Versiohallinta ja Subversion

26.9.2007

Maunu Tuomainen
mttuomai@jyu.fi

Versiohallinta yleisesti

- Ongelma: lähdekoodin ja muun materiaalin säilyttäminen siten, että:
  - se on “kaikkien” saatavilla
  - tuorein versio on aina käytettävissä
  - muutoshistoria on tallella, eli vanhempiinkin versioihin päästään käsiksi

Ad hoc -versiohallinta

- Hahmotelma ad hoc -versiohallinnasta:
  - tallennetaan tiedostot jaetulle levylle ja nimetään tiedostoston versiot juoksevalla numeroinnilla
  - entä jos kaksi henkilöä muokkaa samaa tiedostoa samanaikaisesti?

Samanaikaisten muutosten hallinta

- Voidaan joko kieltää
  - tiedosto lukitaan editoitaessa ja vain yksi käyttäjä muokkaa tiedostoa kerrallaan
- ...tai sallia
  - samanaikaiset muutokset yhdistetään ja vain tiedoston samaan osaan (tarkemmin: tekstitiedoston riviin) kohdistuvat muutokset aiheuttavat ongelmia

Miksi versiohallintaa?

- Tiedon tallessa pysyminen
- Versiohistorian säilyminen
  - uskaltaa tehdä isojakin muutoksia, kun aina voi palata vanhaan toimivaan
- Yhtäaikaisten muutosten hallinta

Subversion

- Subversion on eräs suosituimpia versiohallintajärjestelmiä
- Paljon käytetyn CVS:n “paranneltu versio”
- Kotisivu:

  http://subversion.tigris.org/
- Kaikki oleellinen ohjeistus löytyy kirjasta Version Control with Subversion:

  http://svnbook.red-bean.com/en/1.4/

Peruskäyttö

- Tallennetaan varastoon (repository) ensimmäinen versio
- Haetaan varastosta paikallinen kopio
- Muokataan
- Tallennetaan muutokset varastoon (tarvittaessa konfliktit ratkoen)

Varaston luominen

- Subversion-varaston luominen täytyy tehdä luonnollisesti vain yhden kerran
  - svnadmin create /polku/luotavaan/varastoon
  - Voi olla, että atk-tuki luo teidän projektianne varten varaston palvelinkoneelle valmiiksi, mutta omaa kikkailua varten voi tehdä leluvaraston työasemalle

Varaston osoite

- Varaston osoite koostuu protokolla-osasta sekä varsinaisesta osoitteesta
- file://-alkuinen osoite tarkoittaa samalla koneella sijaitsevaa varastoa
  - Windows-koneella osoitteet muotoa

    file://X:/kansion/nimi
  - Unix-koneella osoitteet muotoa
    - file:///hakemiston/nimi
- Varastoon voidaan olla yhteydessä verkon yli, esimerkiksi svn+ssh:// (jota käytätte atk-tuen tarjoaman varaston kanssa)

Ensimmäisen version tallentaminen

- Tallennetaan ensimmäinen versio (hakemisto “sovellus”) Subversion-varastoon:
  - svn import -m “Ensimmäinen versio” sovellus file:///varaston/polku/projektin_nimi
- Jokaiseen tallennukseen liittyy viesti ja hyvien, kuvaavien viestien liittäminen on tärkeä osa versiohallinnan järkevää käyttöä!
  - Älkää missään nimessä jättäkö viestejä tyhjäksi!

Paikallisen kopion hakeminen

- Haetaan (checkout) varastoon tallennettu ensimmäinen versio paikalliseksi työstettäväksi kopioksi:
  - svn checkout file:///varaston/polku/projektin_nimi/trunk
  - checkout voidaan lyhentää co

Varaston rakenne

- Subversion ei pakota mitään tiettyä varaston rakennetta, mutta yleinen käytäntö on tehdä yhden varaston sisäinen jako seuraavasti:

```
projektin_nimi/
    trunk/
    tags/
    branches/
```

Trunk, tags & branches

- Trunk on kehityksen “päälinja” tai “kehityshaara”.
- Tagit ovat nimettyjä kopioita päälinjan tilasta jonain tiettynä ajanhetkenä. Esimerkiksi julkaisuversiota varten voidaan varaston tila “jäädyttää” tiettynä hetkenä ja jatkaa silti kehitystä trunkissa.
- Branchit ovat kehityshaaroja. Kehitys voidaan haaroittaa vaikkapa uutta isoa ominaisuutta varten.

Tagin luominen

- Tagi trunkin eli päähaaran tai jonkin branchin tilasta luodaan komennolla
  - svn copy file:///varaston/polku/trunk file:///varaston/polku/tags/release-070926 -m “Tagi 070926-julkaisua varten.”

Varastoon tulleiden muutosten haku

- Paikallista työkopiota tulee päivittää aika-ajoin varastoon tulleiden muutosten osalta:
  - svn update
- Päivityksessä varastoon muuttuneet tiedot yhdistyvät paikallisen kopion muutoksiin
- Voi syntyä konfliktitilanteita

Konfliktit

- Konflikti syntyy, kun kaksi käyttäjää muokkaa samaa riviä yhtäaikaisesti
- Versiohallinta osoittaa ongelmakohdat ja ne pitää ratkaista käsin ennen kuin muutokset voi tallentaa varastoon

```
Tämä on tiedosto
jossa on
<<<<<<< .mine
nel-
jä
=======
kolme
>>>>>>> .r3
riviä.
```

Konfliktien ratkaiseminen

- Konfliktissa olevasta tiedostosta syntyy kolme uutta tiedostoa: tiedoston_nimi.mine, tiedoston_nimi.rVANHA ja tiedoston_nimi.rUUSI.
  - .mine sisältää työkopiossa muokatun version
  - .rVANHA sisältää version ennen konfliktoivia muokkauksia
  - .rUUSI sisältää juuri varastosta saadun muokatun version

Konfliktin ratkaiseminen

- Konflikti ratkaistaan muuttamalla varsinainen tiedosto kuntoon (ei sisällä enää konfliktimerkintöjä ja sisältö on halutunlainen) ja komentamalla
  - svn resolved tiedoston_nimi
  - Tämän jälkeen pitää suorittaa vielä commit
- Joissain tapauksissa konfliktin “oikea” ratkaisu on vanhan tai uuden version kopiointi työkopion päälle, mutta usein täytyy tehdä käsityötä ja sovittaa konfliktoivat muutokset yhteen

Muutoksen tallentaminen varastoon

- Muuttunut tiedosto tallennetaan varastoon komennolla commit:
  - svn commit tiedosto -m “Lisätty pari juttua.”
  - commit lyhentyy ci
- Ennen commitia tulee aina tehdä päivitys (update)

Mitä tulikaan tehtyä?

- status- ja diff-komentojen avulla pystytään jäljittämään työkopioon tehtyjä muutoksia:
  - svn status
  - svn diff tiedosto.txt
- log-komento kertoo varastoon tulleista versioista tiedot

Tiedoston lisääminen varastoon

- Kokonaan uudet tiedostot lisätään komennolla add:
  - svn add uusi_tiedosto_tai_hakemisto
- Add on vasta muutos paikalliseen kopioon, joten sen jälkeen pitää suorittaa vielä commit, jotta uusi tiedosto siirtyy myös varastoon
- Vastaavasti löytyy komennot delete, copy ja move. Toiminta lienee ilmeinen ja tarkat ohjeet löytyvät Svnbookista

Versionumerot

- Joka kerta, kun varaston tila muuttuu (esimerkiksi commitin seurauksena), sen versionumero kasvaa
- Kaikki varaston tilan muutokset tallentuvat ja mihin tahansa vanhaan versioon voidaan palata
- checkout-komento voidaan parametrisoida esimerkiksi tietyn ajankohdan mukaan, jolloin päästään historiatietoihin käsiksi

Versiohallinnan integrointi kehitysympäristöön

- Versiohallintaa ei tarvitse välttämättä käyttää komentoriviltä vaan useimmissa (käytännössä kaikissa) kehitysympäristöissä on mahdollisuus käyttää versiohallintaa graafisella käyttöliittymällä
- Esimerkiksi Eclipseen Subversion-liittymän saa Subclipse-pluginilla:

  http://subclipse.tigris.org/

Apuja

- Komento help auttaa ongelmatilanteissa:
  - svn help
  - svn help komento
- Vilkuilkaa Svnbookia. Se paljastaa kaiken

Harjoituksia

- Luo oma varasto
- Luo oma “projektihakemisto” suositellun varastorakenteen mukaisesti (trunk, tags & branches) ja sinne vaikkapa tekstitiedosto
- Vie ensimmäinen versio varastoon
- Tee kaksi checkoutia eri hakemistoihin (simuloidaan useamman kehittäjän tapausta)
- Kokeile tehdä muutos tiedostoon, tee päivitys ja commit

Harjoitukset jatkuvat

- Tee päivitys molempiin työkopioihin ja muuta molemmista tekstitiedoston samaa riviä
- Tarkasta status- ja diff-komentojen avulla tehdyt muutokset
- Tee molemmista commit
- Ratkaise konfliktitilanne ja tee uudestaan päivitys sekä commit
- Totea log-komennolla, että varaston tila on muuttunut

Subversionin käyttö SSH-yhteyden yli

- Projektit käyttävät Subversionia toisella koneella sijaitsevaa varastoa verkon yli
- Turvalliseen tiedonsiirtoon käytetään SSH-protokollaa
- file://-alun sijaan varaston osoite on muotoa svn+ssh://palvelimen.osoite/hakemisto

SSH-avaimet

- Jotta SSH-yhteyden kanssa ei tarvitse joka välissä syöttää salasanaa, kannatta luoda ns. SSH-avaimet, joilla tunnistautuminen hoidetaan automaattisesti
- Ohjeet avainten luontiin Windows- ja Unix-ympäristöihin:
  - http://www.mit.jyu.fi/atk-tuki/ssh.html
  - http://www.cc.jyu.fi/~mmhilleb/avaimet.html