Университет ИТМО

Факультет ФПИ и КТ

Лабораторная работа №2

По дисциплине

“Методы и средства программной инженерии”

Вариант 777666

Выполнили:

Барабанщиков А. Д.

Кочнев Р. Д.

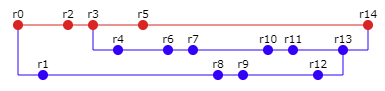
Группа: Р32081

Преподаватель:

Абузов Я. А.

Санкт-Петербург, 2023

## Текст задания



Сконфигурировать в своём домашнем каталоге репозитории svn и git и загрузить в них начальную ревизию файлов с исходными кодами (в соответствии с выданным вариантом).

Воспроизвести последовательность команд для систем контроля версий svn и git, осуществляющих операции над исходным кодом, приведённые на блок-схеме.

При составлении последовательности команд необходимо учитывать следующие условия:

Цвет элементов схемы указывает на пользователя, совершившего действие (красный - первый, синий - второй).

Цифры над узлами - номер ревизии. Ревизии создаются последовательно.

Необходимо разрешать конфликты между версиями, если они возникают.

## Выполнение (Git)

| #!/bin/bash  # Set Red user configuration git config user.name "Red" git config user.email "red@google.com"  # Clear directory rm -rf .git src  # Initialize git repository and create Red branch git init git checkout -b redbranch git add .gitignore  # Create src folder, copy and commit r0 from Red user mkdir src cp -f task\_commits/commit0/\* src/ git add . git commit -m "r0"  # Create Blue down branch, copy and commit r1 from Blue user git checkout -b bluedownbranch cp -f task\_commits/commit1/\* src/ git add . git commit --author="Blue <blue@google.com>" -m "r1"  # Switch to Red branch, copy and commit r2 and r3 from Red user git checkout redbranch cp -f task\_commits/commit2/\* src/ git add . git commit -m "r2" cp -f task\_commits/commit3/\* src/ git add . git commit -m "r3"  # Create Blue up branch, copy and commit r4 from Blue user git checkout -b blueupbranch cp -f task\_commits/commit4/\* src/ git add . git commit --author="Blue <blue@google.com>" -m "r4"  # Switch to Red branch, copy and commit r5 from Red user git checkout redbranch cp -f task\_commits/commit5/\* src/ git add . git commit -m "r5"  # Switch to Blue up branch, copy and commit r6 and r7 from Blue user git checkout blueupbranch cp -f task\_commits/commit6/\* src/ git add . git commit --author="Blue <blue@google.com>" -m "r6" cp -f task\_commits/commit7/\* src/ git add . git commit --author="Blue <blue@google.com>" -m "r7"  # Switch to Blue down branch, copy and commit r8 and r9 from Blue user git checkout bluedownbranch cp -f task\_commits/commit8/\* src/ git add . git commit --author="Blue <blue@google.com>" -m "r8" cp -f task\_commits/commit9/\* src/ git add . git commit --author="Blue <blue@google.com>" -m "r9"  # Switch to Blue up branch, copy and commit r10 and r11 from Blue user git checkout blueupbranch cp -f task\_commits/commit10/\* src/ git add . git commit --author="Blue <blue@google.com>" -m "r10" cp -f task\_commits/commit11/\* src/ git add . git commit --author="Blue <blue@google.com>" -m "r11"  # Switch to Blue down branch, copy and commit r12 from Blue user git checkout bluedownbranch cp -f task\_commits/commit12/\* src/ git add . git commit --author="Blue <blue@google.com>" -m "r12"  # Switch to Blue up branch and merge Blue down branch into it, creating r13 git checkout blueupbranch git merge bluedownbranch -m "Merging (r12 & r11) -> r13"  # Verify if there are merge conflicts if [[ $(git status --porcelain | grep "^UU") ]]; then  echo "Merge conflicts detected. Resolve them manually before proceeding."  while true; do  # Pause for the user to fix conflicts manually  read -p "Press 'y' to proceed after resolving conflicts, or 'n' to cancel: " user\_input  case $user\_input in  [Yy]\* )  # Proceed with merge after conflicts are resolved  git add .  git commit  break;;  [Nn]\* )  # Cancel merge if the user decides not to continue  git merge --abort  exit;;  \* ) echo "Please enter 'y' to proceed or 'n' to cancel.";;  esac  done fi  # Switch to Red branch and merge Blue up branch into it, creating r14 git checkout redbranch git merge blueupbranch -m "Merging (r13 & r5) -> r14"  # Verify if there are merge conflicts if [[ $(git status --porcelain | grep "^UU") ]]; then  echo "Merge conflicts detected. Resolve them manually before proceeding."  while true; do  # Pause for the user to fix conflicts manually  read -p "Press 'y' to proceed after resolving conflicts, or 'n' to cancel: " user\_input  case $user\_input in  [Yy]\* )  # Proceed with merge after conflicts are resolved  git add .  git commit  break;;  [Nn]\* )  # Cancel merge if the user decides not to continue  git merge --abort  exit;;  \* ) echo "Please enter 'y' to proceed or 'n' to cancel.";;  esac  done fi |
| --- |

## Выполнение (SVN)

| #!/bin/bash   function resolve\_tree\_conflicts() {  conflicted\_files=$(svn status | grep '^C' | awk '{print $2}')   # Loop through each conflicted file and resolve conflicts by keeping the working copy  for file in $conflicted\_files; do  svn resolve --accept working "$file"  done  # List all tree conflicted paths  tree\_conflicted\_paths=$(svn status | grep '^C' | awk '{print $2}')  svn resolve --accept working "$path" --depth infinity   # Loop through each tree conflicted path and resolve conflicts  for path in $tree\_conflicted\_paths; do  # Remove the obstructing working copy  rm -rf "$path"    # Use svn update to resolve the conflict  svn update --accept working "$path"  done }   # Create local repository svnadmin create repo REPO\_URL="file://$(pwd)/repo"  # Create the trunk and branches structure svn mkdir $REPO\_URL/trunk -m "Creating trunk" svn mkdir $REPO\_URL/branches -m "Creating branches"  # Create working copy svn co $REPO\_URL/trunk wc cd wc  # Create redbranch  svn copy $REPO\_URL/trunk $REPO\_URL/branches/redbranch -m "Creating redbranch" svn update svn switch $REPO\_URL/branches/redbranch  # Commit r0 mkdir src cp -rf ../task\_commits/commit0/\* src/ svn add ./src/. --force svn commit -m "r0"  # Create Blue down branch, copy and commit r1 from Blue user svn copy $REPO\_URL/trunk $REPO\_URL/branches/bluedownbranch -m "Creating bluedownbranch" svn update svn switch $REPO\_URL/branches/bluedownbranch sudo rm -rf /src mkdir src cp -rf ../task\_commits/commit1/\* src/ svn add ./src/. --force svn commit -m "r1" --username=Blue  # Switch to Red branch, copy and commit r2 and r3 from Red user svn switch $REPO\_URL/branches/redbranch rm -rf /src mkdir src cp -rf ../task\_commits/commit2/\* src/ svn add ./src/. svn commit -m "r2" rm -rf /src mkdir src cp -rf ../task\_commits/commit3/\* src/ svn add ./src/. --force svn commit -m "r3"  # Create Blue up branch, copy and commit r4 from Blue user svn copy $REPO\_URL/trunk $REPO\_URL/branches/blueupbranch -m "Creating blueupbranch" svn update svn switch $REPO\_URL/branches/blueupbranch rm -rf /src mkdir src cp -rf ../task\_commits/commit4/\* src/ svn add ./src/. --force svn commit -m "r4" --username=Blue  # Switch to Red branch, copy and commit r5 from Red user svn switch $REPO\_URL/branches/redbranch rm -rf /src mkdir src cp -rf ../task\_commits/commit5/\* src/ svn add ./src/. --force svn commit -m "r5"  # Switch to Blue up branch, copy and commit r6 and r7 from Blue user svn switch $REPO\_URL/branches/blueupbranch rm -rf /src mkdir src cp -rf ../task\_commits/commit6/\* src/ svn add ./src/. --force  svn commit -m "r6" --username=Blue rm -rf /src mkdir src cp -rf ../task\_commits/commit7/\* src/ svn add ./src/. --force svn commit -m "r7" --username=Blue  # Switch to Blue down branch, copy and commit r8 and r9 from Blue user svn switch $REPO\_URL/branches/bluedownbranch rm -rf /src mkdir src cp -rf ../task\_commits/commit8/\* src/ svn add ./src/. --force svn commit -m "r8" --username=Blue rm -rf /src mkdir src cp -rf ../task\_commits/commit9/\* src/ svn add ./src/. --force svn commit -m "r9" --username=Blue  # Switch to Blue up branch, copy and commit r10 and r11 from Blue user svn switch $REPO\_URL/branches/blueupbranch rm -rf /src mkdir src cp -rf ../task\_commits/commit10/\* src/ svn add ./src/. --force svn commit -m "r10" --username=Blue rm -rf /src mkdir src cp -rf ../task\_commits/commit11/\* src/ svn add ./src/. --force svn commit -m "r11" --username=Blue  # Switch to Blue down branch, copy and commit r12 from Blue user svn switch $REPO\_URL/branches/bluedownbranch rm -rf /src mkdir src cp -rf ../task\_commits/commit12/\* src/ svn add ./src/. --force  svn commit -m "r12" --username=Blue  # Switch to Blue up branch and merge Blue down branch into it, creating r13 svn switch $REPO\_URL/branches/blueupbranch svn update svn merge $REPO\_URL/branches/bluedownbranch --accept=postpone  resolve\_tree\_conflicts svn commit -m "Merged bluedownbranch into blueupbranch (r13)"   svn switch $REPO\_URL/branches/redbranch svn update svn merge $REPO\_URL/branches/blueupbranch --accept=postpone  resolve\_tree\_conflicts svn commit -m "Merged blueupbranch into redbranch (r14)" |
| --- |

## Вывод

В ходе выполнения данной лабораторной работы мы познакомились с системами контроля версий git и svn. Научились применять на практике различный команды этих VCS, работать с пользователями, ветками и разрешать конфликты.