Facultatea Calculatoare, Informatica si Microelectronica

Universitatea Tehnica a Moldovei

Medii Interactive de Dezvoltare a Produselor Soft

Lucrarea de laborator#1

# Version Control Systems si modul de setare a unui server

*Autor:*
Cristian Godonoaga

*lector asistent:*
Irina Cojanu
*lector superior:*
Radu Melnic

**Laboratory work #1**

## 1 Scopul lucrarii de laborator

Studierea unui sistem de control a versiunelor (**Version Control System**) utilizind linia de comanda (**Command Line Interface**).

## 2 Obiective

- Insusirea modului de utilizare a celor mai importante componente a unui VCS descentralizate.

## 3 Laboratory work implementation

### 3.1 Tasks and Points

Basic Level (nota 5 - 6) :

initializeaza un nou repositoriu
configureaza-ti VCS
crearea branch-urilor (creeaza cel putin 2 branches)
commit pe ambele branch-uri (cel putin 1 commit per branch)

Normal Level (nota 7 - 8):

seteaza un branch to track a remote origin pe care vei putea sa faci push
reseteaza un branch la commit-ul anterior
salvarea temporara a schimbarilor care nu se vor face commit imediat
folosirea fisierului .gitignore

Advanced Level (nota 9 - 10):

merge 2 branches
rezolvarea conflictelor a 2 branches
comezile git care trebuie cunoscute

Bonus Point (+1):

Folosirea tag-urilor pentru marcarea schimbarilor simnificative precum release-ul.

### 3.2 Analiza lucrarii de laborator

Inainte de a incepe indeplinirea sarcinelor au fost efectuati citiva pasi aditionali, ce tin de pregatirea calculatorului pentru a putea utiliza acest sistem de versionare a codului. Am ales un sistem descentralizat deoarece acesta ne ofera posibilitati mai ample, referitor la controlul distribuit al versiunelor, gestionarea si revizuirea codului cit si vizualizarea activitatii proectului fata de sistemul centralizat. In special acesta fiind un model de source control ce permite partajarea surselor in mod distribuit intre membrii echipelor fara a depinde de un repozitoriu central.

*Instalarea si setarea sistemului Git sa efectuat conform manualului oferit de situl oficial git-scm.com.*

Utilizind serviciile prestate de reteaua sociala (GitHub) pentru proiecte cu versionarea bazata pe Git, am creat un cont pe *www.github.com* care detine repozitoriul laboratorului efectuat.

 Link la repozitoriu: `https://github.com/CristianGodonoaga/MIDPS`

Pentru a asigura o comunicare securizata a datelor intre doua statii am utilizat protocolul de retea SSH. Acesta se foloseste de criptarea cu ***chei asimetrice*** ce furnizeaza un mod mai sigur de logare (comunicare) intr-un server cu SSH decit folosind o parola obisnuita. La generarea acestor

perechi de key se utilizeaza un mecanizm special care ne ofera doua siruri lungi de caractere: o cheie publica si o cheie privata.

Intrun prim pas se creaza perechea de chei pe masina client:

```
$ssh-keygen -t rsa                                          (Fig. 3.1)
```

Dupa ce am introdus comanda de generare a cheii am primit citeva intrebari ce tine de marirea securitatii prin protejarea cheii privade, cu o parola de acces. Odata ce perechea de chei este generata plasam cheia publica pe serverul GitHub (Fig. 3.2).

Pentru a incepe utilizarea sistemului Git, mai intii este nevoie de setarea configuratiei de baza. Aceste configurari pot fi facute in trei modalitati, ele vor determina cit de amplu va fi aplicarea acestora. Utilizind comenzile:

```
$git config --global user.name "Godonoaga C."
$git config --global user.email "some@email.com"            (Fig. 3.3)
```

am setat numele si posta la nivel de utilizator, ce ne permite utilizarea acestor date in viitor pentru oarecare alt proiect al acestuia fara a le specifica in parte. Deoarece aceste setari depind foarte mult de necesitatea utilizatorului nu exista o varianta exacta ce tine de setarea sistemului Git.

Pentru a da start sistemului si anume pentru a monitoriza schimbarile in proiectul necesar se utilizeaza comanda `$git init`, aceasta va insemna ca noi avem nevoie de monitorizarea schimbarilor a tuturor fisierilor ce se afla in folderul curenta (creareaza un repository).

Din acest moment putem incepe versionarea proiectului nostru insa mai este un pas ce tine de setare, nu toate fisierele ce sunt in proiectul nostu au nevoie de versionare, unele chear fiind executabile generate de procesul de compilare sau efectiv parametrii de configurare ai aplicatiei. Pentru aceasta fisierele sau directoarele ce se doresc a fi ignorate de sistemul de versionare se noteaza intr-un fisier numit `.gitignore`. Voi plasa acest fisier in radacina proiectului pentru a nu cauta foarte mult path-urile ignorate, deoarece acesta este evaluat in mod recursiv si putem avea mai multe fisiere .gitignore.

Salvarea starii fisierelor sau intregului proiect se efectueaza utilizint comenzile de baza pe care le contine Git.

**git command –example ”test”**

Explica fiecare din task-urile realizate. Explica acesta cu cuvintele tale. Cu cit mai bine va fi explicat, cu atit mai putine intrebari vei avea la aparare.

```
git config —global user.name ””
```

```
git config git code
```

### 3.3 Imagini

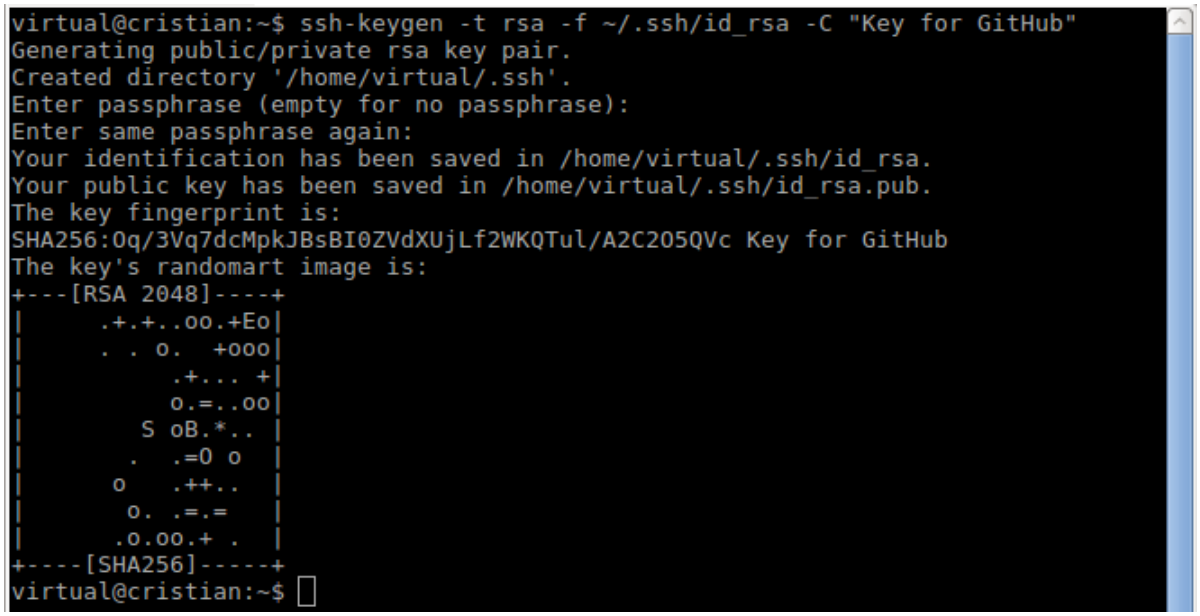


Figure 3.1– Generarea ssh-key

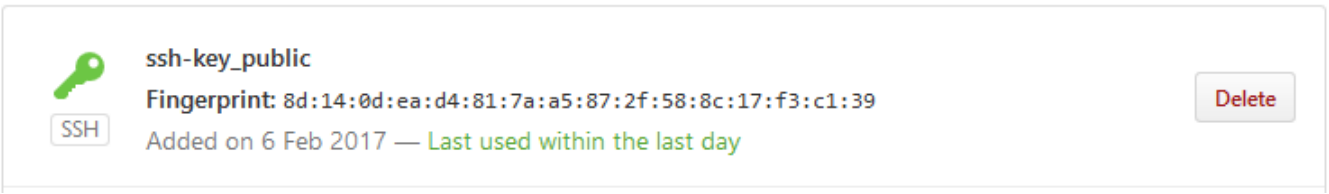


Figure 3.2– Adaugarea ssh-key pe server

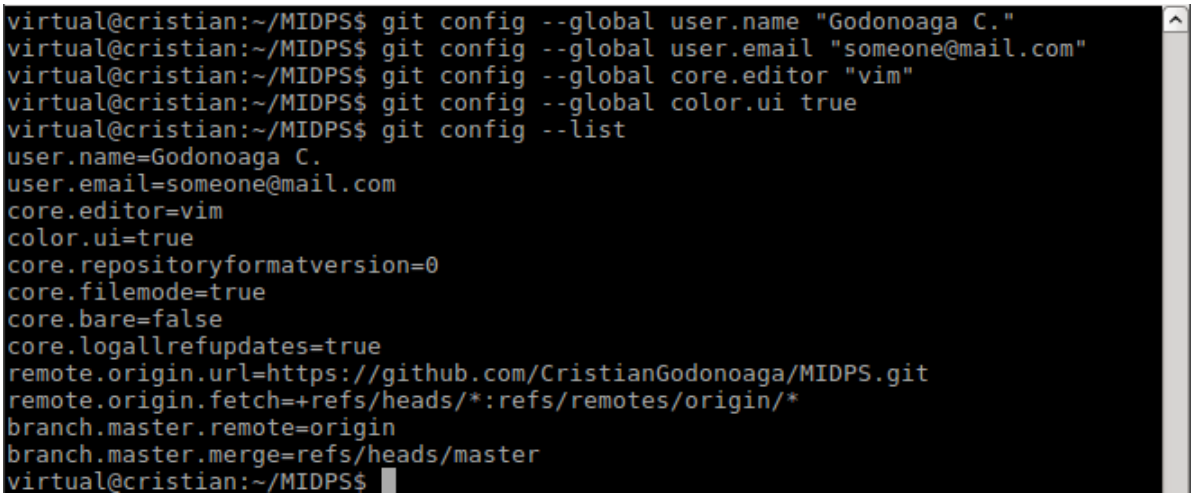


Figure 3.3– Configurare VCS

### Concluzie

Aici trebuie sa fie concluzia ta.

## References


1 Aldebran Robotics, *official page*, www.aldebaran.com/en

2 Timo Ojala, *Multiresolution gray-scale and rotation invariant texture classification with local binary patterns*, 2002

3 Biometric, www.biometricupdate.com/201501/history-of-biometrics