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 şi modul de setare a unui server

A efectuat:
st. gr. TI-144 Gorduz Daniel

A verificat:
lect. univ. Irina Cojan

Chișinău 2016

Tema: Version Control Systems şi modul de setare a unui server.
Scopul lucrării: Studierea bazelor lucrului cu VCS. Obiectivele lucrării:

1. Înţelegerea şi folosirea CLI (basic level)
2. Administrarea remote a maşinilor 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:

Paşii lucrării:

- conecteaza-te la server folosind SSH
- compileaza cel putin 2 sample programs din setul HelloWolrdPrograms folosind CLI
- executa primul commit folosind VCS
- 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)
- 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 Lucrarii:

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

gcc Hello_C.c –o Hello_C şi 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:

```
danie@Daniel MINGW64 /c/Users/danie/Desktop/MIDPS/Lab#2 (master)
$ git branch NBranch
```

- Trasnferul la Branch-ul nou:

```
danie@Daniel MINGW64 /c/Users/danie/Desktop/MIDPS/Lab#2 (master)
$ git checkout NBranch
D       Lab#1.pdf
Switched to branch 'NBranch'
```

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

```
danie@Daniel MINGW64 /c/Users/danie/Desktop/MIDPS/Lab#2 (master)
$ gcc Hello_C.c -o Hello_C

danie@Daniel MINGW64 /c/Users/danie/Desktop/MIDPS/Lab#2 (master)
$ g++ Hello_C++.cpp -o Hello_C++

danie@Daniel MINGW64 /c/Users/danie/Desktop/MIDPS/Lab#2 (master)
$ ./Hello_C
Hello World!
danie@Daniel MINGW64 /c/Users/danie/Desktop/MIDPS/Lab#2 (master)
$ ./Hello_C++
Hello world

danie@Daniel MINGW64 /c/Users/danie/Desktop/MIDPS/Lab#2 (master)
$
```

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

```
danie@Daniel MINGW64 /c/Users/danie/Desktop/MIDPS/Lab#2 (NBranch)
$ git status
On branch NBranch
Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        deleted:    ../Lab#1.pdf
        deleted:    text.txt

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

        Branch1.txt

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

danie@Daniel MINGW64 /c/Users/danie/Desktop/MIDPS/Lab#2 (NBranch)
$ git add .

danie@Daniel MINGW64 /c/Users/danie/Desktop/MIDPS/Lab#2 (NBranch)
$ git commit -m "added new branch file"
[NBranch 859ba33] added new branch file
 Committer: Daniel Gorduz <Daniel Gorduz>
```

- Resetarea la commit-ul precedent:

```
danie@Daniel MINGW64 /c/Users/danie/Desktop/MIDPS (NBranch)
$ git reset --hard 8a4ec6
HEAD is now at 8a4ec6  Adding the NBranch files to the first branch
```

- Merge între două branch-uri:

```
danie@Daniel MINGW64 /c/Users/danie/Desktop/MIDPS/Lab#2 (master)
$ git merge NBranch
Updating 8a4ec6e..859ba33
Fast-forward
 Lab#2/Branch1.txt | 1 +
 Lab#2/text.txt    | 1 -
 2 files changed, 1 insertion(+), 1 deletion(-)
 create mode 100644 Lab#2/Branch1.txt
 delete mode 100644 Lab#2/text.txt
```

- Afişarea situaţiei de conflict:

```
danie@Daniel MINGW64 /c/Users/danie/Desktop/MIDPS (master)
$ 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

In urma efectuarii acestei lucrari de laborator au fost obtinute deprinderi practice de lucru cu git – unul din cele mai populare DVCS. Am ajuns la concluzia ca un VCS este un instrument indispensabil lucrului in echipa. Acest VCS este un instrument ce verifică versiunea proiectului, permitîndu-ne să evităm conflicte

Am observat ca in git, lucru cu branch-urile se face cu mult mai simplu decit in alte VSC-uri cum ar fi SVN. Un lucru placut despre git este viteza si eficienta lui. Repositoriile git-ului ocupa mai putin loc decit cele ale lui SVN, si ca git este distribuit, deci nu este novoie de a fi mereu conectat la retea.