

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 lucrării 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 repository
- 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 semnificative precum release-ul.

3.2 Analiza lucrarii de laborator

Înainte de a începe îndeplinirea sarcinilor au fost efectuați câțiva pași adiționali, ce țin de pregătirea calculatorului pentru a putea utiliza acest sistem de versionare a codului. Am ales un sistem descentralizat deoarece acesta ne oferă posibilități mai ample, referitor la controlul distribuit al versiunilor, gestionarea și revizuirea codului și vizualizarea activității proiectului față de sistemul centralizat. În special acesta fiind un model de source control ce permite partajarea surselor în mod distribuit între membrii echipelor fără a depinde de un repository central.

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

Utilizând serviciile prestate de rețeaua socială (GitHub) pentru proiecte cu versionarea bazată pe Git, am creat un cont pe [www.github.com](https://github.com) care deține repositoryul laboratorului efectuat.



Link la repository: <https://github.com/CristianGodonoaga/MIDPS>

Pentru a asigura o comunicare securizată a datelor între două stații am utilizat protocolul de rețea SSH. Acesta se folosește de criptarea cu **chei asimetrice** ce furnizează un mod mai sigur de logare (comunicare) într-un server cu SSH decât folosind o parolă obișnuită. La generarea acestor

perechi de key se utilizeaza un mecanism 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 private, 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 fisierelor 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 (snapshot) sau intregului proiect se efectueaza utilizint comenzile de baza pe care le contine Git. Ele ne permit gestionarea procesului de versionare. La indeplinirea lucrarii de laborator sunt utilizate doar o parte, insa ele contin o multime destul de vasta si este nevoie de practica pentru a percepe modul de functionare si pentru a capata deprindere de utilizare.

Un alt mecanism ce ne ofera sistemul git este ramificarea acestui proces de versionare. Posibilitatile oferite de acesta sunt multe iar baza fiind clonarea starii unei ramuri, unde pot fi aplicate modificari care nu afecteaza ramuza parinte.

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 command —example "test"**


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

```
git config git code
```

3.3 Imagini

```
virtual@cristian:~$ ssh-keygen -t rsa -f ~/.ssh/id_rsa -C "Key for GitHub"
Generating public/private rsa key pair.
Created directory '/home/virtual/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/virtual/.ssh/id_rsa.
Your public key has been saved in /home/virtual/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:0q/3Vq7dcMpkJBsBI0ZVdXUjLf2WKQTul/A2C205QVc Key for GitHub
The key's randomart image is:
+---[RSA 2048]---+
|      .+..+Eo|
|      . .  +00|
|      .+... +|
|      o.=..oo|
|      S oB.*..|
|      . . =0 o|
|      o .++..|
|      o. . =.|
|      .o.oo.+ .|
+-----[SHA256]-----+
virtual@cristian:~$
```

Figure 3.1 – Generarea ssh-key

**ssh-key_public**

SSH

Fingerprint: 8d:14:0d:ea:d4:81:7a:a5:87:2f:58:8c:17:f3:c1:39
Added on 6 Feb 2017 — Last used within the last day

Delete

Figure 3.2 – Adaugarea ssh-key pe server

```
virtual@cristian:~/MIDPS$ git config --global user.name "Godonoaga C."
virtual@cristian:~/MIDPS$ git config --global user.email "someone@mail.com"
virtual@cristian:~/MIDPS$ git config --global core.editor "vim"
virtual@cristian:~/MIDPS$ git config --global color.ui true
virtual@cristian:~/MIDPS$ git config --list
user.name=Godonoaga C.
user.email=someone@mail.com
core.editor=vim
color.ui=true
core.repositoryformatversion=0
core.filemode=true
core.bare=false
core.logallrefupdates=true
remote.origin.url=https://github.com/CristianGodonoaga/MIDPS.git
remote.origin.fetch=+refs/heads/*:refs/remotes/origin/*
branch.master.remote=origin
branch.master.merge=refs/heads/master
virtual@cristian:~/MIDPS$
```

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