

Systemy kontroli wersji **Bazaar** i **Mercurial**

Michał Bugno Krzysztof Goj

18 kwietnia 2008, Akademia Górniczo-Hutnicza

VCS

VCS czyli Version Control System pozwala na

- przetrzymywanie historii zmian w plikach
- powrót do ewentualnych poprzednich wersji pliku
- równoczesną pracę kilku osób przy projekcie
- późniejsze rozwiązywanie ewentualnych konfliktów (jeśli ten sam plik edytowała więcej niż jedna osoba)
- tworzenie gałęzi, czyli odłączanie się w pewnym momencie od głównego projektu

Narzędzia

VCS wyposażony jest w zestaw przydatnych narzędzi, które ułatwiają pracę:

- `log` – wyświetla listę rewizji wraz z ewentualnymi komentarzami do rewizji
- `diff` – pozwala na wyświetlenie różnic między ostatnią rewizją a naszymi aktualnymi zmianami
- `revert` – cofa aktualne zmiany
- `blame` – najciekawsze :) pozwala na podgląd, kto zmienił którą linię w pliku
- `praise` – j.w. ale w tym pozytywnym kontekście :)

CVS – Concurrent Versions System

- stworzony w latach .80
- brak atomowych commitów!
- brak suportu dla przenoszenia plików/katalogów
- obsługiwał już branching

SVN - Subversion

- stosunkowo młody – 2000 rok
- tworzony na CVS potem przeszedł na *siebie*
- cel – kompatybilność z CVS
- atomowość
- przesyłane różnice a nie całe pliki
- properties: executable, ignore, mime-type
- popularny

D czyli Distributed

- bardzo młode projekty – Bazaar, Mercurial, Git 2005 r.
- inna ideologia – nie ma centralnego repozytorium
- wprowadzają dużo usprawnień względem SVN
- *troszkę* bardziej skomplikowane

Co to jest rozproszony system kontroli wersji?

- brak *centralnego* serwera
- popularne akcje jak *commit*, *log* są **lokalne**

Zalety

- niezależność od dostępności serwera (brak internetu != FAIL)
- bezpieczeństwo - każda gałąź to osobny projekt *wraz* z logami, historią, itp itd.
- łatwiejsze dodawanie *dużej* funkcjonalności do projektu wymagającej dłuższej pracy bez aktualizacji
- lepsza współpraca ludzi *poza projektem*, którzy chcieliby dodać coś do projektu

Wady

- dojrzałość – SVN jest już dobrze *otestowany* przez użytkowników
- przetrzymywanie całej historii zmian – zajętość miejsca
- stopień skomplikowania – *trochę* więcej komend, inna idea

Instalacja

Instalacja jest na wszystkich systemach prosta, szybka i bezbolesna.

- wszędzie: źródła
- GNU/Linux: paczki
- Windows: instalator
- MacOS X: MacPorts, paczki (Mercurial)

init – rozpoczęcie wersjonowania katalogu

dwa sposoby

Bazaar

```
pykonik bzr init repo  
pykonik ls  
repo  
pykonik cd repo  
repo ls -A  
.bzr
```

Mercurial

```
repo hg init  
repo ls -A  
.hg
```

można zamienić `bzr` ↔ `hg`

add – dodanie plików

```
repo mkdir test other  
repo touch a1 a2 test/a{3,4} other/a5
```

Bazaar

```
repo bzz add  
added a1  
added a2  
added other  
added test  
added other/a5  
added test/a3  
added test/a4
```

Mercurial

```
repo hg add  
adding a1  
adding a2  
adding other  
adding test  
adding other/a5  
adding test/a3  
adding test/a4
```

add – dodanie plików

```
repo mkdir -p dir/subdir  
repo touch file dir/subdir/file
```

Bazaar

```
repo bzr add file dir  
added file  
added dir  
added dir/subdir  
added dir/subdir/file
```

Mercurial

```
repo hg add file dir  
adding dir/subdir/file
```

ignore – ignoruj pliki

Bazaar

```
repo bzz ignore "*.o"  
repo bzz ignore "RE:d.pa"  
repo cat .bzrignore  
    *.o  
    RE:d.pa  
~/bazaar/ignore
```

Mercurial

```
repo vim .hgignore  
repo cat .hgignore  
  
syntax: glob  
*.o  
syntax: regexp  
d.pa
```

per-user, per-user*project (hgrc)

```
[ui]  
ignore = ścieżka_do_pliku
```

status – stan kopii roboczej

```
repo echo "new content" >> a1  
repo touch a3
```

Bazaar

```
repo bzip rm a2  
deleted a2  
repo bzip status  
removed:  
  a2  
modified:  
  a1  
unknown:  
  a3
```

Mercurial

```
repo hg rm -after  
removing a2  
repo hg status  
M a1  
R a2  
? a3
```

diff

Bazaar

repo **hg diff**

```
diff -r 19ade018a43e
a1
--- a/a1    Thu Apr 10
14:34:08 2008 +0200
+++ b/a1    Thu Apr 10
14:34:30 2008 +0200
@@ -0,0 +1,1 @@
+new content
```

Mercurial

repo **bzr diff**

```
=== modified file 'a1'
--- a1 2008-04-10
12:33:42 +0000
+++ a1 2008-04-10
12:34:30 +0000
@@ -0,0 +1,1 @@
+new content

=== removed file 'a2'
=== removed file
```

revert – wycofaj zmiany

Bazaar

```
repo echo stuff >> a1
repo bzz rm a2 file
repo bzz revert
      +N a2
      +N file
      M a1
repo  ls a1*
      a1 a1..1~
```

Mercurial

```
repo echo stuff >> a1
repo hg rm a2 file
repo hg revert
      reverting a1
      undeleting a2
      reverting file
repo  ls a1*
      a1 a1.orig
```

commit – stworzenie rewizji

Bazaar

```
repo bzr ci -m "Init"
```

```
Committing to:  
pykonik/repo/  
added a1  
...  
added test/a4  
Committed revision 1.
```

```
repo vim a1
```

```
repo bzr ci
```

Otwiera sesję edytora

```
repo bzr ci -m "Changes in a1"
```

Mercurial

```
repo hg ci -m "Init"
```

```
repo vim a1
```

```
repo hg ci
```

Otwiera sesję edytora

```
repo hg ci -m "Changes in a1"
```

bzr log – wyświetlenie historii zmian

repo **bzr log**

```
-----  
revno: 2  
committer: Michal Bugno  
<michal.bugno@gmail.com>  
branch nick: repo  
timestamp: Sun 2008-04-06  
22:16:40 +0200  
message:  
  Changes in a1  
-----
```

```
revno: 1  
committer: Michal Bugno  
<michal.bugno@gmail.com>  
branch nick: repo  
timestamp: Sun 2008-04-06  
21:54:17 +0200  
message:
```

hg log – wyświetlenie historii zmian

opcja -v włącza więcej szczegółów

repo **hg log -v**

```
changeset: 8:a6ba8f70ce7eba6a73e5715a439b06a9584106ee
user:      Krzysiek Goj <bin-krzysiek@poczta.gazeta.pl>
date:      Thu Apr 10 02:04:19 2008 +0200
files:     pykonik_dvcs.tex
description:
Bardzo ciekawe zmiany.
- nowe slajdy
- więcej przykładów
```

branch/clone – stworzenie kopii repozytorium

Bazaar

```
pykonik bzr branch repo other  
Branched 2 revision(s).  
pykonik cd other  
other bzr log  
other vim a1  
other bzr mv test dir  
other bzr st  
renamed:  
test ⇒ dir  
modified:  
a1  
other bzr ci -m "..."
```

Mercurial

```
pykonik hg clone repo other  
3 files updated, 0 files  
merged, 0 files removed,  
0 files unresolved  
pykonik cd other  
other hg log  
other vim a1  
other hg mv dir test  
other hg st  
M a1  
A test/a3  
R dir/a3  
other hg ci -m "..."
```

serve

Bazaar

pykonik **bzr serve –port 2345**

listening on port: 2345

Mercurial

pykonik **hg serve -p 5432**

pykonik **hg serve -v**

listening at
<http://abulafia:8000/>

Konflikty

- `bzr` dobrze radzi sobie z rozwiązywaniem (założenia DVCS!)
- `foo.BASE`, `foo.THIS`, `foo.OTHER`
- znaczniki w plikach – miejsca konfliktu
- `bzr resolved` – oznaczenie konfliktu jako rozwiązany

Praca zcentralizowana – czyli bzd jako svn

Bazaar może działać jak svn – istnienie jednego, centralnego repozytorium, które jest uaktualniane przez użytkowników, każdy *commit* poprzedzany jest *update*-em.

- branch i checkout – różnice
- konwersja branch \Rightarrow checkout – bzd bind
- checkout \Rightarrow ... \Rightarrow update \Rightarrow ... \Rightarrow commit

Gdy brak centralnego serwera...

Kiedy jesteśmy odcięci od internetu/sieci nadal jest możliwość wersjonowania plików w *checkout*-cie.

- `bzr ci --local` – zapisz zmiany lokalnie
- `bzr unbind` – checkout \Rightarrow branch
- `bzr bind ...` – branch \Rightarrow checkout

bzr merge – połączenie branch'y

```
repo bzr ci -m "..."
repo bzr branch ../other

M a1
R test/ ⇒ dir/
All changes applied
successfully.
repo bzr st

renamed:
test ⇒ dir
modified:
a1
pending merges:
John Doe 2008-04-06
renamed and modified

repo bzr ci -m "merged"
```

Praca zdecentralizowana z bzd

- main branch – umowny branch główny
- własny branch do zmian – w nim edytujemy/naprawiamy/dodajemy
- zapisywanie zmian: różne metody (gatekeeper lub zwykły commit)

Ciekawostki

- `uncommit`
- `/.bazaar/bazaar.conf` – email, editor
- aliasy do różnych komend, np. `bzr commit --strict`,
`status --short`
- `commit --fixes lp:5`
- pluginy – integracja z innymi VCSami
 - hg – readonly
 - git – readonly
 - hgimport – importowanie mercuriala
 - svn – współpraca z bindingami svn

konfiguracja

- ~/.hgrc
- .hg/hgrc

[ui]

username = Krzysiek Goj <bin-krzysiek@poczta.gazeta.pl>

ignore = /home/goj/.hgignore-global

[extensions]

mq =

hbisect =

push i pull, czyli współpraca z innymi

```
repo hg push docelowe_repo  
repo hg pull źródłowe_repo  
repo hg update  
repo hg merge
```

hooki – jak reagować na wydarzenia i wymuszać niektóre rzeczy

pykonik **cat .hg/hgrc**

```
[hooks]
pretxncommit.missing_add_remove = ! hg status
| grep -q '[?!]'
```

- również moduły pythona
- incoming
- update
- ...i 10 innych

inne tematy

- cgi
- bisect
- konfiguracja
- branch
- rozszerzenia

Linki

Bazaar

- <http://bazaar-vcs.org>
- <http://launchpad.net>

Mercurial

- <http://www.selenic.com/mercurial/wiki/>
- <http://hgbook.red-bean.com/>
- <http://www.ivy.fr/mercurial/ref/v1.0/Mercurial-Usage-v1.0.pdf>
- <http://sharesource.org/>