Version Control Systems
(Systémy na správu verzí)

Software Configuration Management (SCM) Systems

Jan Outrata

2007

O čem to je - základy spravování verzí

- zachovat historii vývoje (procházet, návrat k předchozím verzím, kdy, kdo, co změnil)
- spolupráce více vývojářů, přístup k poslední vývojové verzi
- vývoj více paralelních větví
- síťová práce - zasílání změn (patchů) po síti
- nejen programátorský kód! - dokumentace, data (binární), ...

repozitář (repository)

- skladiště všech verzí + metadata
- z něj pracovní adresář
- ukládání do repozitáře - DB, pro každý soubor soubor historie, balíky se změnami (changeset)
- lokální nebo na serveru

- přístup k repozitáři - lokální, server, jednoduché web. rozhraní, kombinace (SSH, WebDAV)

revize (commit, changeset)

- stav projektu v určitém čase (mezi změnami) nebo seznam změn vedoucích k (mezi)stavu (changeset)
- identifikace a popis - numerická, symbolická (oprava-chyby), hash
- buď pro každý soubor zvlášť (CVS, bk) nebo pro celý projekt (SVN)
- řešení konfliktů při vytváření nové revize - změny v repozitáři i v prac. kopii, které nelze automaticky sloučit, sloučit ručně nebo vybrat jednu

větev (branch)

- práce s více větvemi zároveň (stabilní, vývojová)
- problémy se slučováním (merge) větví (aktuální výzkum)
- cherrypicking - výběr jen některých změn z větví

Jak se jednotlivé systémy liší - hlavní přístupy

vývojový model:

centralizovaný

- jeden hlavní repozitář, pracovní kopie projektu (ne repozitáře!)
- model klient/server
- propagace změn z pracovní kopie do hlavního repozitáře = push
- problém havárie repozitáře
- numerická identifikace revizí, př. CVS, SVN

decentralizovaný (distribuovaný)

- klonování hlavního repozitáře do lokálních repozitářů (+ pracovní kopie z nich), nezávislý paralelní lokální vývoj (větší svoboda vývojářů, komerčně ve firmách naopak nevýhoda)
- propagace změn z lokálních repozitářů do hlavního = push

- stažení změn z lokálních repozitářů do hlavního = pull
- hlavní repozitář je jen synchronizační bod vývojářů anebo bez něj jen jednoduchá výměna změn (patchů)
- havárie hlavního repozitáře neohrožuje vývoj
- systém komplikovanější (slučování změn)
- identifikace revizí unikátní napříč lokálními a hlavním repozitářem (hash, pro odkazování stačí prvních pár znaků)

ukládání revizí:

- stavy celého projektu - jednoduché, hodně místa, obtížnější získat jednotlivé změny, př. CVS, git
- změny mezi stavy - jednoduchá výměna jednotlivých změn, méně místa, složitější (vytváření stavů), př. Darcs
- kombinace

Prakticky - operace nad repozitářem, práce s ním

- inicializace/vytvoření repozitáře

```
cvs -d/repo init
snvadmin create /repo --fs-type fsfs
svk admin create /repo
darcs initialize
git-init-db, cg-init
bzr init/init-repo /repo
hg init /repo
```

- vložení nového adresáře/projektu (import) - popis

```
cvs import adr.
svn/svk import adr. file:///repo
darcs add -r adr
git-add, cg-add
bzr add adr
hg add adr
```

- export z repozitáře

```
cvs export
svn export
darcs dist
git-tar-tree, cg-export
```

```
bzr export
hg export/archive/bundle
```

“pracovní cyklus”:

- vytažení prac. kopie (checkout) - adresář repozitáře (CVS, .svn, _darcs, .bzr, .hg (jeden), ...)

```
cvs co -D"datum" -r"revize"
svn co -r revize/{datum}
git checkout, cg-fetch
bzr co -r revize
hg co revize
```

- aktualizace prac. kopie (update) - konflikty

```
cvs up -D
svn up -r
darcs pull
git-pull, cg-update
bzr up
hg up revize
```

- přidání/smazání/kopie/přesun souboru/adresáře

```
cvs add/delete
svn add/delete/copy/move
darcs add/remove/mv
git-add/git-mv, cg-add/cg-rm
bzr add/mkdir/mv/remove
hg add/remove/rename
```

- kontrola změn (status) před commitem

```
cvs status
svn status
darcs whatsnew
git-status, cg-status
bzr st -r revize
hg st
```

- změny mezi revizemi/prac. kopií (diff)

```
cvs diff -D -D
svn diff -r revize:revize
darcs diff --from-patch --to-patch
git-diff id..id, cg-diff id..id
bzr di -r revize..revize
hg diff -r revize -r revize
```

- anotace - u každého řádku ve které revizi a kým naposledy změněn

```
cvs annotate
svn annotate
darcs annotate
git-whatchanged
bzr ann -r revize
hg annotate -r revize
```

- undo (revert) - zrušit změny v pracovní kopii

```
cvs unedit
svn revert
darcs revert
git-revert, cg-restore
bzr revert -r revize
hg revert -r revize
```

- zahrnutí změn (commit) - popis, konflikty

```
cvs ci
svn ci
darcs record
git-commit, cg-commit
bzr ci -m popis
hg ci -m popis
```

další:

- seznam změn (log) - od revize k revizi, data k datu, ...

```
cvs log
svn log
darcs changes
git-log, cg-log
bzr log
hg log
```

- označení revize (tag) - použití místo identifikace revize, př. stable

```
cvs tag
svn copy
darcs tag
git-mktag, cg-tag
bzr nick
hg tag
```

- vytvoření nové větve

```
cvs tag -b VETEV
svn copy
darcs get url
git-branch
bzr branch
hg clone
```

- slučování větví (merge) - změny mezi větvemi do prac. kopie, konflikty, cherrypicking

```
cvs up -j VETEV -j VETEV
svn merge file:///repo@revize file:///repo@revize
darcs pull
git-merge, cg-merge
bzr merge
hg merge
```

hooky - před/po akci (typicky commit) vlastní akce, např. automatická kompilace

distribuované:

- klonování repozitáře - vytvoření lokálního z hlavního

```
svk mirror
dargs get
git-clone, cg-clone
bzr branch
hg clone
```

- push - zaslání (“natlačení”) změn z repozitáře do repozitáře

```
svk push file:///repo@revize
darcs push url
git-push /repo, cg-push
bzr push url
hg push url
```

- pull - stáhnutí změn z repozitáře do repozitáře, cherrypicking, př. Linus

```
svk pull file:///repo@revize
darcs pull url
git pull /repo, cg-pull
bzr pull url
hg pull url
```

síťově:

- server s repozitářem, různé metody přístupu

  cvs

  - cvspserver (komunikace: nešifrované heslo) nebo cvsd: `-d:pserver:login@server:/repo` (nebo `$CVSROOT`) `login`
  - ssh: pserver → ext a `$CVS_RSH=ssh`

  svn

  - snvserve: `svn://server/repo`
  - Apache/WebDAV: `http://server/repo`
  - ssh: `svn+ssh://login@server/repo` (`$SVN_SSH`)

  darcs

  - `http://server/repo` (pull)
  - ssh (push)

git
  - `rsync://`, `http://`, `ssh://`
  - git-daemon: `git://`

bzr
  - `http://`, `(s)ftp://`
  - smart server (bzr serve): `bzr://`, `bzr+ssh://`

hg
  - `http://`, `ssh://`
  - HTTP server (hg serve): `http://`

administrace: uživatelé, práva, hooky, ignorované soubory, . . .

```
cvs admin
snvadmin
svk admin
editace souboru v _darcs/prefs/
git-repo-config
editace souborů v .bazaar/
editace souborů .hgrc, .hgignore
```

nápověda:

```
cvs -H příkaz
snv help příkaz
svk help příkaz
darcs help příkaz
git-help příkaz, cg-help
bzr help příkaz
hg help příkaz
```

Příklady - OSS/FS (až na výjimky)

CVS (cvs)

- v době vzniku (1989) inovativní, dnes historické, použití ze zvyku
- přísně centralizované - distribuovaná varianta DCVS
- hodně problémů: neatomické commity (změny ve více souborech každá zvlášť, jak získat celou změnu?, přerušení → nekonzistentní stav), špatná podpora větvení, minimální podpora pro (opakované) slučování větví (nezachovává se historie obou, jen nová revize v jedné), problém mazání adresářů, přejmenování souborů/adresářů → kopie, problém u velkých a binárních souborů (zamykání), neefektivní síťová komunikace
- používá např. GCC (náročné!)
- "CVS není odpověď, CVS je otázka. Ne je odpověď. Theodore Ts'o"
- WebCVS, ViewCVS, GUI (gcvs/WinCVS/CVSGui, Cervisia)
- online book, tutoriály http://www.cvshome.org

Subversion (svn)

- reimplementace CVS ("CVS 2.0"), řešení problémů CVS - slučování větví pořád problém (nepamatuje si, které záplaty již byly aplikovány na dané větvi), neefektivní síťová komunikace
- centralizované - distribuovaná varianta SVK nebo skripty SVN:Mirror, svn-push
- ukládá do DB - existuje i souborový backend fsfs
- BSD-stará licence (Apache, ne GPL kompat.!)
- používá např. KDE, GCC uvažuje
- WebSVN, ViewSVN, GUI (RapidSVN)
- online book, tutoriály http://subversion.tigris.org/

GNU Arch (tla)

- první distribuovaný, může fungovat i centralizovaně (přes práva)
- flexibilní a mocný, dobrý v cherrypicking a větvení vůbec ("star merge")

- dnes nespravovaný, komplikované UI (ale „chytré", repozitář stačí nasdílený adresář), pomalý
- nefunguje dobře na Windows (dlouhá divná jména souborů - +={}, symb. linky, práva, nový řádek)
- ViewARCH http://gnuarch.org/

SVK (svk)

- distribuovaná varianta SVN, http://svk.bestpractical.com/
- dobré slučování větví a cherrypicking ("star merge" z Archu), rychlé
- nezralé? (nedodělaná dokumentace - online book) http://svk.elixus.org/

Bazaar/Bazaar-NG (bzr)

- distribuovaný, vychází z GNU Archu/Darcsu, jednodušší UI, portabilní (NG v Pythonu) http://www.bazaar-vcs.org/

Darcs (darcs)

- distribuovaný, zajímavý (napsaný v Haskellu)

- dobrý elegantní návrh (“fyzikální”, patch algebra podobná kvantové mechanice)
- ukládá jen změny (patche), pracovní adresář = repozitář
- problém u velkých projektů
- CGI pro webový přístup
- online manuál http://darcs.net

Monotone (mtn)

- distribuovaný, průkopník nového přístupu (“Monotone design school”)
- objektový model - SHA-1 hashe objektů (soubor, strom, commit)
- flexibilní (skriptování), kryptografie (podepisování commitů, ověřování při operacích)
- pomalý!
- žádné web. rozhraní či GUI http://monotone.ca/

CodeVille (cdv)

- distribuovaný, objektový model Monotone
- pořádné slučování větví (Precise CDV Merge)
- spíše výzkumné použití http://www.codeville.org/

Mercurial (hg)

- distribuovaný, objektový model Monotone, velmi podobný Gitu
- rychlý (i na velmi velké projekty)
- ukládá změny jako Darcs http://www.selenic.com/mercurial/

Bitkeeper, BK/Pro (bk)

- první rozšířený distribuovaný
- atomické commity pomocí changesetů (soubor změn jednotlivých suborů)
- dobrý ve slučování větví (repozitář pro každou větev, weaves)
- hodně příkazů
- ideál?, proprietární! (BitMover, Inc., byla volně dostupná verze, ale jen pro nedělající na VCS), dříve použití na linuxové jádro

- vlastní web. rozhraní i GUI
- help na homepage http://www.bitkeeper.com/

GIT/Cogito (git-*, cg-*)

- distribuovaný, objektový model Monotone, původně pro linuxové jádro vyvinutý vývojáři (po zrušení volného bk)
- ukládá celé soubory (komprimované)
- více specializovaných příkazů (UNIX like), git low level → nadstavba Cogito (jednodušší CVS-like UI)
- rychlý! (cache adresářů)
- původně pro UNIX like systémy
- GITweb, qgit, gitk
- Git Howto http://git.or.cz/

další - Aegis, CVSNT, Vesta (nejen SCM, i „make"), Superversion, ... RSC, SCCS :-)

proprietární - Perforce, Rational ClearCase, ...

srovnání - http://better-scm.berlios.de/

Na čem se dělá - řešení problémů

- hodně nových přístupů, nových systémů, duplikace práce, štěpení sil $\rightarrow$ snahy o sjednocování, spolupráci, společný výzkum (http://revctrl.org)
- problém sjednocování (merge) větví, v současnosti "three-way merge" (báze B - poslední společný předek, sloučit A a C = kombinace změn $B \rightarrow A$ a $B \rightarrow C$) - může být hodně konfliktů nebo naopak špatný výsledek bez konfliktů ("criss-cross merge")
- weaves (interleaved delta) - normálně historie jako změny soubor vůči souboru, weaves má historii jako změny na úrovni řádků (původně ve starém SCCS - 30 let!, složitější implementace (s objektovým modelem) $\rightarrow$ Precise CDV Merge
- cherrypicking - problém u objektového modelu Monotone, weaves dobré
- homework problem - opravy chyb dostat do původní chybové revize (opravy nedůležité, důležitý výsledek) $\rightarrow$ dnes stabilní a vývojová větev
- UI - složité příkazy $\rightarrow$ GUI