

RAPORT

La disciplina: Medii interactive de dezvoltare a produselor soft

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

A efectuat:
st.gr.TI-144

Olaru Cristian

A verificat:
lect.univ.

Cojocaru Svetlana

Lucrarea de laborator Nr.2

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

Scopul lucrării: Studiarea bazelor lucrului cu VCS.

Obiectivele lucrării:

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

Modul de lucru:

Pașii lucrării:

- conectează-te la server folosind SSH
- compilează cel puțin 2 sample programs din setul HelloWorldPrograms folosind CLI
- execută primul commit folosind VCS
- inițializează un nou repository
- configurează-ți VCS
- crearea branch-urilor (creează cel puțin 2 branches)
- commit pe ambele branch-uri (cel puțin 1 commit per branch)
- reseteaza un branch la commit-ul anterior
- merge 2 branches
- conflict solving between 2 branches
- Scrie un script care va compila HelloWorldPrograms projects.

Efectuarea lucrării:

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

gcc Hello1.c -o Hello1 și g++ Hello2.cpp -o Hello2

7. Respectiv, au fost executate fișierele noi cu extensia **.exe** în Command Prompt, apelînd la ele prin: **/Hello1** și **/Hello2**.
8. A fost creat un branch nou, unde au fost încărcate fișierele compilate în CLI, împreună cu imaginile respective și fișierele originale **.c** și **.cpp**.
9. Am făcut merge dintre branch-ul **master** și **CLIconpiling**, unde se aflau fișierele compilate.
10. Am creat o situație de conflict, creînd un fișier **Conflict.txt** și încărcîndu-l pe branch **master**.
11. Am creat un branch nou **ConflictBranch**, în care am modificat conținutul fișierului **Conflict.txt**, astfel făcînd opțiunea merge imposibilă.
12. Am rezolvat conflictul și am făcut merge între branch-uri.
13. A fost creat scriptul de compilare a programelor **run.bat**.

Imagini:

Inițializarea git-ului și adăugarea **.gitignore** și **README.md**

```
home@home-?? MINGW64 ~/Desktop
$ cd MIDPS

home@home-?? MINGW64 ~/Desktop/MIDPS
$ git init
Initialized empty Git repository in C:/Users/home/Desktop/MIDPS/.git/

home@home-?? MINGW64 ~/Desktop/MIDPS (master)
$ git status
On branch master

Initial commit

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

        .gitignore
        README.md

nothing added to commit but untracked files present (use "git add" to track)
```

Efectuarea commit-ului

```
home@home-?? MINGW64 ~/Desktop/MIDPS (master)
$ git add .

home@home-?? MINGW64 ~/Desktop/MIDPS (master)
$ git status
On branch master

Initial commit

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:   .gitignore
        new file:   README.md

home@home-?? MINGW64 ~/Desktop/MIDPS (master)
$ git commit -m "Adding .gitignore and README.md files"
[master (root-commit) 529fecc] Adding .gitignore and README.md files
2 files changed, 41 insertions(+)
create mode 100644 .gitignore
create mode 100644 README.md
```

Crearea unui nou branch

```
home@home-?? MINGW64 ~/Desktop/MIDPS (master)
$ git branch CLICompiling
```

Transferul la branch-ul nou

```
home@home-?? MINGW64 ~/Desktop/MIDPS (master)
$ git checkout CLICompiling
Switched to branch 'CLICompiling'
```

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

```
home@home-?? MINGW64 ~/Desktop/MIDPS (CLICompiling)
$ git status
On branch CLICompiling
Untracked files:
  (use "git add <file>..." to include in what will be committed)

    Laborator 2/GCC Compiler/Hello, is it me you're looking for.txt

nothing added to commit but untracked files present (use "git add" to track)
home@home-?? MINGW64 ~/Desktop/MIDPS (CLICompiling)
$ git add .
home@home-?? MINGW64 ~/Desktop/MIDPS (CLICompiling)
$ git commit -m "This is not the commit you're looking for."
[CLICompiling cb997e0] This is not the commit you're looking for.
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 Laborator 2/GCC Compiler/Hello, is it me you're looking for.txt
```

Resetarea la commit-ul precedent

```
home@home-?? MINGW64 ~/Desktop/MIDPS (CLICompiling)
$ git reset --hard d746a7f
HEAD is now at d746a7f Adding the CLI files to the first branch
```

Merge între două branch-uri

```
home@home-?? MINGW64 ~/Desktop/MIDPS (master)
$ git merge master CLICompiling
Updating 919d8cf..d746a7f
Fast-forward
 Laborator 2/GCC Compiler/Compiler.PNG | Bin 0 -> 114219 bytes
 Laborator 2/GCC Compiler/Hello1.c      | 9 ++++++++
 Laborator 2/GCC Compiler/Hello2.cpp    | 9 ++++++++
 Laborator 2/GCC Compiler/ReadMe.md     | 24 ++++++++
4 files changed, 42 insertions(+)
create mode 100644 Laborator 2/GCC Compiler/Compiler.PNG
create mode 100644 Laborator 2/GCC Compiler/Hello1.c
create mode 100644 Laborator 2/GCC Compiler/Hello2.cpp
create mode 100644 Laborator 2/GCC Compiler/ReadMe.md
```

Afişarea situaţiei de conflict.

```
home@home-?? MINGW64 ~/Desktop/MIDPS (master)
$ git merge ConflictBranch
Auto-merging Laborator 2/GCC Compiler/Conflict.txt
CONFLICT (add/add): Merge conflict in Laborator 2/GCC Compiler/Conflict.txt
Automatic merge failed; fix conflicts and then commit the result.
```

Executarea script-ului:

```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\home>cd Desktop
C:\Users\home\Desktop>cd MIDPS
C:\Users\home\Desktop\MIDPS>cd Laborator 2
C:\Users\home\Desktop\MIDPS\Laborator 2>cd Script
C:\Users\home\Desktop\MIDPS\Laborator 2\Script>run.bat
C:\Users\home\Desktop\MIDPS\Laborator 2\Script>gcc HelloWorld.c -o HelloWorld.exe
C:\Users\home\Desktop\MIDPS\Laborator 2\Script>HelloWorld
Hello, world !
```

Concluzii:

În lucrarea dată de laborator, ne-am familiarizat cu VCS (Version Control Systems) și am lucrat cu CLI. Am înțeles cum să creez un repository și să lucrez asupra lui în Git Bash de pe un remote server, cât și crearea branch-urilor pentru lucrul în subproiecte pentru proiectele mari. VCS este un instrument ce verifică versiunea proiectului, permitându-ne să evităm conflicte. El este util pentru lucrul cu proiecte mari unde participă un număr mare de oameni, fiecare lucrând asupra segmentului său.

Link spre repository: <https://github.com/olarucristian/midps/tree/master/Lab2>