Systemy zarządzania wersjami

Norbert Potocki

24 października 2007

Po co? Dla kogo? Dlaczego?

- aby ułatwić kontrolę nad projektem
- aby panować nad wersjami
- aby móc śledzić zmiany
- dla każdego projektu, przy którym pracuje więcej niż jedna osoba
- dla ludzi ekstremalnych: zawsze :)
- bo jest to wygodne
- bo jest to praktyczne
- bo tak robią profesjonaliści - czyli MY :)

Przykładowe systemy zarządzania wersjami

- CVS
- Subversion
- Git - używany przez kernel.org
- IBM Rational ClearCase
- Microsoft Visual SourceSafe (dla małych projektów)
- Visual Studio Team System (dla dużych projektów)
- Mercurial (napisany w Python'ie)
- Darcs (napisany w Haskell'u)
- SVK (napisany w Perl'u)

CVS

- zalety
  - projekt open-source
  - jeden ze starszych systemów (powstał w 1980 roku)
  - dobrze udokumentowany i przetestowany
  - system o dużej funkcjonalności
- wady
  - od dawna nie dodaje się do niego nowej funkcjonalności
  - nie wspiera atomowych operacji commit
  - nie pozwala zmieniać nazw plików poddanych zarządzaniu wersjami
  - słabe wsparcie dla Unicode
  - nie pozwala zarządzać dowiązaniami symbolicznymi (są one potencjalnie niebezpieczne)

Subversion (SVN)

- zalety
  - projekt open-source
  - naprawia braki CVS'a - powstał jako jego następca w 2000 roku (CollabNet Inc.)
  - jest aktywnie rozwijany
  - bardzo efektywnie przechowuje pliki binarne
  - pozwala na "niemal" dowolne zmiany statusu plików, nie powodując utraty historii wersji
- wady
  - problemy ze zmianą nazwy pliku - działa na zasadzie kopiuj-usuń
  - trudne administrowanie repozytorium

Wygodne nakładki graficzne

- eSvn - wielosystemowa, oparta o QT
- KDESvn - zaprojektowana specjalnie dla KDE
- Netbeans - zawiera moduł integracji z Subversion
- RapidSVN - wielosystemowa
- TortoiseSVN - prawdopodobnie najlepsza nakładka dla Windows, integruje się z explorerem

Interfejsy web'owe

- ViewVC — używane przez MPlayer'a
- WebSVN — wspierana przez Tigris
- sventon — aplikacja napisana w Java do przeglądania repozytoriów

Na zachętę - kto używa SVN?

- MPlayer
- GCC
- Apache
- KDE
- GNOME
- Google Code :)

Darmowe repozytoria SVN

- MIMUW students Server - potrzebne “wtyki” u wykładowców :)
- SourceForge (sourceforge.net)
- Berlios.de (www.berlios.de)
- CVSDude (cvsdude.com)
- Google (code.google.com/hosting)
- CodeSpaces (www.CodeSpaces.com)

Elementy instalacji Subversion

- svn — klient Subversion
- svnadmin — narzędzie do tworzenia i zarządzania repozytoriami
- svnserve — samodzielny serwer repozytoriów Subversion
- inne — dla zaawansowanych :)

Trochę teorii

- Subversion — system scentralizowany, którego centrum jest repozytorium
- repozytorium przechowuje dane poddane zarządzaniu wersjami oraz historię ich zmian
- repozytorium posiada typową hierarchę drzewiastą - jak większość systemów plików :)
- metodologia “copy-modify-merge”
- kopie lokalne o katalog “.svn”

Subversion - najważniejsze polecenia

- help — najważniejsze !!! - wyświetla pomoc
- import — wgrywa strukturę katalogów do repozytorium
- list — wyświetla strukturę repozytorium
- checkout — pobiera dane z serwera SVN i tworzy lokalną kopię roboczą danego repozytorium
- commit — wysyła zmiany wprowadzone w kopii roboczej do repozytorium
- update — aktualizuje kopię roboczą do wersji znajdującej się w repozytorium
- status — pokazuje zmiany w kopii roboczej w stosunku do ostatnio pobranej wersji z repozytorium
- add / delete / copy / move — podstawowe zarządzanie plikami
- revert — cofnij wprowadzone zmiany
- log — wyświetla historię zmian

polecenie: svn help

- składnia — svn help [KOMENDA]
- ważniejsze parametry — brak
- działanie — wyświetla listę możliwych komend; gdy podana jest komenda wypisuje pomoc na jej temat

polecenie: svn import

- składnia — svn import [ŚCIEŻKA] URL
- ważniejsze parametry
  - -m opis — dodaj opis do informacji o aktualizacji
  - - -username użytkownik — nazwa użytkownika
  - - -password hasło — hasło dla użytkownika
- działanie — wykonaj operację commit na pliku (bądź katalogu) nie podlegającym kontroli wersji. Zazwyczaj używa się go aby wgrać pierwszą wersję źródeł do repozytorium

polecenie: svn list

- składnia — svn list [CEL[@WERSJA]]
- ważniejsze parametry
  - -r wersja — komenda odnosi się do wersji “wersja”
  - - -xml — wyświetl w formacie XML
- działanie — wyświetl strukturę plików i katalogów repozytorium w wersji WERSJA (lub aktualnej, jeśli WERSJA nie została podana)

polecenie: svn checkout

- składnia — svn checkout URL[@WERSJA]... [ŚCIEŻKA]
- ważniejsze parametry
  - -r wersja — komenda odnosi się do wersji “wersja”
- działanie — pobierz repozytorium spod adresu URL w wersji WERSJA i utwórz lokalną kopię roboczą w ŚCIEŻKA

polecenie: svn commit

- składnia — svn commit [ŚCIEŻKA]
- ważniejsze parametry
  - -m opis — dodaj opis do informacji o aktualizacji (WYMAGANE!!)
  - -F plik — pobież opis z pliku “plik“
- działanie — wyślij zmiany dokonane w lokalnej kopii roboczej do repozytorium

polecenie: svn update

- składnia — svn update [ŚCIEŻKA]
- ważniejsze parametry
  - -r wersja — komenda odnosi się do wersji “wersja”
- działanie — pobierz zmiany z repozytorium do lokalnej kopii roboczej. W trakcie pobierania danych Subversion będzie wypisywał nazwy zmodyfikowanych plików wraz z kodami literowymi oznaczającymi wprowadzoną zmianę:
  - A — dodany
  - D — usunięty
  - U — zaktualizowany
  - C — spowodował kolizję
  - G — złączony (operacją merge)

polecenie: svn status

- składnia — svn status [ŚCIEŻKA]
- ważniejsze parametry — brak
- działanie — wyświetl informacje o zmianach wprowadzonych do lokalnej kopii w stosunku do wersji pobranej przy ostatniej synchronizacji z repozytorium. Wypisuje nazwy plików oraz literowy kod oznaczający wprowadzoną zmianę. Kodów tych nie przytaczamy, gdyż jest ich dużo. Można o nich poczytać po wykonaniu polecenia "svn help status".

polecenie: svn add / delete / copy / move

- składnia — svn [add OR delete OR copy OR move] ŚCIEŻKA1 [ŚCIEŻKA2]
- ważniejsze parametry
  - -m opis — dodaj opis do informacji o aktualizacji
- działanie — wykonaj operację [dodawania OR usuwania OR kopiowania OR przenoszenia] pliku w repozytorium. Zmiany zostają zatwierdzone po wykonaniu najbliższej operacji “commit”

polecenie: svn revert

- składnia — svn revert ŚCIEŻKA
- ważniejsze parametry — brak
- działanie — przywróć dany plik do stanu z poprzedniej synchronizacji z repozytorium

polecenie: svn log

- składnia — svn log [ŚCIEŻKA]
- ważniejsze parametry
  - - -limit liczba — ustaw limit wyświetlanych wiadomości na “liczba”
- działanie — wyświetl historię zmian dla repozytorium lub pliku ŚCIEŻKA

Rozwiązywanie kolizji

- jeżeli po wprowadzeniu zmian w lokalnej kopii pliku wykonamy polecenie “update” możemy otrzymać kolizję
- rozwiązywanie kolizji jest proste a zarazem trudne :)
- Subversion dodaje do pliku znaczniki zmian..
- .. i tworzy pliki tymczasowe zawierające Twoją wersję pliku; wersję z repozytorium oraz wersję z poprzedniego “update”
- po rozwiązaniu kolizji wykonujemy “svn resolved nazwa_pliku”

Trochę biologii - o gałęziach

- gałąź — linia rozwojowa istniejąca niezależnie od innych linii rozwojowych, posiadająca jednak wspólną z nimi historię (do pewnego momentu)
- Subversion nie posiada wewnętrznego pojęcia gałęzi — wszystko odbywa się poprzez kopiowanie poddrzewa plików

Bibliografia

- http://en.wikipedia.org/wiki/Concurrent_Versions_System
- http://en.wikipedia.org/wiki/Subversion_(software)
- http://en.wikipedia.org/wiki/List_of_revision_control_software
- http://en.wikipedia.org/wiki/Comparison_of_revision_control_software
- http://subversion.tigris.org
- http://svnbook.red-bean.com