

Ministerul Educației al Republicii Moldova
Universitatea Tehnică a Moldovei

RAPORT

Lucrarea de laborator nr. 2

Medii Interactive de dezvoltare a produselor software

Tema: Version Control Systems și modul de setare a unui server

A efectuat:

st. gr. TI-142

Chicu Roman

A verificat:

lect. univ.

Irina Cojan

Chișinău 2016

Tema: Version Control Systems și modul de setare a unui server.

Scopul lucrării: Studierea bazelor lucrului cu VCS. Obiectivele lucrării:

1. Înțelegerea și folosirea CLI (basic level)
2. Administrarea remote a mașinilor linux machine folosind SSH (remote code editing)
3. Version Control Systems (git || mercurial || svn)
4. Compileaza codul C/C++/Java/Python prin intermediul CLI, folosind compilatoarele gcc/g++/javac/python

Modul de lucru: Pașii

lucrării:

- conecteaza-te la server folosind SSH
- compileaza cel puțin 2 sample programs din setul HelloWorldPrograms folosind CLI
- executa primul commit folosind VCS
- initializeaza un nou repository
- configureaza-ti VCS
- crearea branch-urilor (creeaza cel puțin 2 branches)
- commit pe ambele branch-uri (cel puțin 1 commit per branch)
- seteaza un branch to track a remote origin pe care vei putea sa faci push (ex. Github, Bitbucket or custom server)
- reseteaza un branch la commit-ul anterior
- merge 2 branches
- rezolvarea conflictelor a 2 branches

Efectuarea Lucrării:

- ✓ A fost creat un repository pe github.com.
- ✓ A fost stabilită conexiunea cu serverul prin generarea keygen-ului SSH prin instrucțiunea ssh-keygen, ea fiind adăugată în setări.
- ✓ A fost creat un fișier și inițializat git-ul prin instrucțiunea git init.
- ✓ A fost introdus .gitignore și README.md prin instrucțiunea git add, git commit și git push.
- ✓ Au fost create două programe, una în C și alta în C++, cu extensia respectivă .c și .cpp
- ✓ Cu ajutorul la Command Prompt, ele au fost compilate cu instrucțiunea:

gcc Hello_C.c -o Hello_C și g++ Hello_C++.cpp -o Hello_C++ .

- ✓ Respectiv, au fost executate fişierele noi cu extensia .exe în Command Prompt, apelînd la ele prin: /Hello_C şi /Hello_C++.
- ✓ A fost creat un branch nou, unde au fost încărcate fişierele compilate în CLI, împreună cu imaginile respective şi fişierele originale .c şi .cpp.
- ✓ Am făcut merge dintre branch-ul master şi NBranch, unde se aflau fişierele compilate.
- ✓ Am creat o situaţie de conflict, creînd un fişier Conflict.txt şi încărcîndu-l pe branch master.
- ✓ Am creat un branch nou ConflictBranch, în care am modificat conţinutul fişierului Conflict.txt, astfel făcînd opţiunea merge imposibilă.
- ✓ Am rezolvat conflictul şi am făcut merge între branch-uri.

Screen-uri ale executarii comenzilor:

- Crearea unui nou Branch:

```
Roman@Roman-Pc MINGW32 /e/MIDPS/lab 2 (master)
$ git branch NBranch
```

- Trasnferul la Branch-ul nou:

```
Roman@Roman-Pc MINGW32 /e/MIDPS/lab 2 (NBranch)
$ git checkout master1
M      lab 2/source.exe
Switched to branch 'master1'
```

- Crearea si executarea Hello_C.c şi Hello_C++.cpp:

```
Roman@Roman-Pc MINGW32 /e/MIDPS/lab 2 (master1)
$ ./source
Buna ziua
Roman@Roman-Pc MINGW32 /e/MIDPS/lab 2 (master1)
$ ./source1
Buna ziua

C:\Users\Roman\Desktop\MIDPS>gcc source.c -o source
C:\Users\Roman\Desktop\MIDPS>g++ source.cpp -o source
```

- Adaugarea unui fisier .txt, care v-a fi resetat:

```
Roman@Roman-Pc MINGW32 /e/MIDPS/lab 2 (master1)
$ git status
On branch master1
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   source.exe

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        source1 .cpp
        source1 .exe
        source1 .o

no changes added to commit (use "git add" and/or "git commit -a")
```

- Resetarea la commit-ul precedent:

```
Roman@Roman-Pc MINGW32 /e/MIDPS/lab 2 (master1)
$ git reset --hard
HEAD is now at 85068b5 Schimbare
```

- Merge între două branch-uri:

```
Roman@Roman-Pc MINGW32 /e/MIDPS/lab 2 (master1)
$ git pull
Updating 45b603c..3fc8c3b
Fast-forward
 lab 2/branch.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 lab 2/branch.txt
```

- Afișarea situației de conflict:

```
$ git merge MBranch
CONFLICT (rename/delete): Lab#2/Used_Files/Branch1.txt deleted in
MBranch and renamed in HEAD. Version HEAD of Lab#2/Used_Files/Br
anch1.txt left in tree.
Automatic merge failed; fix conflicts and then commit the result.
```

Concluzie

În urma efectuării acestei lucrări de laborator au fost obținute deprinderi practice de lucru cu git – unul din cele mai populare DVCS. Am ajuns la concluzia că un VCS este un instrument indispensabil lucrului în echipă. Acest VCS este un instrument ce verifică versiunea proiectului, permitându-ne să evităm conflicte. Am observat că în git, lucru cu branch-urile se face cu mult mai simplu decât în alte VCS-uri cum ar fi SVN. Un lucru plăcut despre git este viteza și eficiența lui.

Repositoriile git-ului ocupa mai putin loc decit cele ale lui SVN, si ca git este distribuit, deci nu este nevoie de a fi mereu conectat la retea.