
První vydání

Subversion pro každého

Ladislav Prskavec

Předmluva

Proč bychom potřebovali verzovací systém?

Toto je častá otázka, kterou slyším. Spousta lidí používá sdílené adresáře pro práci s dokumenty. Přejmenovávají soubory a adresáře. Pokud znáte dokumenty typu Projekt-Final-Update!.doc apod. tak víte o čem mluvím.

Každý programátor pracuje v jiné podadresáři?

Tak to trochu může lidem připadat, ale programátoři používají verzovací systém, který dokumentovou databází, která sleduje změny a pomůže v mnoha věcech.

Záloha a Obnova

Soubory, které ukládáme do verzovacího systému si nesou informaci o času uložení a není problém skočit v čase co existuje repozitář kam chcete třeba co jste dělali 29.7.2009? Není problém.

Synchronizace

Pokud si verzujete například konfiguraci nastavení linuxu. Příkady najdete na githubu v repositářích nazvaných dotfiles. Tak si pomocí verzovacího systému lehce udržujete konfiguraci na

všech počítačích, které používáte. Obdobně není problém to dělat s jakýmkoliv dokumenty.

Jednoduché Undo

Modifikujete soubor a nefunguje to a potřebujete vrátit změny zpět. Není problém obnovit poslední dobrou verzi. Maličkost.

Undo

Kdykoliv máte k dispozici undo, pokud například najdete chybu, kterou jste udělali před rokem není problém se vrátit zpět a podívat se jaké změny jste dělali a proč.

Sledování změn

Soubory jsem změněny, ale verzovací systém uchovává také zprávu, kde se často vysvětluje proč změna byla udělaná, dost často obsahuje například odkaz do bug trackeru apod. To jsou věci, které se často ze samotných souborů vyčíst nedají.

Sledování vlastníka

Každá změna je podepsaná a můžete kdykoliv zjistit, kdo který řádek modifikoval. To se hodí pokud potřebujete zjistit kdo změnu udělal.

Pískoviště (sandbox)

Výhoda verzovacích systémů je právě, že můžete pracovat na velkých změnách v izolované oblasti bez toho aby jste se báli, že ovlivníte celek.

Větve a merge

Větve slouží jako velké pískoviště, můžete mít desítky větví, které mohou po dlouhou dobu fungovat izolovaně a později pokud budete chtít spojíte (merge) je kam potřebujete.

Motivace

Sdílené adresáře jsou rychlé a jednoduché, ale rozhodně tyto vlastnosti verzovacích systémů jsou něco co je o dost lepší.

Verzovacích systémů jsou desítky, od SCCS (1972), přes CVS (1990), Subversion (2000) k distribuovaným jako je Darcs (2002), Git (2005) nebo Mercurial (2005). K nejnovějším přírůstkům patří Verocity (2011), které přidává například přímo podporu SCRUMu.

INTERACTIVE Předmluva.1 Historie verzovacích systémů

The screenshot shows a digital timeline interface titled "TIMELINE-VCS". At the top, there are tabs for "CHART" and "LIST", along with a search bar. The main area displays a timeline with various historical milestones. One prominent entry is "Source Code Control System (SCCS) 1972", which is highlighted with a red box. Below this entry is an image of the book cover for "Applying RCS and SCCS" by Marc J. Rochkind. The book cover features a black and white illustration of two raccoons. The timeline continues with other entries, though they are not clearly legible in the screenshot. A navigation arrow is visible at the bottom right of the timeline area.

Úvod

1

Co je Subversion a jak to funguje.

Co je Subversion?

Historie

Subversion vznikl officiálně 31. srpna 2001 a verze 1.0 byla k dispozici v roce 2004.

1.1 září 2004 (FSFS)

1.2 květen 2005 (locks)

1.3 prosinec 2005 (svnserv)

1.4 září 2006 (svnsync)

1.5 červen 2008 (merge tracking support)

1.6 březen 2009

1.7 říjen 2011 (working copy 2)

Subversion

Subversion je nástroj ze skupiny **SCM** (správa obsahu zdrojových kódů - Source Content Management). Nejvíce rozšířený nástroj v minulých letech a částečně i dnes je **CVS**, který má několik zásadních nevýhod, které donutili CollabNet k tomu, aby se vrhli do vývoje Subversion.

Subversion slouží ke zprávě zdrojových kódů, které jsou napsány v jakémkoliv jazyce jak programovacím tak lidském. Nativně podporuje kódování UTF-8 a proto není problém psát dokumenty anglicky, rusky nebo svahilsky. SVN uchovává vytváří revize, to znamená, že uchovává stav dokumentů z doby, kdy jste dali vykonat příkaz commit, který slouží k odeslání zdrojových dat na server. Můžete si pomocí Subversion spravovat verze textového dokumentu jako je například tato kniha psaná v Docbooku a nebo zdrojové kódy v libovolném programovacím jazyce (třeba i vlastním).

Subversion patří k centralizovaným SCM systémům jako je CVS, Perforce, Clearcase a další. Existují také distribuované SCM systémy, které nemají klasickou architekturu client/server. Například Bazaar, Mercurial nebo Git. Porovnání a odkazy na jiné SCM najdete na stránkách Better SCM

Jak funguje Subversion

Architektura

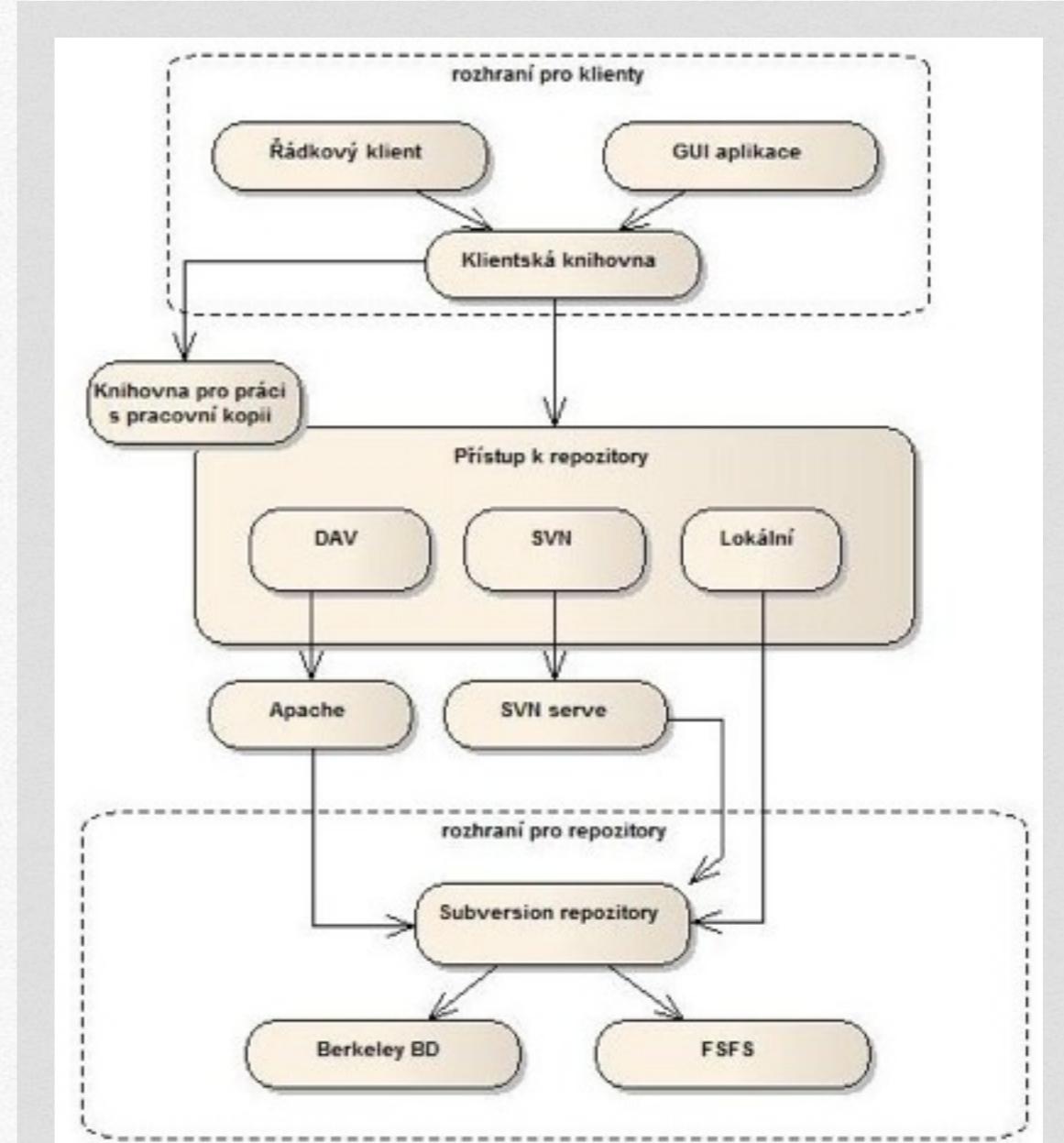
Subversion má architekturu client server. [Obr. 1.2.1](#) znázorňuje jak je to celé uspořádané. Vlastní repository na serveru může mít uložitě buď v Berkley DB a nebo ve formátu FSFS. FSFS je nyní standard a Berkley DB se používalo v minulosti. Mezi jednotlivými verzemi se také stále ukládání do **FSFS** vylepšuje, aby databáze zabírala méně místa a byly revize efektivněji ukládané.

Vlastní přístup k repository se řídí způsobem jak server provozujete. Základem je protokol **DAV** (http, https), **SVN** nebo lokální.

Klientská část umí pracovat přímo s repository i s pracovní kopií (working copy) a podporuje všechny zmíněné protokoly.

Protokoly a jejich možnosti

V následující části porovnáme možnosti a doporučím vám jak při volbě postupovat. Z mých zkušeností je potřeba zvážit něko-



OBR. 1.2.1 Architektura Subversion

lik okolností a volit něky kompromis mezi rychlostí a bezpečností.

Autentizace

Apache + mod_dav_svn	svnserve	svnserve ssh
HTTP(S), X.509, LDAP, NTLM, a další dostupné mechanismy v Apache httpd	CRAM-MD5, LDAP, NTLM, a jiné dostupné mechanismy SASL	SSH

Uživatelské účty

Apache + mod_dav_svn	svnserve	svnserve ssh
Soubor s uživateli, případně jiný mechanismus dostupný v Apache httpd (LDAP, SQL, etc.)	Soubor s uživateli, případně mechanismy dostupné v SASL (LDAP, SQL, etc.)	Účty v operačním systému

Autorizace

Apache + mod_dav_svn	svnserve	svnserve ssh
Práva pro čtení a zápis se týkají celého repozitáře nebo cesty.	Práva pro čtení a zápis se týkají celého repozitáře nebo cesty.	Práva pro čtení a zápis se týkají celého repozitáře

Directiva SVNPathAuthz velmi sníží rychlosť protože, se musí kontrolovať cesta a ne len repository, proto s touto volbou opatrne a používajte ju len ak potrebujete.

Šifrování

Apache + mod_dav_svn	svnserve	svnserve ssh
SSL	SASL	SSH

Logování

Apache + mod_dav_svn	svnserve	svnserve ssh
apache log, custom log	bez logování	bez logování

Custom log umí zaznamenávať jednotlivé akcie SVN klienta.

Interoperabilita

Apache + mod_dav_svn	svnserve	svnserve ssh
WebDAV klienti	SVN klienti	SVN klienti

Webový klient

Apache + mod_dav_svn	svnserve	svnserve ssh
jednoduchý náhled nebo ViewVC	ViewVC a podobné	ViewVC a podobné

Master slave replikace

Apache + mod_dav_svn	svnserve	svnserve ssh
mod_proxy, svnsync	svnsync	svnsync

Rychlosť

Apache + mod_dav_svn	svnserve	svnserve ssh
Pomalé, limitované http	rychlé	rychlé

Základní nastavení

Apache + mod_dav_svn	svnserve	svnserve ssh
komplexní	jednoduché	složitější

V těchto porovnání není lokální přístup a to z toho důvodu, že ten se dá použít jen ve specifickém případě a to když je klient a server na jednom stroji. Přístup obejde zabezpečení a řídí se právy na filesystému.

Lokální `file:///` protokol se dá dobře využít například na **CIE** pokud na něm máte i repository a zrychli to celou práci s ním.

Základní pojmy

Repository (repozitář, centrální úložiště)

Umožňuje organizovat projekt a spravovat jeho verze. Fyzicky je uloženo na souborovém systému serveru. K repository se přistupuje přes Repository Access Layer (RA) systému Subversion a jeho správa se provádí klientskými nástroji.

Branch (větev)

Slouží k organizaci repository, jedná se o jakousi analogii s adresáři. Pokud se z repository vyzvedne větev, na klientovi vznikne adresářová struktura, která přesně odpovídá větvím v repository.

Revision (revize)

Revize je pořadové číslo každé změny. Slouží ke sledování změn ve větvích v čase. Každá změna v nějaké větvi vytvoří no-

vou revizi v rámci celé repository. Revize obsahuje informace o tom, co bylo změněno, kdo změnu provedl, poznámku a čas.

Pracovní kopie (working copy)

Kopie dat z určité větve z repository v aktuální revizi na pevný disk lokálního klienta. Do pracovní kopie je možné provádět změny, které je možné commitem uložit zpět do repository.

Commit

Odeslání změn provedených od posledního commitu do repository. Commit je nejčastěji používaná změna při práci s repository. Pokud se provádí commit celé pracovní kopie, jedná se o atomickou operaci, jsou odeslány veškeré změny ve všech objektech ve správě verzí; pokud dojde k nějaké chybě při přenosu, není commit pro ostatní uživatele repository zviditelněn, není vytvořena nová revize.

Konflikt

Konflikt je stav, který signalizuje, že stejný objekt, který má být právě commitován, byl změněn někým jiným a nachází se v repository v aktuální revizi v jiné podobě, než jaký je v pracovní kopii. Nelze provést commit celé pracovní kopie, pokud se v ní nachází jeden nebo více souboru v konfliktu.

Cheap-copy

Technika, kterou se realizují kopie prováděné v rámci repository. Objekty nejsou v repository fyzicky duplikovány, ale jsou vytvořeny tzv. odkazy (link) na kopírované objekty. Zjednodušeně lze chápat takovýto link jako informaci o URL s číslem revize. Díky tomu má SVN nízké nároky na datový prostor.

Workflow

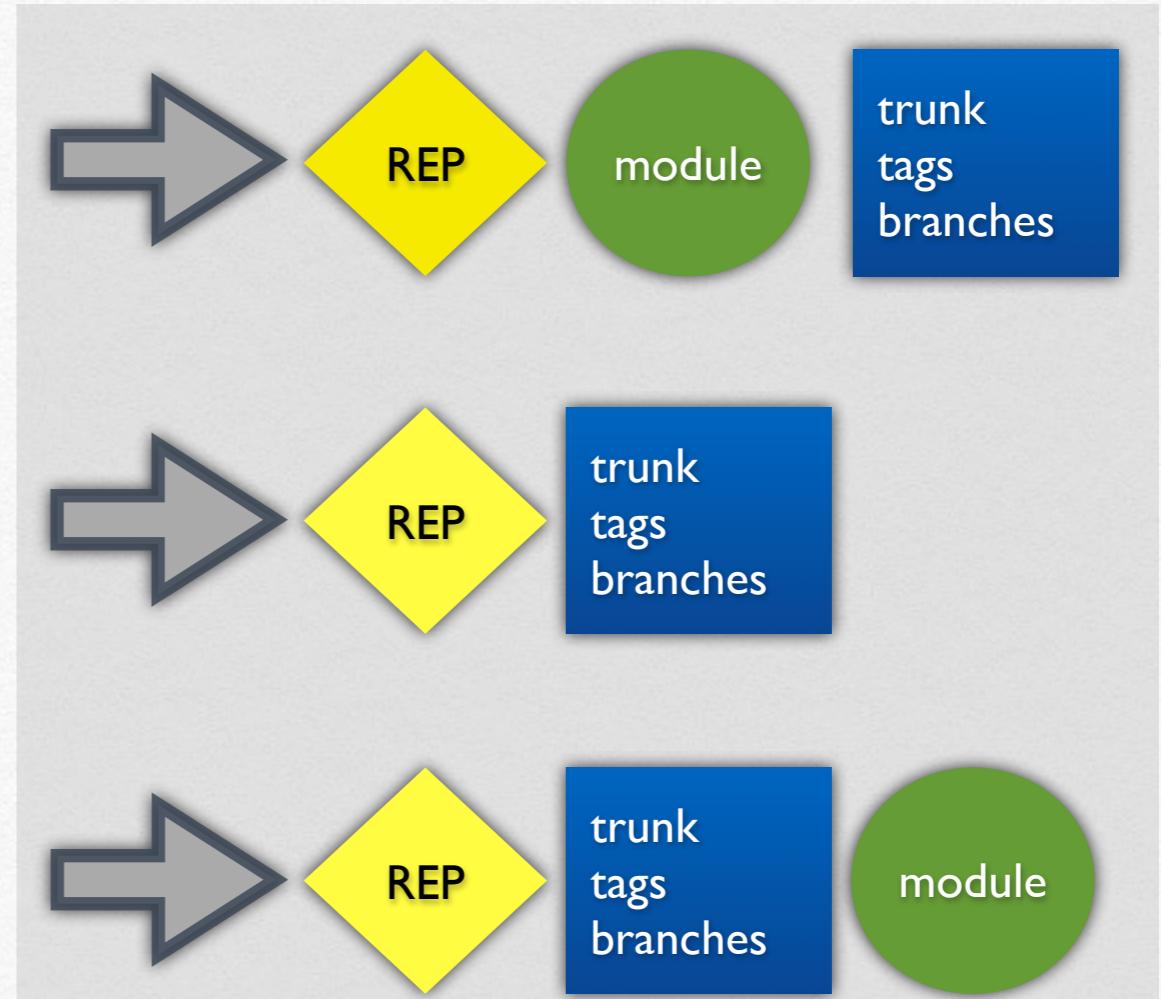
Doporučená struktura repozitory

Struktura repository, musí respektovat workflow a obsah projektu. Zavisí na tom zda máte hodně malých projektů nebo jeden velký.

Na [Obr. 1.4.1](#) jsou naznačeny tři možné struktury repozitory. Adresáře trunk, tags, branches jsou nutné pro správnou funkci SVN i když se mohou jmenovat jak uznáte za vhodné. Ale je zvykem je takto pojmenovat, usnadní vám to spolupráci s ostatními vývojáři všude po světě.

Moduly se to hodí například pro zákaznické projekty, které jsou rozčleněny podle domén nebo jmen zákazníků.

Většina vývoje jde lineárně v čase a postupně děláte vývoj v trunku a případně nové verze vydáte pod tagem a pracujete dál. Pokud chcete pracovat paralelně na více projektech použijete větve (branches).



OBR. 1.4.1 Struktura repository

Pokud vytváříte nový projekt měli by jste si strukturu předem rozmyslet, případně se držet doporučení. Struktura se dá měnit i později, ale můžete přijít o historii pokud přesouváte adresáře.

Instalace

Začít je potřeba instalací.



Získáme Subversion

Subversion je k dispozici ve formě zdrojových kódů nebo ve formě binárních balíčků pro různé operační systémy. Subversion je dostupný pro Windows, Mac OS, různé distribuce linuxu i pro některé unixy. Veškerý potřebný software stáhněte na stránkách <http://subversion.apache.org>

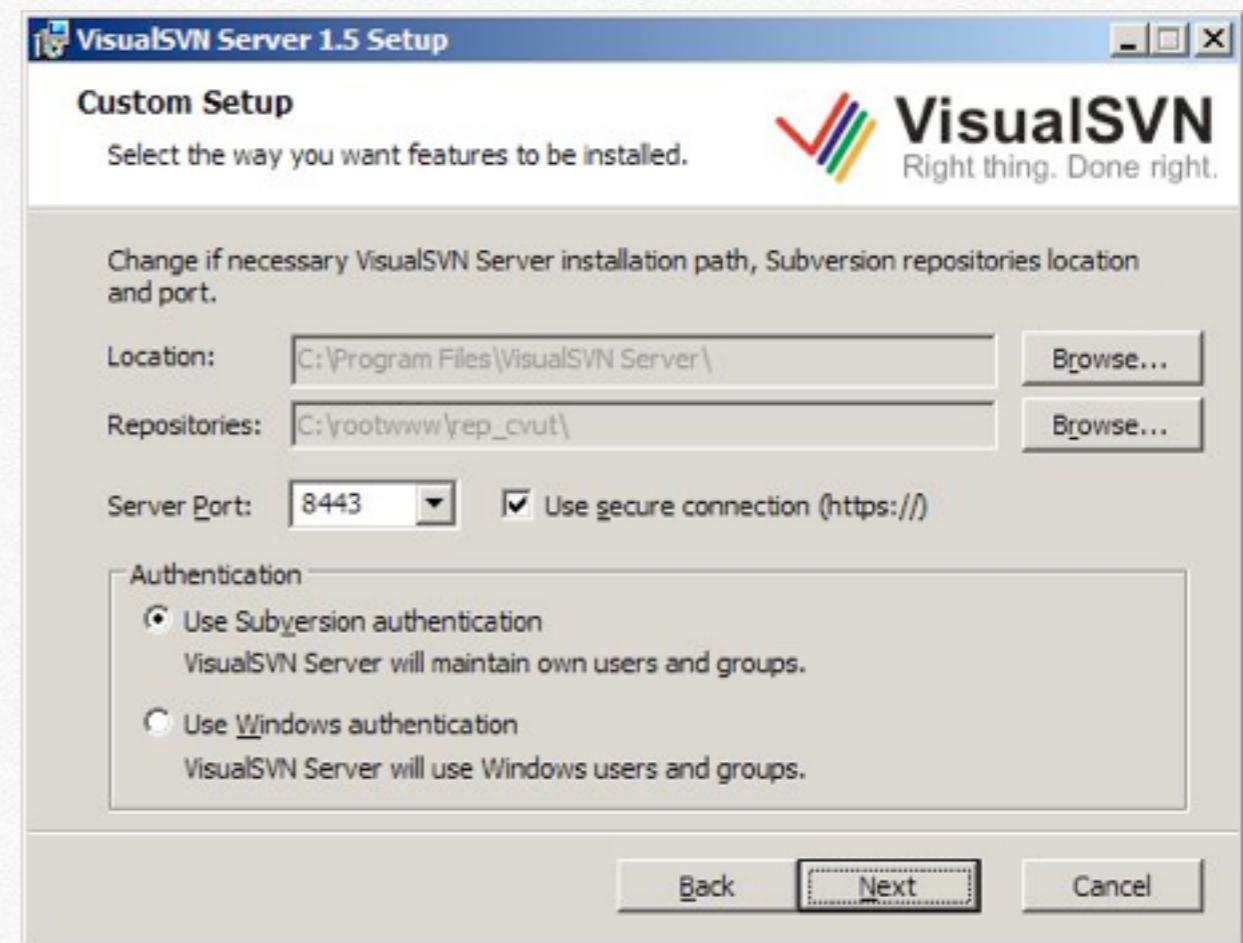
Windows

Ve Windows je několik programů, které si můžete pro práci zvolit a to buď klasického řádkového klienta a server s podporou svnserv. Nebo jako klienta použijete TortoiseSVN (TSVN), který umožňuje běh jako klient i server (s omezením na lokální přístup k repozitory). Pokud chcete přistupovat přes http a https doporučil bych k instalaci VisualSVN Server.

Pokud nepotřebujete, aby někdo přistupoval k vašim lokálním repozitory doporučuji T SVN. Tento program umí všechny funkce a má pěkné prostředí, které se integruje do Windows Exploreru. Je nejpokročilejší klient, který umožňuje i tvorbu hooks skriptů na straně klienta.

VisualSVN server

VisualSVN je jen Apache server s mod_dav_svn doplněný administrativní konzolí a webovým prohlížečem repozitory.



OBR. 2.1.1 VisualSVN server instalace

Instalace je VisualSVN serveru je jednoduchá. Stáhněte si instalátor, který vás pomocí wizarda provede. Můžete navolit kam se server nainstaluje i kde máte adresář s repozitory. Můžete také použít kromě Subversion autentifikace i Windows autentifikaci. Volitelné je i https připojení.

Nastavení práv pro jednotlivé repozitory se dá provést pomocí administrační konzole. VisualSVN také disponuje webovým prohlížečem repozitory, který je dostupný na adrese <https://localhost:8443/> pokud to máte v základním nastavení lokálně, jinak na příslušné adrese a portu.

Pro nastavení práv slouží VisualSVN Server Manager, kde máte vidět jak repozitory, tak uživatele a skupiny. Potom přes pravé tlačítko a volbu Security nastavíte práva bud' pro všechna repozitory najednou nebo u každého zvlášť.

Linux

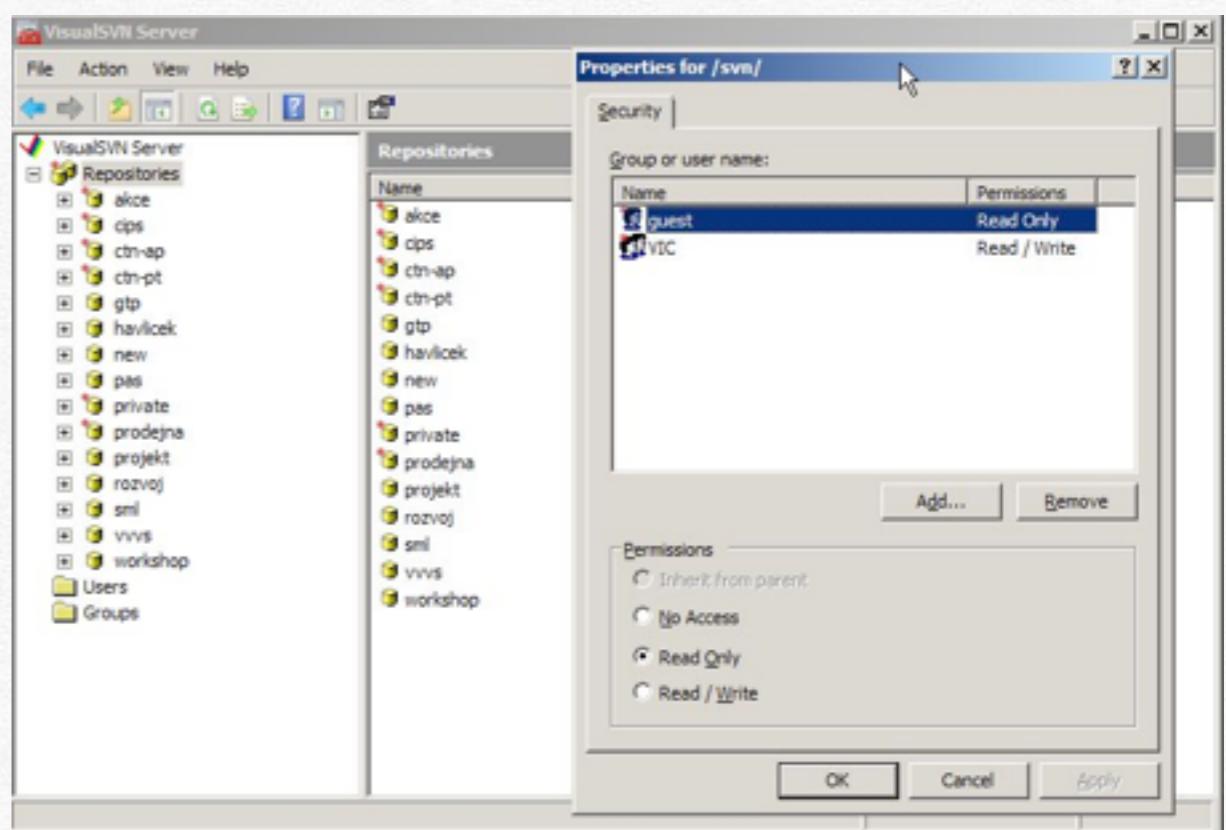
Pro instalaci na Ubuntu použijte zdroje přímo z repozitory.

```
sudo apt-get install subversion  
sudo apt-get install subversion-tools
```

Balík subversion obsahuje jak klienta tak server (svnserv). V balíku subversion-tools jsou utility (svn-backup-dumps, svn-clean, svn2cl, mailer apod.)

Já používám svn2cl pro generování changelogů, mailer používám v post-commitech, abych věděl co se děje v repozitářích.

Pro přístup lze použít `file:///`, `svn://` nebo `svn+ssh://`. Pokud chcete použít `http://` nebo `https://` musíte to udělat přes Apache a WebDAV.



OBR. 2.1.2 VisualSVN admin konzole

```
sudo apt-get install libapache2-svn
```

Konfigurace se potom provede v `/etc/apache2/mods-available/dav_svn.conf`. Ukázka je s přepínačem `SVNListParentPath`, kdy nám dovolí přístup ke všem repositórii, které máme v adresáři `/srv/svn`.

```
<Location /svn/>
    DAV svn
    SVNPath /srv/svn
    SVNListParentPath On
    AuthType Basic
    AuthName "Test subversion repository"
    AuthUserFile /etc/subversion/passwd
    Require valid-user
</Location>
```

REVIEW 2.1 Instalace

Question 1 of 2

Který zkratka odpovídá přístupu pomocí DAV

- A. svn
- B. svn+ssh
- C. https
- D. file



Check Answer



Začínáme

Základní funkce A-Z.



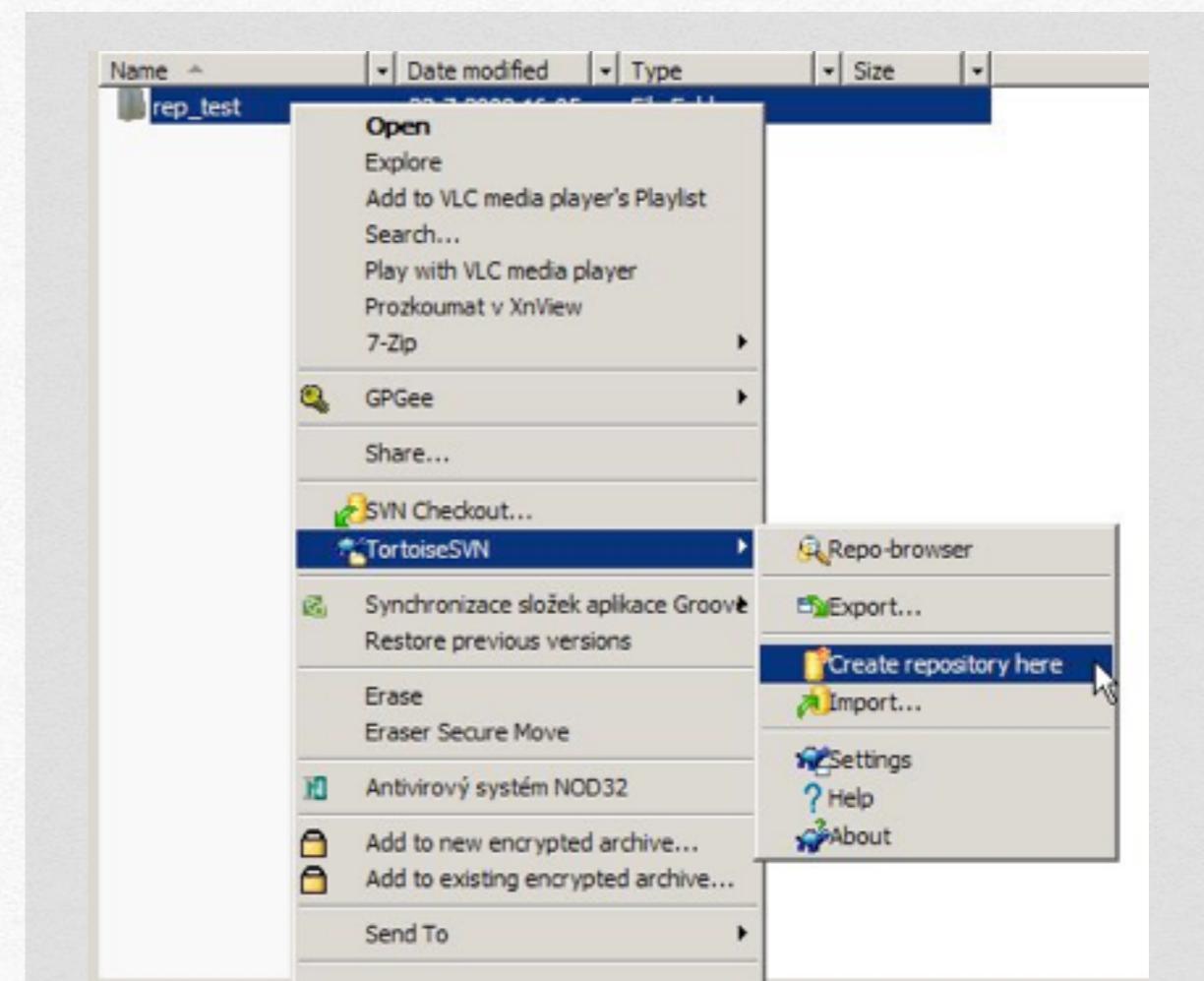
TortoiseSVN

Vytváříme repozitory

Vytvoříme adresář pro repozitory např. c:/rep a v něm adresář pro projekt např. rep_test a na adresáři přes pravé tlačítka se dostaneme do menu TSVN, kde před volbu Vytvořit repozitory zde (Create repository here) vytvoříme repozitory jak ukazuje obrázek. Po úspěšném vytvoření repozitory si můžeme prohlédnout adresář a uvidíme soubory co vytvořil Subversion.

Struktura repozitory je na dalším výpisu. V adresáři conf/ jsou konfigurační soubory pro přístup k repozitory, v adresáři db/ jsou uložená data a souboru current je aktuální revize. V adresáři hooks/ jsou šablony pro hooks skripty, které umožňují vykonat nějakou akci před určitou akcí v Subversion.

Po vytvoření repozitory je dobré otevřít repozitory browser a vytvořit základní strukturu pomocí přímaku vytvořit adresář "create folder".



OBR. 3.1.1 Vytvoření repozitory

V repozitory můžete vytvořit vlastní adresáře, ale doporučuje se nejdříve vytvořit strukturu do které naimportujete svoje adresáře.

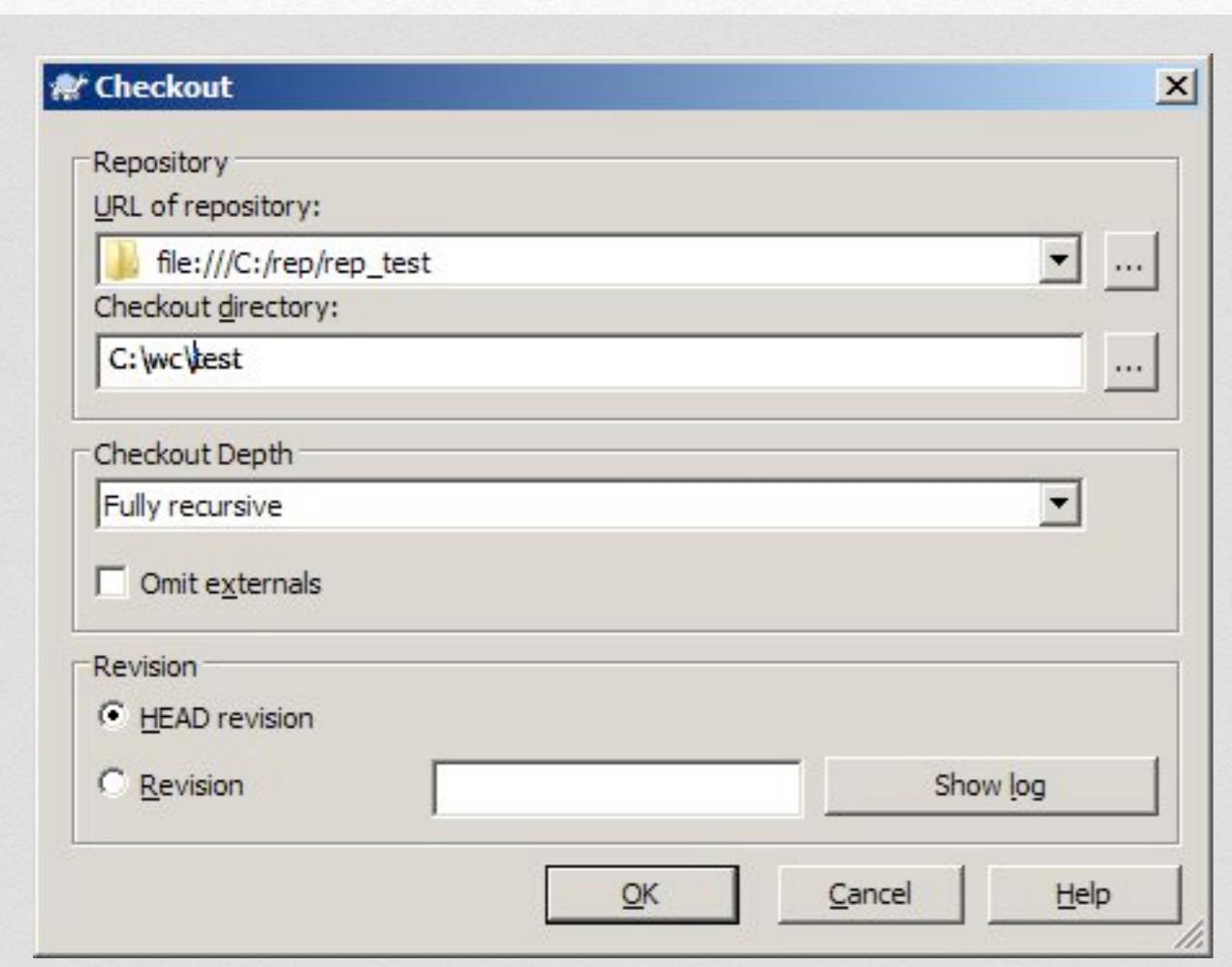
Struktura slouží k tomu, abychom vedli hlavní kmen projektu (trunk) a mohli také vést větve (branches) a dělat nálepky (tags) pro určité vydané verze.

Import zdrojových kódů

Pokud už máme něco co chceme do repozitory importovat uděláme to pomocí příkazu import. Jinak můžeme tento krok přeskočit, protože strukturu můžeme vytvořit pomocí repozitory browseru.

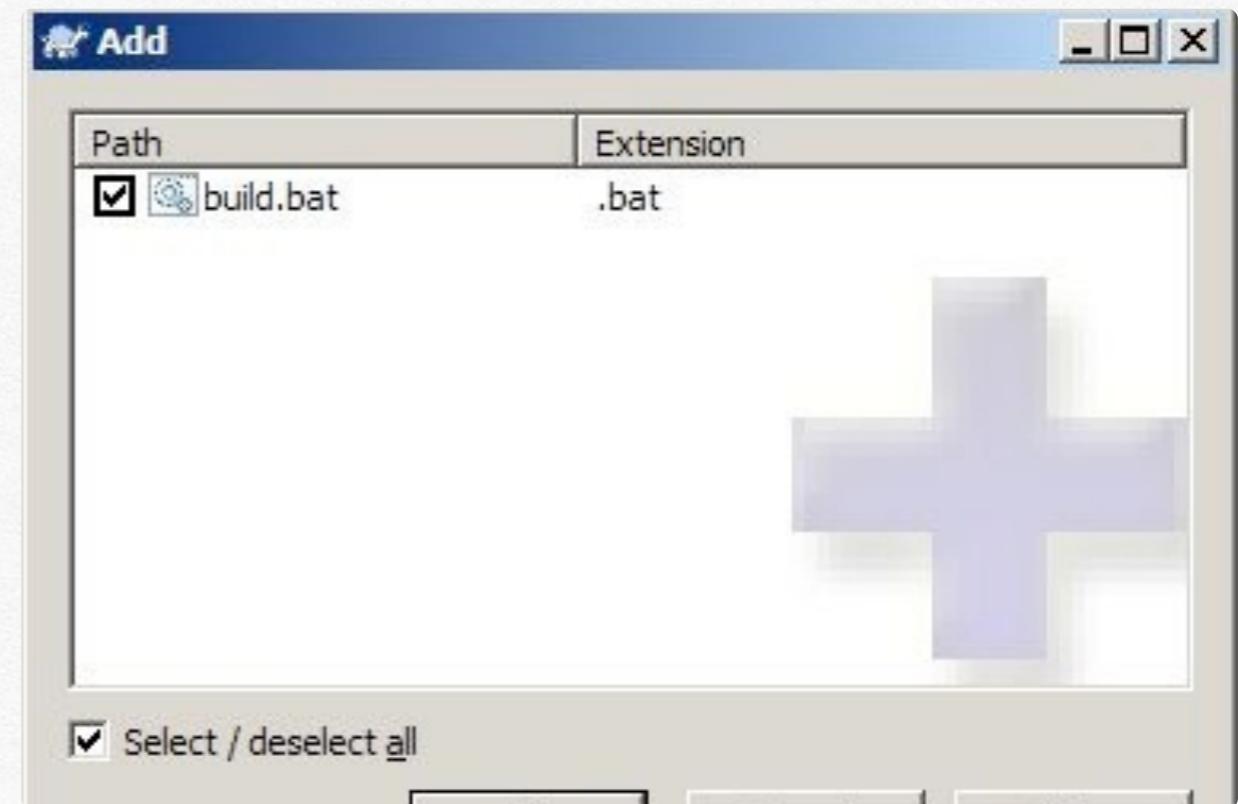
Vytváříme pracovní kopii

Provedeme checkout na repozitory a začneme pracovat.



OBR. 3.1.2 TSVN Checkout

GALLERY 3.1 TortoiseSVN



Příkaz svn add slouží k přidaní souborů do pracovní kopie, aby jste je mohli commitovat do repository.



Subversion řádkový klient

Ve stručnosti zopakuji stejné postupy jako ve TSVN.

Vytvoříme repozitory

```
svnadmin create --fs-type fsfs c:/rep/rep_test
```

Rozdíl mezi Windows a Linuxem je jen ve způsobu zadávání cesty k repozitory. Obdobně to můžete aplikovat na všechny příklady. SVN se ve funkčnosti mezi Windows a Linuxem nijak neliší.

```
svnadmin create --fs-type fsfs /srv/svn/rep_test
```

V řádkovém klientu nemůžeme vytvořit adresáře přímo, ale uděláme je rovnou při prvním importu souborů, tento postup lze použít i přes **TSVN**.

```
cd c:/tmp  
mkdir test  
cd test  
mkdir trunk  
mkdir tags  
mkdir branches  
svn import --message "Initial import"  
file:///C:/rep/rep_test
```

Potom vytvoříme pracovní kopii

```
svn checkout file:///C:/rep/rep_test/trunk  
c:/wc/test
```

Běžný pracovní postup

Nejdříve si aktualizujte svoji pracovní kopii.

```
svn update
```

nebo

```
svn up
```

Skoro každý příkaz má zkratku, která se hodí aby jste to nemuseli rozepisovat. Stačí se podívat na svn help a uvidíte je.

Potom vytvoříte nové soubory, smažete staré soubory, změníte obsah. Pomocí příkazu svn diff si ověříte můžete zobrazit změny své pracovní kopie proti verzi v repozиторiu.

```
svn diff
```

Pokud jste pracovali s pracovní kopii a přidávali nebo mazali soubory, je dobré vědět zda jsou již přidané či v jakém stavu jsou to zjistíme pomocí svn status a potom můžeme chybějící soubory např. pomocí svn add přidat.

```
svn status
```

Změny odešlete je pomocí commitu do repozitory.

```
svn commit -m "Text zprávy"
```

Postup se opakuje, jen update nemusíte dávat pokaždé pokud jste si jistý, že nikdo s repozitory mezikm nepracoval. Případně můžete zamknout repozitory během práce na změnách. Pokud nenastane konflikt vystačíte si s tímto jednoduchým postupem.

Pokud dojde ke konfliktu můžete změny spojit (merge) nebo svoje vrátit pomocí svn revert.

Export pro hosting, ftp apod.

Pokud by vám vadí adresáře, které svn vytváří automaticky a uchovává v nich svoje informace, tak těch se zbavíte pomocí exportu.

Pokud chcete do aktuálního adresáře všechny soubory vyexportovat stačí zadat tento příkaz a soubory se tam uloží. Dá se také pracovat s pracovní kopii nebo zadávat i cestu pro export.

```
svn export file:///C:/rep/rep_test/trunk
```

Pokud budete chtít vyexportovat jen soubory co se změnili v poslední revizi bude to trochu složitější a budete potřebovat zjistit změněné soubory pomocí příkazu svnlook.

```
# změněné soubory  
for i in $( svnlook changed -r $REV $REPOSPATH ) ;  
do  
    if [ "${#i}" -gt "2" ]  
        then pathexport $REPOS $i  
    fi  
done
```

Pokročilejší funkce

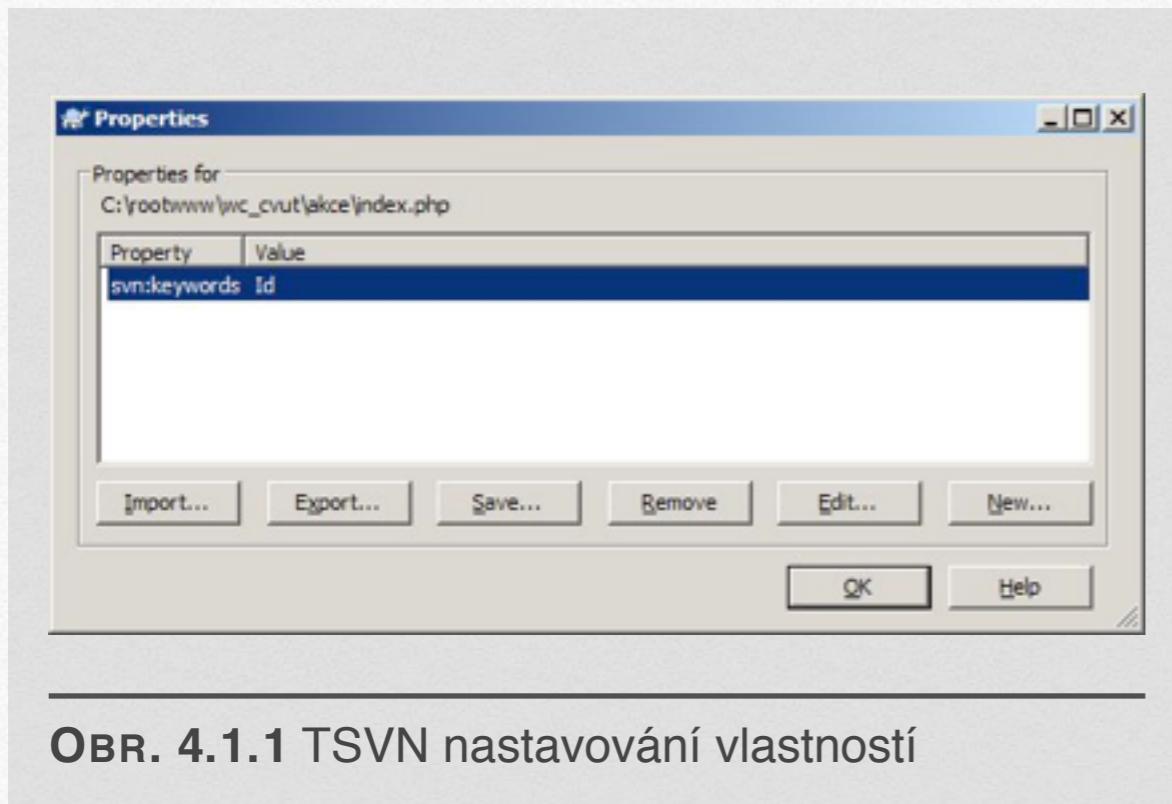
Nastavování vlastností,
větve a tagy.



Vlastnosti

Vlastnosti (properties) umožňují uchovávat další informace o jednotlivých souborech. Nastavují se automaticky nebo ručně.

Mezi ty automatické patří od verze 1.5 takzvaný "merge tracking". Někteří klienti ho umí používat a zobrazují potom informace o tom kde co bylo spojeno.



OBR. 4.1.1 TSVN nastavování vlastností

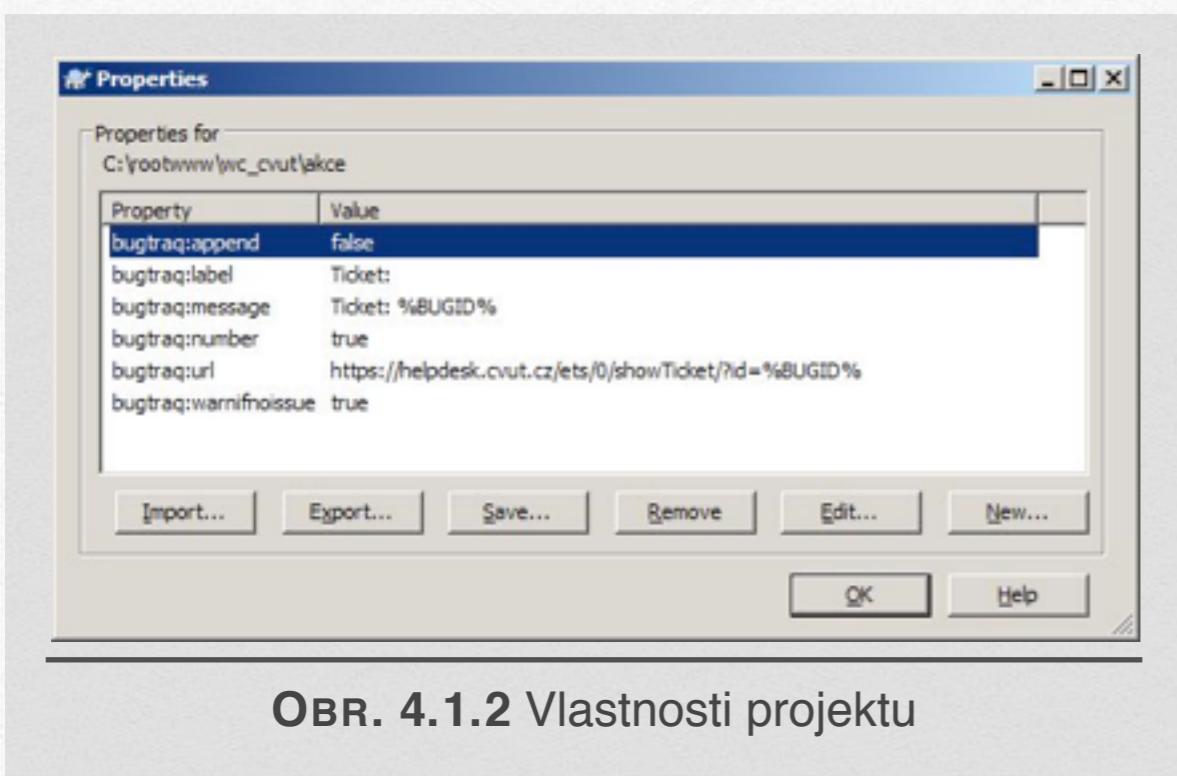
Určitě si každý kdo listoval v projektech, které jsou zpracovávány pod Subversion, v hlavičce informace o verzi a autorovy (\$Id: svn-kniha.xml 50 2009-07-20 15:25:15Z abtris \$). Tato informace je generována automaticky pomocí vlastnosti svn:keywords=Id. Kromě Id může tato vlastnost nabývat více možností a to Author, Date, Revision a HeadURL. Pro verzi dokumentu se ale obvykle používá Id.

```
<?php
/**
 * Pridani obecne
 * @version SVN: $Id: svn-kniha.xml 50 2009-07-20 15:25:15Z abtris $
 * @package Akce2008
 * @filesource
```

Příklad použití v PHP

Vlastnosti projektu

U projektů se dá nastavit například integrace s systémem pro správu chyb (bug, issue tracker) a může být zcela libovolný, klidně jednoduchá aplikace napsaná v **PHP** nebo i profesionální jako je **JIRA**.



OBR. 4.1.2 Vlastnosti projektu

Branches a tags - větve a štítky

Tagy slouží pro zachování určité verze repozitory v čase. V angličtině tomu říkají snapshot. Udělá se jednoduše kopie vaší verze kterou chcete označit štítkem.

```
svn copy file:///C:/rep/rep_test/trunk
file:///C:/rep/rep_test/tags/verze-1 \
-m "Verze 1 mého zkušebního projektu."
```

Tagy nemusí představovat nutně jen kopii v repozitory, ale můžete zachovat i například současnou pracovní kopii (wc). To

se hodí pokud je projekt obsáhlý nebo projevuje chyby na které teď nemáte čas, ale chcete se k nim někdy vrátit.

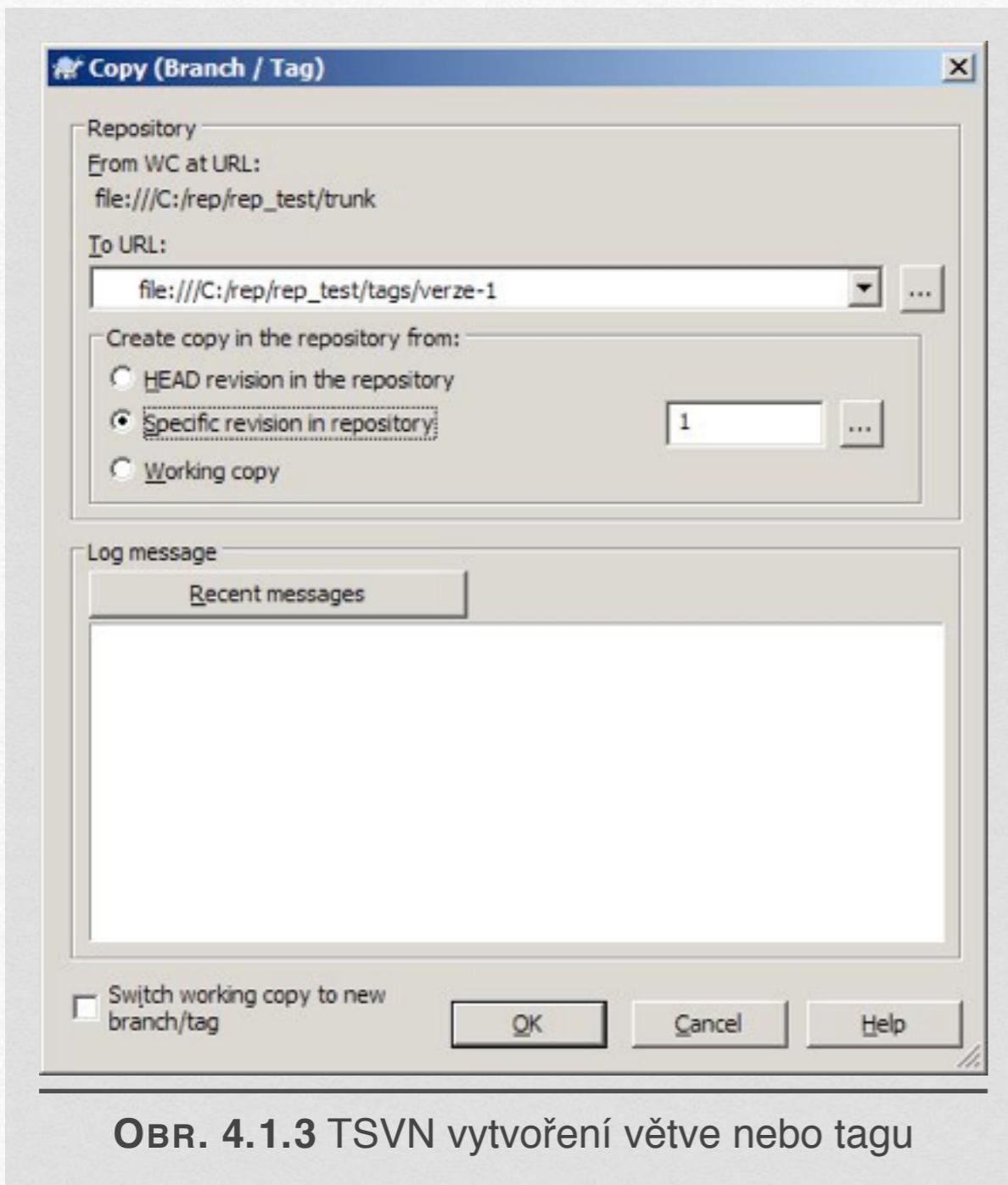
```
svn copy c:/wc/test
file:///C:/rep/rep_test/tags/problematicka-wc \
-m "Kopie mojí wc."
```

V TSVN je situace jednoduchá tam máte příkaz Branch/tag pomocí kterého vytvoříte příslušný tag nebo branch (větv). Jak je vidět z následujícího obrázku máte k dispozici několik voleb, které vám dají možnost vytvořit kopii z HEAD (poslední verze v repozitory) nebo s libovolné verze kterou specifikujete číslem revize a nakonec z příslušné pracovní kopie projektu. Dole pak je volba, která umožní vám nejenom udělat kopii, ale také přepnout WC na vámi udělaný branch/tag. To se používá pokud vytváříte branch a chcete v ní pokračovat.

Pokud už víte co jsou tagy také vás nepřekvapí, že branches jsou v rámci repozitory jsou to samé jen se s nimi trochu jinak pracuje. Tagy slouží k zaznamenání určité 1 verze, ale branch slouží k tomu aby se dále upravovala, opravovala apod.

Uvedu jeden příklad např. projekt má 2 verze. Na verzi 2.0 se pracuje v /trunk a verze 1.0 je vydána v /tags/v1-0. Ted' ale někdo najde ve verzi 1.0 chybu tak se vytvoří branch v /branches/1-0/ a pracuje se na odstranění chyb, udělá se několik commitů, ty se otestují a když je vše v pořádku vydá se

verze v tags /tags/v1-1. Nezávisle na tom se pracuje na verzi 2. Možných schémat práce s větvemi je několik, toto byl jen příklad.



OBR. 4.1.3 TSVN vytvoření větve nebo tagu

Migrace repozitory

Celé repozitory můžete uložit do souboru.

```
svnadmin dump repository >file
```

Celý soubor jde stejně dobře nahrát.

```
svnadmin load repository < file
```

Ve windows můžete udělat export všech repozitory např. pomocí tohoto skriptu (přepínač –incremental revize jako diff proti předchozí verzi).

```
dir c:\rep /b /o /AD >dir1.txt
```

```
FOR /F %%M IN (dir1.txt) DO ( ECHO %%M svnadmin  
dump c:/rep/%%M --incremental >%%M.dmp zip -m  
-9 -g %%M.dmp.zip %%M.dmp )
```

Mirror repository

Mirror repository pomocí svnsync

Pokud potřebujete read-only mirror, např. kvůli veřejné dostupnosti v případě Open Source projektů nebo jako zálohu má SVN od verze 1.4 nástroj svnsync.

V praxi potřebujete nejdříve mirror inicializovat a potom pravidelně spouštět synchronize, nejlépe post-commitem nebo cronem.

Pokud děláte mirror z nějakého staršího svn repozitory (existuje již několik let) tak můžete narazit na několik problémů s kterými poradím co dělat.

Inicializace mirroru

```
svnsync init file:///srv/svn/rep-mirror  
https://svn.firma.cz/svn/rep-zdroj \  
--source-username svnuser --source-password svnpass \  
--sync-username svnmirroruser --sync-password svnmirrorpass
```

```
--sync-username svnmirroruser --sync-password svnmirrorpass
```

Vlastní synchronizace

```
svnsync synchronize file:///srv/svn/rep-mirror \  
--source-username svnuser --source-password svnpass \  
--sync-username svnmirroruser --sync-password svnmirrorpass
```

Řešení možných problémů

Příkazy je lepší psát na jeden řádek, ale v linuxu by nemělo vadit ani rozdělení na více. U synchronizace mi to vyhodilo chybu a byl jsem nucen to dát na jeden řádek. Tady to pro přehlednost nechávám.

```
svnsync: Cannot accept 'svn:log' property because it is not encoded in UTF-8
```

Tato chyba bývá způsobena špatnými znaky v logu např. češtinou z windows kódování a je potřeba opravit záznam v svn:log.

Oprava log message se provede bud' pomocí svn propset -r111 --revprop svn:log -F fixedlogfile nebo svnadmin setlog /srv/svn/

rep-zdroj -r 111 fixedlogfile. Pro svn propset musíte mít povolený a nastavený pre-revprop-change hook skript.

svnsync: Cannot accept non-LF line endings in
'svn:log' property

Pokud se vyskytne tato chyba je potřeba upgradovat SVN server na verzi 1.6.3, kde už umí převádět konce řádků automaticky.

Merge v příkladech

Příkaz merge slouží ke spojování zdrojových kódů, v praxi většinou nastávají tyto případy:

Merge změn z nějaké branche do trunku.

Pokud jste to opravili na jednom místě takto to přenesete i jinam.

```
svn merge --reintegrate  
file:///C:/rep/rep_test/branches/test
```

Reverzní merge

Návrat ve trunku nebo věti k předchozí verzi. Vrátíte zpět změny, které jste udělali mezi revizemi HEAD:80.

```
svn merge -r HEAD:80  
file:///C:/rep/rep_test/trunk
```

Kompletní merge do pracovní kopie.

Pomocí syntaxe URL@REV se dá přímo zadat revize. První URL (levá strana pro porovnání) a druhé URL (pravá strana pro porovnání) a cíl (pracovní kopie).

```
svn merge  
file:///C:/rep/rep_test/branches/test@150 \  
file:///C:/rep/rep_test/branches/jiny-test@212  
\ c:/wc/test
```

Pokud provádíte merge z příkazové řádky, hodí se také přepínač --dry-run s kterým si merge vyzkouší nanečisto. Informace o merge se dá zjistit také pomocí příkazu svn merge-info.

Integrace SVN s editory a IDE

Subversion má vynikající integraci s mnohým software a IDE, to na něm bude oceňovat.



Eclipse

Eclipse má nativně podporu jen pro CVS, pro Subversion musíte nainstalovat plugin Subclipse nebo Subversive, který používá například Zend Studio for Eclipse.

Po delší zkušenosti s Zend Studiem doporučuji všem spíše Subversive.

The screenshot shows the Eclipse Platform interface with several open windows:

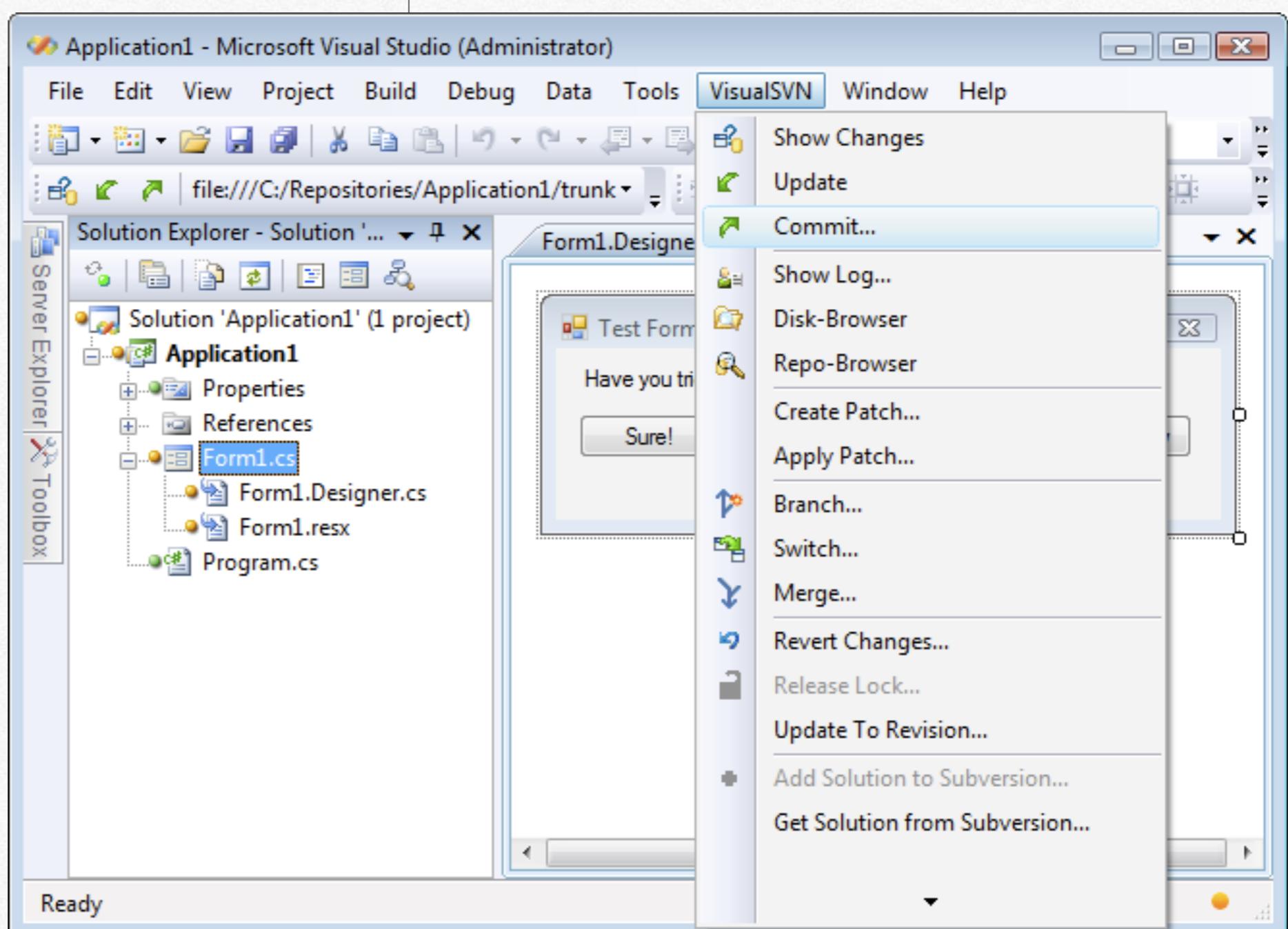
- PHP - Compare add.php <workspace> and versions - Eclipse Platform**: A comparison window showing the difference between the workspace file (`add.php`) and the repository file (`add.php`). The code is displayed in two panes with line numbers. Differences are highlighted in red.
- History**: A table showing the history of changes for the file `/trunk/add.php`. It lists revisions, dates, authors, comments, and bug IDs. Some entries are:

R...	Date	Author	Comment	Bug-ID
*259	28.5.08 17:08	xprskave	phpdoc update	
258	28.5.08 17:03	xprskave	phpdoc update	
254	28.5.08 16:30	xprskave	phpdoc update	
253	28.5.08 16:16	xprskave	phpdoc update files/dir cleanup	
252	28.5.08 16:10	xprskave	phpdoc update	
215	18.4.08 10:15	xprskave	Fix: editace rss zobrazovani ve skupinach	
109	3.12.07 12:54	abbris	Editace uživatelů (kompletní)	
97	1.12.07 18:54	abbris	Pridavání hodnot do číselníku	
- Affected paths**: A table showing affected paths for the commit "phpdoc update". It lists:

A..	Affected paths	Description
M	/trunk/add.php	phpdoc update
M	/trunk/delete.php	
M	/trunk/edit.php	
M	/trunk/inc/Calend...	
M	/trunk/inc/config.i...	
M	/trunk/inc/mysql_c...	

Microsoft Visual Studio

Pro práci se Subversion budete potřebovat AnkhSVN 2.0 nebo VisualSVN.



NetBeans, Komodo IDE, SciTE a další

SVN řádkový klient se používá v mnoha editorech i v IDE. Jako ukázku tu mám obrázek z NetBeans, které poskytuje celkem pěkný komfort a využívají jen řádkového klienta což přinese výhodu např. při vydání nové verze Subversion, stačí aktualizovat SVN klienta a nemusíte shánět podporu pro svoje IDE.

Komodo IDE patří ke komerčním editorům, který podporuje mnoho jazyků (PHP, Perl, Python, Ruby, Tcl, Javascript, Actionscript) a ze SCM podporuje kromě Subversion, Perforce a CVS.

JetBrains (RubyMine, PHPStorm, IntelliJ IDEA) mají velmi kvalitní IDE

pro Java, PHP, Ruby a další, kde je vestavěná podpora pro SVN, Git a další VCS.

The screenshot shows the NetBeans IDE interface with the title bar "Akce - NetBeans IDE Early Access for PHP". The left sidebar displays a project structure with files like 'styl', 'templates', 'tests', 'upload', 'CHANGELOG [Modified]', 'add.php [Modified]', etc. The main area shows a diff view for 'add.php' comparing 'Original' and 'Working Copy' versions. The code compares two versions of the same PHP script, highlighting differences in line numbers 31-32, 34-35, 36-37, 38-39, 40-41, 42-43, 44-45, 46-47, 48-49, 50-51, 52-53, 54-55, and 56-57. The 'Working Copy' version contains additional code, such as the addition of 'FILTER_SANITIZE' filters and a logger info message. The right side of the interface includes a 'Run-time watches' panel.

Distribuované systémy

Verzovací systému s architekturou klient server nejsou jedinou cestou.



Od Subversion k Mercurialu

Mercurial (hg) je další distribuovaný systém, který je napsaný v pythonu a podporuje ho například google ve svém code.google.com a také existuje hosting bitbucket.org, který umožňuje zdarma hosting open source projektů jako code.google.com nebo github.com pro git. Služby github.com a bitbucket.org mají také své placené varianty pro komerční využití.

Keywords v hg normálně nejsou, řeší se to přes standardní doplněk, který je potřeba zaplnout a nastavit v konfiguračním souboru .hgrc.

Export - obdoba svn export je v hg příkaz archive, který ale umožňuje také vytvořit ze souborů přímo archív (zip, tar, tgz, tbz2).

Log v xml formátu kompatibilním se svn log --xml. Pro logování jsem vytvořil script, který umí to co standardní parametr xml v SVN. Parametrem scriptu je výstupní soubor.

```
echo '<?xml version="1.0"?>\n<log>\n' >$1.xml
hg log --template '<logentry revision="{rev}">
<author>{author|obfuscate}</author>
<date>{date|isodate}</date>
<msg>{desc|escape}\n</msg>
<paths><path>{files}</path></paths>
</logentry>\n' $1/ >>$1.xml
echo '</log>\n' >>$1.xml
```

Script pro mercurial s template ve stejném formátu jako svn log --xml

DVCS a SVN

Některé distribuované systémy (git, bazaar a jistě další) mají utility, které umožňují, že lokálně pracujete jako s distribuovaným systémem, ale push provádít do SVN repozitory.

Ukázka postupu v gitu (git-svn)

```
git svn clone file:///srv/svn/repos/test
```

...

nějaké změny

...

```
git commit -a
```

```
git svn dcommit
```

Ukázka postupu v bazaar (bzr)

```
bzr checkout file:///srv/svn/repos/test
```

```
bzr unbind
```

...

nějaké změny

...

```
bzr commit
```

```
bzr bind
```

```
bzr push file:///home/svn/repos/tes
```

Tento postup lze zkrátit pomocí bzr commit --local potom nemusíme použít bind a unbind.

Další nástroje



Nástroje, které vám
pomohou s pracovat se
Subversion.

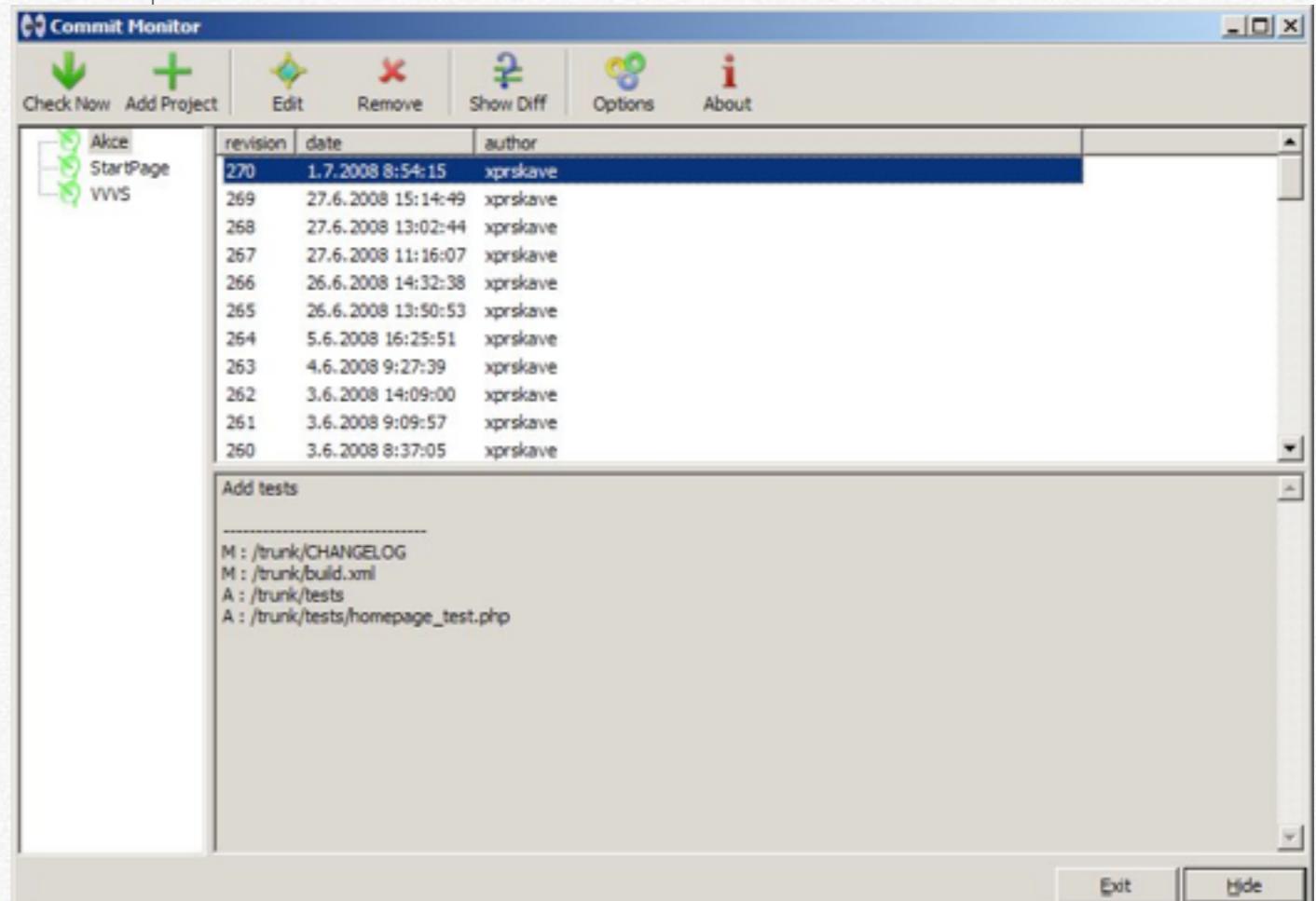
Commit Monitor

Program Commit Monitor pro Windows umožňuje sledovat změny v repozitóriu, které si do něj zadáte a upozorní vás pokud někdo provede commit. Autorem je Stefan Künig (<http://tools.tortoisessvn.net/>). Podobný program je ještě například SVN Notifier.

Bohužel neznám podobnou aplikaci pro Linux, obvykle se to nahrazuje mailovou konferencí kam chodí automaticky všechny commity pro projekt.

Existují webové nástroje, které podobné informace poskytují o projektech. Google Code a Github například poskytuje RSS o změnách. Podobně třeba Trac nebo jiné systémy, které mají podporu pro SVN.

Případně si to můžete udělat sami, jako například já v PHP.



StatSVN

StatSVN je program napsaný v Javě a slouží k analýze logů.

Stánout si ho můžete na

<http://sourceforge.net/projects/statsvn/>.

Použití je potom jednoduché jako první parametr dáte log v xml formátu a jako druhý cestu k aktuální pracovní kopii a potom výstupní adresář pro generované html soubory.

```
java -jar statsvn.jar  
c:\rep\test\test_changelog.xml \\\  
c:\rep\test\ -output-dir c:\tmp\test
```

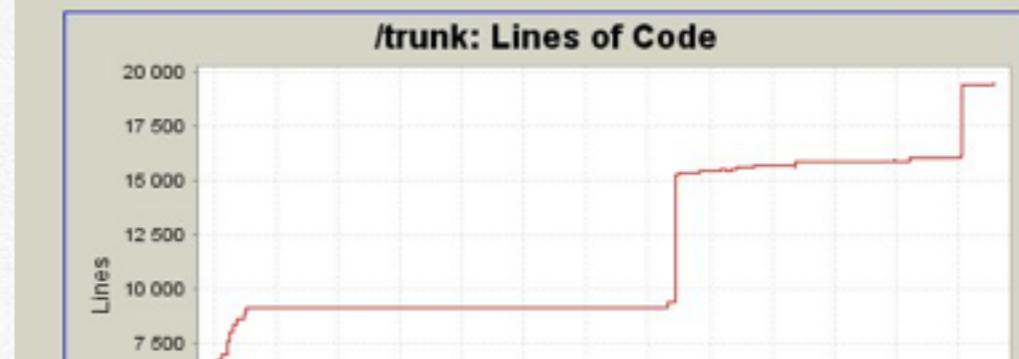
Výstup potom vypadá jako na obrázku. Tento program je zdarma, ale umí většinu statických věcí podobně jako profesionální programy pro práci s repozitory jako je FishEYE i když není zdaleka tak v líbivém kabátě.

Development Statistics for /trunk

Generated: 2008-06-18 15:00
Report Period: 2007-11-29 to 2008-06-05
Total Files: 105
Total Lines of Code: 19436
Developers: 2

- [Developers](#)
- [Commit Logs](#)
- [Lines of Code](#)
- [File Statistics](#)
- [Directory Sizes](#)
- [Repo Heatmap](#)
- [LOC and Churn](#)

Lines of Code



svn2cl

Generování changelogu pomocí svn2cl

Subversion utilita svn2cl je běžně dostupná v balíku subversion-tools v Debianu/Ubuntu.

Například pro generování changelogu pro tuto knihu se dělal takto:

```
/usr/bin/svn2cl --strip-prefix=trunk \
    --break-before-msg=2 \
    --group-by-day \
    --separate-daylogs \
    -i \
    --authors=authors.xml \
    file:///localhost/svn/svn-kniha/trunk/ \
    -o changelog.html \
```

--html

Soubor authors.xml obsahuje toto:

```
<?xml version="1.0" encoding="utf-8"?>

<authors>
    <author uid="abtris"> Ladislav Prskavec
    &lt;ladislav@prskavec.net&gt;;
    </author>
</authors>
```

Literatura



Všechno ani v této knize
nenajdete, proto se hodí
hledat i jinde.

SVN klienti

[TSVN] TortoiseSVN. URL: <http://tortoisessvn.tigris.org/>.

[SUBCLIPSE] Subclipse. URL: <http://subclipse.tigris.org/>.

[SUBVERSIVE] Subversive. URL:
<http://www.eclipse.org/subversive/>.

[ANKH SVN] AnkhSVN. URL: <http://ankhsvn.open.collab.net/>.

[RSVN] RapidSVN. URL: <http://rapidsvn.tigris.org/>.

[VISUALSVN] VisualSVN : Subversion for Visual Studio. URL:
<http://www.visualsvn.com/visualsvn/>.

SVN servery

[VSVN] VisualSVN server. URL:
<http://www.visualsvn.com/server/>.

Programovací prostředí - IDE

[ECLIPSE] Eclipse. URL: <http://www.eclipse.org/>.

[ZEND] Zend Studio for Eclipse. URL:
<http://www zend com/en/products/studio/>.

[NB] NetBeans. URL: <http://www.netbeans.org/>.

[KOMODO] Komodo IDE. URL:
http://www.activestate.com/Products/komodo_ide/.

[VS] MS Visual Studio. URL:

<http://msdn.microsoft.com/cs-cz/vstudio/>.

Dokumentace

[SVNBOOK] SVN book. URL: <http://svnbook.red-bean.com/>.

Berkley DB

Oracle Berkeley DB

Related Glossary Terms

Drag related terms here

Index

Find Term

Related Glossary Terms

Drag related terms here

Index

Find Term

Kapitola 1 - Jak funguje Subversion

CVS

Concurrent Version System

Related Glossary Terms

Drag related terms here

Index

Find Term

Kapitola 1 - Co je Subversion?

DAV

Distributed Authoring and Versioning

Related Glossary Terms

Drag related terms here

Index

Find Term

Kapitola 1 - Jak funguje Subversion

FSFS

An FSFS repository is smaller than a BDB repository.

Related Glossary Terms

Drag related terms here

[Index](#)

[Find Term](#)

[Kapitola 1 - Jak funguje Subversion](#)

JIRA

Software pro správu chyb od firmy Atlassian

Related Glossary Terms

Drag related terms here

Index

Find Term

Kapitola 4 - Vlastnosti

PHP

Programovací jazyk, PHP: Hypertext Preprocessor (původně "Personal Home Page")

Related Glossary Terms

Drag related terms here

Index

Find Term

Kapitola 4 - Vlastnosti

SCM

Source Control Management

Related Glossary Terms

Drag related terms here

Index

Find Term

Kapitola 1 - Co je Subversion?

SVN

Subversion

Related Glossary Terms

Drag related terms here

Index

[Find Term](#)

[Kapitola 1 - Jak funguje Subversion](#)

TSVN

TortoiseSVN - nejlepší klient pro Windows

Related Glossary Terms

Drag related terms here

Index

Find Term

Kapitola 3 - Subversion řádkový klient