UKEOPPGAVER 13: KONFIGURASJONSSTYRING

Formål: I denne oppgaven skal dere få litt hands-on med versjonskontrollsystemet Subversion. Meningen er at du skal prøve å relatere prinsippene det ble forelest om til konkrete bruksscenarier i et enkelt, men mye brukt versjonskontrollsystem. Det gis ikke løsningsforslag, da de fleste detaljer er beskrevet i oppgaveteksten.

Forutsetninger: Oppgaveteksten vil bruke begreper fra forelesning

---

Innledning

I denne oppgaven skal du bli kjent med versjonskontrollsystemet Subversion. Dere skal jobbe i par. Hvert par skal jobbe mot hvert sitt Subversion-repository, men hver person skal ha sitt lokale arbeidsområde. Dette kan gjøres enten ved at dere sitter på hver deres maskin, eller at dere har hvert deres linux-shell på sammen maskin. Hvis du ikke finner en partner kan du simulere en partner ved å ha to linux-shell oppe, og jobbe på to ulike kataloger.

Bruk gjerne online-ressurser som http://svnbook.red-bean.com/en/1.4/index.html underveis.

Subversion har en klient-tjener arkitektur, der kommunikasjon mellom klient og tjener kan foregå via protokoller som http eller ssh. Det er kjempeviktig at du har klart for deg forskjellen på å gjøre endringer lokalt i ditt arbeidsområde (klient), og å gjøre endringer i det sentrale repositoriet (server).

Oppgave 1. Opprette et Subversion-repository

Vanligvis gjøres dette fra kommandolinje, med `svnadmin create <sti>`.

På ifi har man et webgrensesnitt for å gjøre dette:

Gå til
https://wwws.ifi.uio.no/system/svn

Velg ”Create new repository”
Skriv inn navnet inf1050 og trykk *submit*

Oppgave 2. Lage en standard initiell struktur i repositoriet

Fra nå av skal du ikke bruke web-grensesnittet lenger. Subversion er kommandolinje-orientert, men det finnes også grafiske add-ons som (kanskje) gjør bruken enda enklere.
En standard måte å organisere et Subversion-repository på er for hvert prosjekt/system å ha underkatalogene *trunk*, *branches* og *tags*. Trunk er navnet for ”mainline” i Subversion, her foregår normal videreutvikling. Forgreninger samles under katalogen branches. Tags brukes til å navngi systemversjoner. Les eventuelt mer på http://svnbook.red-bean.com/en/1.2/svn.branchmerge.maint.html

En av dere lager nå disse tre katalogene lokalt (under et fritt valgt område, eks. inf1050). Kommandoen gir du i et linux-shell.

```
inf1050> mkdir trunk
inf1050> mkdir branches
inf1050> mkdir tags
```

og gir kommandoen

```
inf1050> svn import . -m "ny import"
https://svn.ifi.uio.no/repos/users/<user>-inf1050
```

<user> i kommandoen over erstattes med ditt brukernavn. Dere kan nå sjekke at det sentrale repositoriet deres inneholder de riktige filene:

```
inf1050> svn list https://svn.ifi.uio.no/repos/users/<user>-inf1050
```

Mer info kan dere få om endrede filer i lokalt arbeidsområde med:

```
inf1050> svn status .
```

Og mer versjonshistorikk om hvert enkelt element med

```
inf1050> svn -v status .
```

Oppgave 3. Etablere lokale arbeidsområder

Hver av dere gjør følgende:

Gå til en passende katalog der du skal ha ditt arbeidsområde. For deg som opprettet repositoryet unngår du trøbbel hvis du nå oppretter en ny katalog, der du henter en fresh versjon av det som nå ligger i repositoriet.

```
svn checkout https://svn.ifi.uio.no/repos/users/<user>-inf1050
```

Du vil nå se at du har fått et lokalt hierarki som ser ut som følger

<user>-inf1050 |- trunk
|- branches
|- tags

Oppgave 4. Opprette nye filer under versjonkontroll
En av dere gjør følgende:

Opprett to filer i katalogen ”trunk”:

```
<user>-inf1050> echo "innhold i fil1.txt" > trunk/fil1.txt
<user>-inf1050> echo "innhold i fil2.txt" > trunk/fil2.txt
```

La Subversion vite at dette er Subversion-filer:

```
<user>-inf1050> svn add trunk/fil1.txt
<user>-inf1050> svn add trunk/fil2.txt
```

Og commit (checkin) til repository:

```
<user>-inf1050> svn commit -m ’Nye filer’
```

Den av dere som ikke gjorde de foregående steg kan nå gjøre
```
<user>-inf1050> svn update
```

Denne kommandoen oppdaterer arbeidsområdet ditt med nye filer og versjoner fra repository

Oppgave 5. Normal parallell utvikling

Subversion bruker en optimistisk strategi for parallell utvikling, som betyr at endringer kan gjøres parallelt på samme fil av flere utviklere. Merging og eventuell konflikter må håndteres ved commit. I praksis viser det seg at dette fungerer (overraskende) bra i de fleste tilfeller.

Simuler nå seriøs parallell systemutvikling som følger:
- En av dere endrer litt tekst i trunk/fil1.txt og trunk/fil2.txt (bruk en vanlig editor)
- Den andre endrer litt tekst i trunk/fil1.txt

Begge prøver så å overføres sine endringer til repository:
```
 <user>-inf1050> svn commit –m 'noen endringer…'
```

En av dere vil nå få en melding:
```
svn: Commit failed (details follow):
svn: Your file or directory 'fil1.txt' is probably out-of-date
svn: resource out of date; try updating
```

Dette betyr at det har vært gjort en commit på fila siden du hentet den fra repository. For å hente (og eventuelt) merge endringer inn i ditt arbeidsområde fra repository brukes igjen svn update:

```
<user>-inf1050> svn update
```

De parallelle endringene i fil1.txt vil Subversion prøve å merge automatisk. Hvis det blir en merge-konflikt vil du se det av innholdet i fil1.txt. I så fall må disse rettes manuelt. Til slutt gjør den av dere som fikk konflikt:

```
<user>-inf1050> svn commit –m 'håndtert konflikter…'
```

Oppgave 6. Opprette en branch

Vi ser nå for oss at en av dere skal videreutvikle "systemet", men den andre skal gjøre en bugfix.

En av dere fortsetter derfor å jobbe i trunk, mens den andre må opprette en bugbranch. Det siste gjøres i Subversion med:

<user>-inf1050> `svn copy -m 'lager bugbranch'`
```
https://svn.ifi.uio.no/repos/users/<user>-inf1050/trunk
https://svn.ifi.uio.no/repos/users/<user>-inf1050/branches/v1Bugs
```

Du som skal jobbe i bugbranch henter så inn filer til ditt lokale arbeidsområde
```
<user>-inf1050> svn update
```

og går til v1Bugs-katalogen

```
<user>-inf1050> cd branches/v1Bugs
```

Så editerer bugutvikler fila fil2.txt, slik at den inneholder disse to linjene (og ingen blanke):

```
tillegg i bugbranch
innhold i fil2.txt
```

trunk-utvikler gjør følgende endringer i trunk/fil2.txt.

```
innhold i fil2.txt
tillegg i trunk
```

Hver av dere gjør følgende for å sjekk inn endringer.

Hver av dere gjør nå

```
svn commit -m 'endret i fil2.txt'
```

Oppgave 7. Merge branch inn i trunk (Bedre norsk: Flette forgrening inn i hovedlinje)

Denne oppgaven begynner å bli avansert. Hvis du er i ferd med å få liten tid, holder det at du leser og tenker gjennom oppgaveteksten. Merging av branch inn i trunk er kanskje den mest brukte av de mer avanserte features i versjonskontrollsystemer. Subversion har en litt keitete måte å gjøre dette på, men alt blir en vanesak:

En slik merge-operasjon i Subversion kan beskrives som å **identifisere endringene** som er gjort i bug-branch og så **applisere** disse endringene på siste versjon i trunk. Den vanligste måten å gjøre dette på er å finne versjonsnummeret der branchen ble laget, og sammenligne med versjonen HEAD (spesialsymbol for **siste endring**). Versjonsnummeret er globalt for hele repositoriet i Subversion: For hver gang man gjør en endring i repositoriet inkrementeres versjonsnummeret med 1.

Versjonsnummeret der bugbranchen ble laget kan finnes ved:

```
<user>-inf1050> svn log --stop-on-copy
https://svn.ifi.uio.no/repos/users/<user>-inf1050/branches/v1Bugs
```

Den siste linja i listingen du da får vil inneholde revisjonsnummeret (med en r foran) der angitt bugbranch ble laget

En av dere gjør nå følgende:

```
<user>-inf1050> cd trunk
<user>-inf1050/trunk> svn merge -r 10:HEAD (10 byttes ut med ditt nummer)
https://svn.ifi.uio.no/repos/users/<user>-inf1050/branches/v1Bugs
```

Denne kommandoen finner alle endringene mellom 10 og HEAD i bugbranch og appliserer dem til siste versjon i trunk (katalogen du står i).

Du kan nå inspisere filene som rapporteres endret. Hvis du har fulgt oppskriften nøyaktig vil fil2.txt se ut som følger:

```
bug 1 fix
innhold i fil2.txt
trunk additions
```

Dette er teknisk sett er en korrekt merging av de to branchene. Hvis det dreier seg om programkode er du imidlertid ikke garantert at resultatet er korrekt.