`cvs(1)` - Mikä se on?

Pekka Kanerva
`pekka@tcm.hut.fi`

v1.0 22.4.1999
v1.1 29.6.2000

1 Mitä missä milloin

`cvs(1)` on **versionhallintajärjestelmä**

- Esimerkkitilanne: koodiin tulee bugeja. `cvs(1)`:ää käyttäen voidaan helposti hakea se vanha versio, jossa bugi on koodiin aiheutunut
- Toki voidaan aina tallettaa jokaisen tiedoston jokainen versio... Tämä ei kuitenkaan ole kovin järkevää eikä edes helppoa. `cvs(1)` toimii fiksummin; se tallettaa jokaisesta tiedostosta vain yhden version ja pitää kirjaa tiedostojen muutoksista eri versioiden välisten differenssien avulla
- `cvs(1)` mahdollistaa usean henkilön samoja tiedostoja koskevan yhtaikaisen kehitystyön. Emacs osaa varmistaa ettei samaa tiedostoa yhtaikaisesti käsittelevät henkilöt ylikirjoita toistensa tekemiä muutoksia. Valitettavasti muut editorit eivät tätä osaa, vaan muutoksia saattaa hävitä ylikirjoitusten takia. `cvs(1)` on ratkaisu ongelmaan.

`cvs(1)` **ei ole**

- build-järjestelmä. Tämä hoidetaan `make(1)`:n ja `Makefile`:ien avulla
- korvike ohjelmoinnin manageroinnille
- korvike ohjelmistosuunnittelijoiden väliselle kommunikaatiolle
- ei sisällä toiminnallisuutta muutosten hallintaan
  esim. muutokset useaan lähdekooditiedostoon saattavat aiheuttaa toteutettavan ohjelmiston toiminnoissa loogisen muutoksen. tätä `cvs(1)` ei havaitse
- automaattinen testausohjelmisto
- ei sisällä valmista prosessimallia. Jotkin järjestelmät tarjoavat menetelmiä varmistaa, että muutokset käyvät läpi tietyt tarkistus- ja valvonta-askeleet ennen kuin muutokset voidaan hyväksyä. Tiettyjä tällaisia toiminteita *voidaan* toteuttaa `cvs(1)`:llä, mutta se vaatii jonkin verran lisää työtä.

- ihmeitä tekevä järjestelmä. Mikäli kaksi eri henkilöä editoi *samaa* tiedostoa samanaikaisesti tehden päällekkäin meneviä muutoksia, ei `cvs(1)` osaa yhdistellä näitä älykkäästi. `cvs(1)` ei voi tietää kumman tekemät muutokset ovat tärkeämpiä tai paremmin ajantasalla olevia. Tällaisten tilanteiden yhteydessä eri versioiden yhdistely on tehtävä ihmisvoimin.

1.1 `cvs` ja TeSSA

Viu loi uuden tunnuksen `tessa` jolle ei ole salasanaa → kukaan ei voi logata sisään ko. tunnukselle. `~tessa`-hakemistoon on kuitenkin 770-oikeudet, ja `tessa`-ryhmä omistaa hakemiston. Näinollen kaikki ryhmän jäsenet voivat siihen kirjoittaa. Uudet ryhmän jäsenet pitää kuitenkin lisätä ko. ryhmään, sillä nykyinen status on

```
tessa::104:sos,ursula,yki,atk,thasu,juho,pekka,telo,mlaukka,oskari,
                karvonen,tslampin,pnr,arissane,llehtola
```

Täydellinen polku `tessa`-hakemistoon on siis `/u/home/tessa`. Loin tämä alle edelleen `cvs`-repositorylle alihakemiston `cvsroot`, johon (ainakin pitäisi...) on kaikilla ryhmän jäsenillä kaikki oikeudet. Samoin kaikkien `cvsroot`:in alihakemistojen oikeuksiksi tulee automaagisesti yllämainittu `770`

2 “`cvs` in 21 days”

`cvs(1)` on niin laaja ohjelmistokokonaisuus, että sen lyhyeen kuvaukseen ei riitä edes yksi sivu. Onneksi `cvs`:n käyttö ei edellytä kaikkien hienouksien hallitsemista. Peruskomentojen osaaminen mahdollistaa `cvs`:n tehokkaan käytön versionhallinnassa. Seuraavassa esittelen pääpiirteittäin viisi peruskomentoa:

`cvs checkout` *modules ...*
: Tämä on välttämätön komento ennen kaiken työskentelyn aloittamista. `checkout` hakee ja luo työskentelykopion halutuista moduuleista omalle kovalevylle. Tätä paikallista kopiota voi editoida ja muuttaa häiritsemättä muiden työskentelyä. `checkout` luo aina vähintään yhden uuden alihakemiston.

`cvs update`
: Kun olet hakenut repositorystä työkappaleet omalle kovalevyllesi, jatkavat muut työskentelyä mahdollisesti samojen tiedostojen kanssa. `update` päivittää muiden mahdollisesti tekemät muutokset omiin työkappaleisiisi.

`cvs add` *file ...*
: Tätä käskyä käyttäen voit lisätä uusia tiedostoja versionhallintaan. Huomaa, että varsinaiseen repositoryyn muutokset tapahtuvat *vasta*, kun ajat `cvs commit`-käskyn! Mikäli halutaan lisätä uusia sorsia repositoryyn, tulee käyttää `cvs import`-käskyä. `cvs add` on tarkoitettu vain yksittäisten tiedostojen lisäämiseksi sellaisiin moduleihin, jotka on jo aiemmin kopioitu työkappaleiksi repositorystä.

`cvs remove` *file* ...
: Poistaa tiedostoja, jotka haluat hävittää. Muutokset repositoryssä tapahtuvat vasta, kun ajat `cvs commit`-käskyn.

`cvs commit` *file* ...
: Tätä käskyä käytetään, kun halutaan "julkaista" tehdyt muutokset muille repositoryn käyttäjille.

3 Graafiset käyttöliittymät

Sekä jCVS että tkCVS ovat laitteistoriippumattomia (Tcl/Tk on saatavissa UNIXeille, Windozelle ja Macintoshille). Näin ollen niitä voi ajaa kaikissa käyttämissämme järjestelmissä.

Toinen tapa olisi pyytää/anella/vaatia/toivoa ylläpitoa asentamaan jompi kumpi tai molemmat serverille ja ajaa niitä suoraan caffeinesta. (Windoze & Exceed). Tällöin voitaisiin

- hoitaa koodaus yhdellä alustalla (Caffeine/Solaris) paitsi niiltäosin kun pitää kommunikoida kortinlukijan kanssa
- ei tarvitsisi taikoa jokaiseen koneeseen erikseen GUI-systeemiä
- *luultavasti* graafisten clienttien liittäminen/käyttäminen ja toimimaan saaminen olisi mahdollisimman helppoa, kun ei tarvitsisi liikkua verkon yli koneesta toiseen.

4 Lisätietoja

1. `cvs`:n manuaalisivut: `cvs(1)` ja `cvs(5)`
2. Kattavin opas `cvs`:ään on Per Cederqvistin ja kumppaneiden kirjoittama *Version Management with CVS*
   `http://www.loria.fr/~molli/cvs-index.html`
3. Yahoo!:n versionhallintaa käsittelevä haara sisältää muutamia `cvs`:ää sisältäviä linkkejä
   `http://dir.yahoo.com/Computers_and_Internet/Software/Programming_Tools/Software_Engineering/Configuration_Management/CVS___Concurrent_Versions_System/`
4. jCVS: `cvs`:ää käyttävä Java-client löytyy osoitteesta
   `http://www.ice.com/java/jcvs/`
5. tkCVS: Tcl/Tk-pohjainen client löytyy osoitteesta
   `http://www.NeoSoft.com/tcl/ftparchive/sorted/apps/tkcvs-6.0/`
6. Graafisita käyttöliittymistä ja vähän muustakin `cvs`:ään liittyvästä löytyy tietoa ja kuvausta (pakolliset screenshotit) myös osoitteesta
   `http://www.cyclic.com/cvs/linux-gui.html`