco, enquanto aquelas feitas em um ramo (branch) afetem apenas o ramo em questão. Em um projeto de desenvolvimento de software é comum a existência de estados "intermediários" entre o acréscimo ou mudança de um serviço em um software. Se as alterações correspondentes a estes estados forem efetivadas no tronco, possivelmente todos os demais membros de um projeto podem ser negativamente afetados. O emprego de um ramo permite que alguns membros da equipe trabalhem no ramo, façam alterações que não serão efetivadas no tronco enquanto estiverem instáveis, evitando que outros membros sejam afetados. Quando a estabilidade do ramo, provavelmente decorrente do acréscimo de um serviço que exigia várias alterações estiver finalizada, as diferenças entre o ramo e o tronco podem ser estabelecidas e as mudanças transferidas para o tronco de forma controlada. Convém ressaltar que enquanto alguns trabalhavam em um ramo, o restante pode continuar trabalhando no trunk.

Carimbo (tag): Arquivos e diretórios podem ser alterados como resultado da execução de um projeto. Todo projeto tem marcos. Quando desejamos associar um marco atingido com uma determinada revisão, armazenada no repositório, empregamos um carimbo. Toda revisão tem um número único, por exemplo, 261, que certamente não é sugestivo para o fato desta revisão corresponder a versão 1.0 de determinado software. Em casos como este, esta revisão é melhor conhecida e gerenciada através de carimbos. Um carimbo é a associação de um nome a uma revisão.

3. Comandos básicos

Checkout : Consiste em obter uma cópia do repositório central para o seu computador. Para realizar um checkout deve-se ter a URL do repositório.
Ex: http://relprev-022013.googlecode.com/svn/trunk/

Update : Atualizar o repositório local para a versão mais atual do repositório central.

Commit : Salvar alguma alteração e enviar para o repositório central criando uma nova "revision".

Diff : É a diferença entre as "revisions".

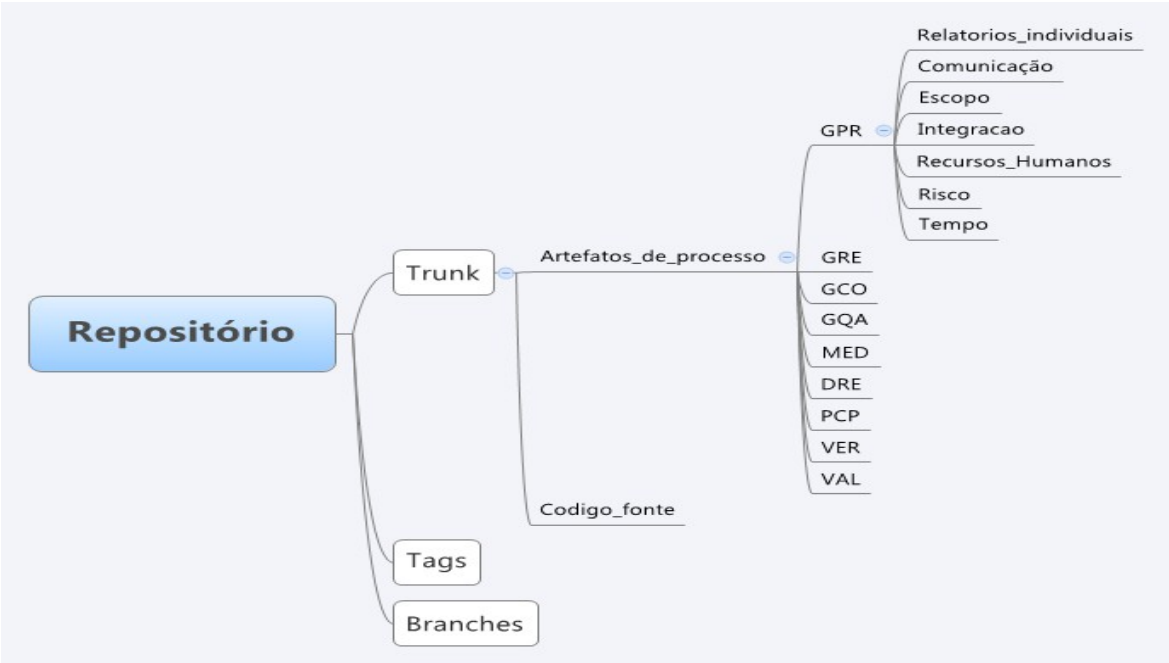
4. Padrão utilizado pela Fábrica de Software do INF-UFG

• **Padrão de mensagem de commit:** A mensagem de commit deve seguir o seguinte padrão:
ref #<número da atividade> <mensagem>

ref # ou refs #: Termo fixo para indicar referência.
<número _da_atividade>: Número da atividade do redmine que é satisfeita pelo commit.
<mensagem>: Mensagem sucinta explicando o motivo do commit
ex.: ref #897 refatoração. Definição da equipe.

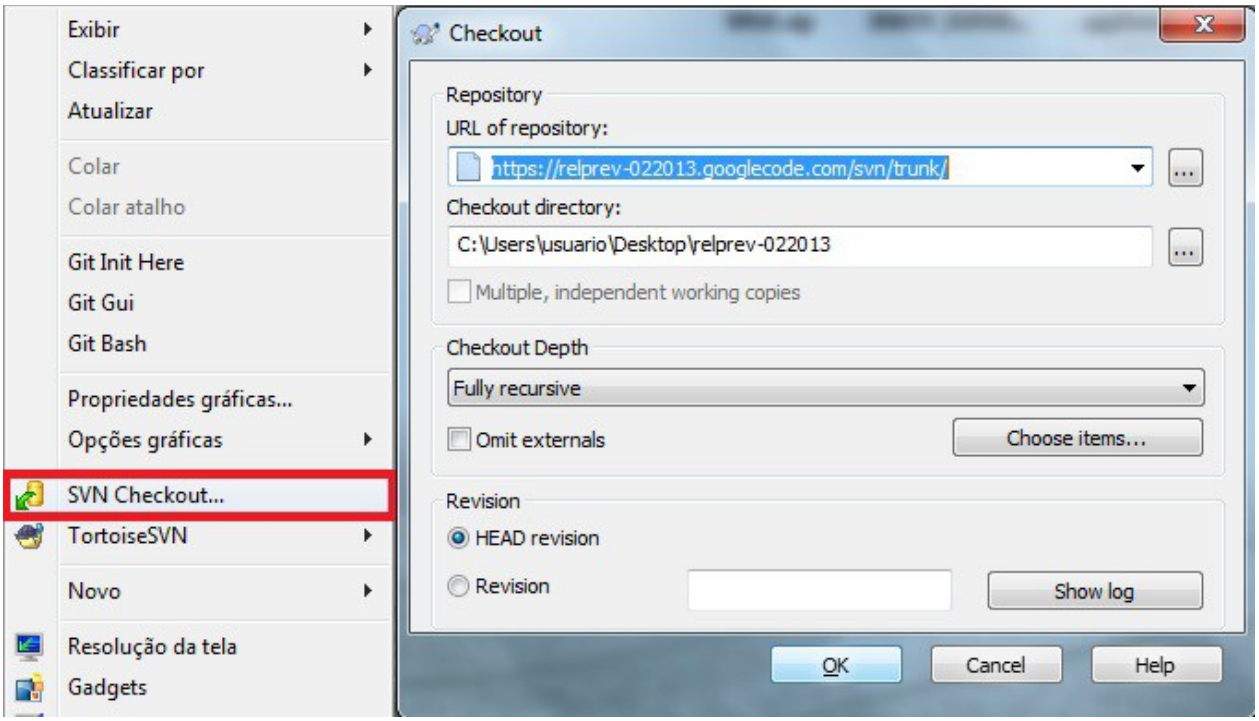
• **Identificação dos itens de configuração:** A identificação dos itens de configuração deve seguir a seguinte regra:
<SIGLA_DO_PROJETO>-<SEMESTRE_INICIO_PROJETO><ANO_INICIO_PROJETO>-<IDENTIFICAÇÃO_GRUPO_PROCESSO><NOME_ARQUIVO>.<EXTENSÃO>
Ex.: RELPREV-022013-GCO-Plano_de_gerencia_configuracao.pdf
O nome dos artefatos não devem possuir espaços.
A regra de identificaçào dos itens de configuração não se aplica a código fonte.

• **Estrutura de diretórios:** A estrutura de diretórios deve seguir o layout abaixo:

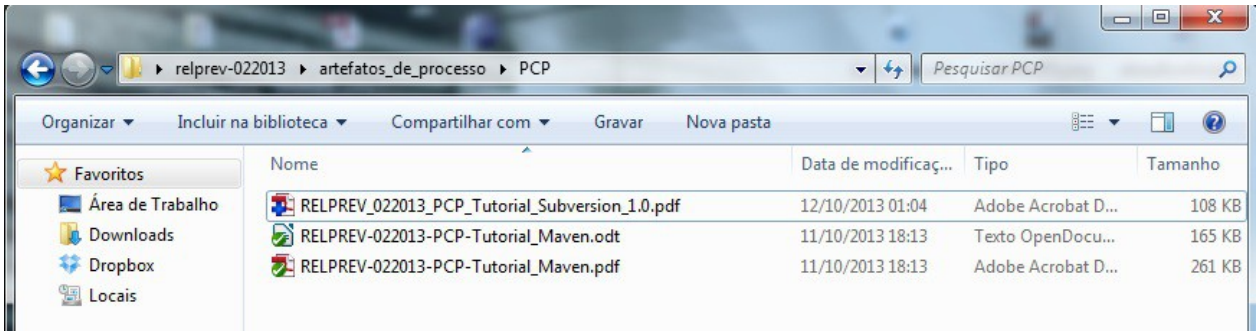


5. Instalação do Tortoise SVN (Client) e exemplo de uso

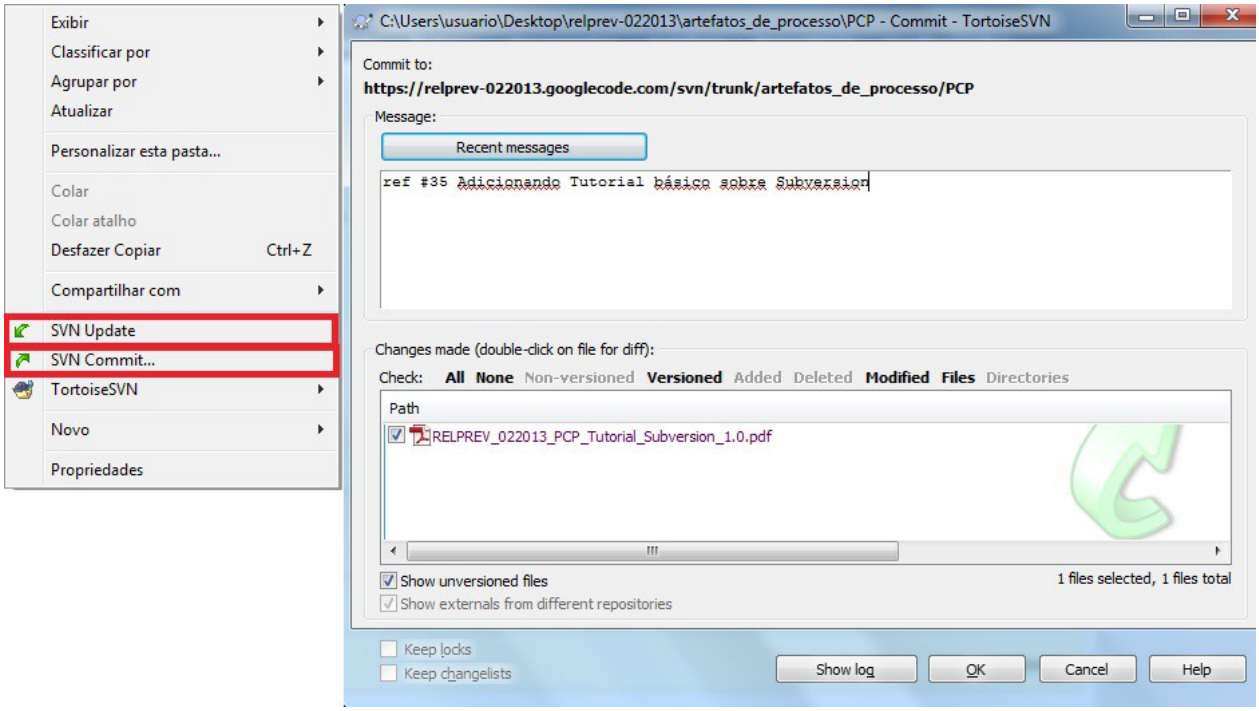
O Tortoise SVN é um client Subversion, com ele será possível realizar o controle de versões com técnicas descritas acima, além de outras técnicas que serão usadas ao longo do projeto. A versão do Tortoise SVN que iremos usar é a 1.8.2, abaixo segue o link para download: http://tortoisesvn.net/downloads.html. Após a instalação o primeiro passo é fazer o checkout do repositório. Para isso escolha um local onde será armazenado o repositório, clique com o botão direito e faça como na imagem abaixo:



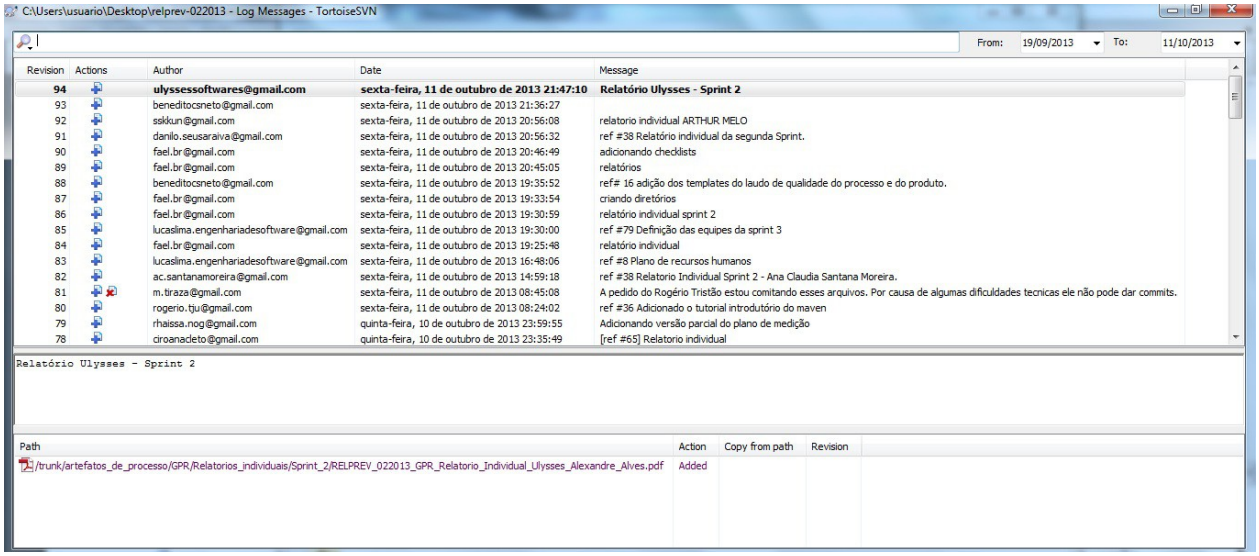
Pronto, está com uma *cópia de trabalho* no seu computador, agora vamos fazer a primeira alteração (Neste caso, adicionando o arquivo PDF como mostra a figura) e enviá-la ao repositório central com o comando Commit:



OBS: É importante lembrar que sempre antes de realizar um *commit* deverá ser feito um *update*, garantindo assim que você irá gerar uma nova revisão a partir da revisão mais recente, evitando possíveis conflitos no repositório. Após clicar em Commit irá abrir uma janela como a da figura onde irá aparecer todos os arquivos afetados e um espaço onde poderá colocar uma mensagem detalhando o commit. (No exemplo segui o padrão de commit da Fábrica de Software já citado neste tutorial).



Outro ponto importante é o log das revisões, um modo de visualizar o log é ir na pasta raiz do repositório, ou em um arquivo apenas, e ir em “Show log”. Irá abrir uma janela semelhante a essa da imagem abaixo com todos os detalhes da revisão incluindo autor, data, mensagem, ações (added,deleted,modified,etc) e o número da revisão.



6. Referências

http://tortoisesvn.net/docs/nightly/TortoiseSVN_pt_BR/tsvn-basics.html

http://svnbook.red-bean.com/en/1.7/svn-book.html