Университет ИТМО
Факультет программной инженерии и компьютерной техники

Методы и средства программной инженерии. Лабораторная работа №2. Системы контроля версий

Смирнов Виктор Игоревич P32131
Шиняков Артём Дмитриевич R32372
Вариант 1009

2023

Содержание

1 Задание

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:

1. Цвет элементов схемы указывает на пользователя, совершившего действие (красный - первый, синий - второй).
2. Цифры над узлами - номер ревизии. Ревизии создаются последовательно.
3. Необходимо разрешать конфликты между версиями, если они возникают.

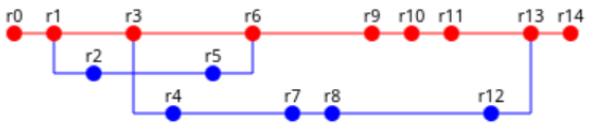

Рис. 1: История ревизий репозитория

2 Взаимодействие с репозиторием через SVN

```
#!/bin/bash

. ci/svn/lib/head.sh --source-only
TARGET="main"
. ci/svn/lib/dsl.sh --source-only

begin
  log "pwd: $(pwd)"

  call clean
  call r_init
  call r0
  call r1
  call r2
  call r3
  call r4
  call r5
  call r6
  call r7
  call r8
  call r9
  call r10
  call r11
  call r12
  call r13
  call r14

  svn checkout file://$HOME/.svnrepos/$REPO_NAME out
  cd out
```

```
  svn log > $VSC_NAME-log.txt
  cd ..
end
```

```
#!/bin/bash

. ci/svn/lib/head.sh --source-only
TARGET="init"
. ci/svn/lib/dsl.sh --source-only

begin
  log "repo is $REPO_NAME"

  mkdir -p ~/.svnrepos/
  svnadmin create ~/.svnrepos/$REPO_NAME

  log "repository created"

  svn mkdir -m "Create repository structure." \
    file://$HOME/.svnrepos/$REPO_NAME/trunk \
    file://$HOME/.svnrepos/$REPO_NAME/branches \
    file://$HOME/.svnrepos/$REPO_NAME/tags

  log "repository initialized"

  mkdir -p playground/$REPO_NAME
  cd playground/$REPO_NAME

  svn checkout file://$HOME/.svnrepos/$REPO_NAME/trunk trunk
  cd trunk

  svn add --force .
  svn commit -m "Initial import."
  svn update
end
```

```
#!/bin/bash

. ci/svn/lib/head.sh --source-only
BRANCH="trunk"
COMMIT="commit0"
TARGET="$BRANCH:$COMMIT"
. ci/svn/lib/dsl.sh --source-only

begin
  enter

  edit A.java
  edit B.java
  edit E.java
  edit F.java

  svn add *
  svn commit -m "$TAG added A, B, E, F"
end
```

```
#!/bin/bash

. ci/svn/lib/head.sh --source-only
BRANCH="trunk"
COMMIT="commit1"
TARGET="$BRANCH:$COMMIT"
. ci/svn/lib/dsl.sh --source-only

begin
  enter

  edit A.java
  edit B.java
  edit E.java
  edit F.java

  svn commit -m "$TAG edited A, B, E, F"
```

```
end
```

```
#!/bin/bash

. ci/svn/lib/head.sh --source-only
BRANCH="feature-1"
COMMIT="commit2"
TARGET="$BRANCH:$COMMIT"
. ci/svn/lib/dsl.sh --source-only

begin
  branch trunk "$BRANCH" "Creating a branch for a feature #1"

  enter

  edit A.java
  edit B.java
  edit E.java
  edit F.java

  svn commit -m "$TAG edited A, B, E, F"
  log "commited changes to $BRANCH"
end
```

```
#!/bin/bash

. ci/svn/lib/head.sh --source-only
BRANCH="trunk"
COMMIT="commit3"
TARGET="$BRANCH:$COMMIT"
. ci/svn/lib/dsl.sh --source-only

begin
  enter

  edit "*"
  edit "3yNy8wQeGi.Xzj"
  edit A.java
  edit B.java
  edit E.java
  edit F.java

  svn add "*"
  svn add "3yNy8wQeGi.Xzj"

  svn commit -m "$TAG edited A, B, E, F, added *, 3yNy8wQeGi.Xzj"
  log "commited changes to $BRANCH"
end
```

```
#!/bin/bash

. ci/svn/lib/head.sh --source-only
BRANCH="feature-2"
COMMIT="commit4"
TARGET="$BRANCH:$COMMIT"
. ci/svn/lib/dsl.sh --source-only

begin
  branch "trunk" $BRANCH  "Creating a branch for a feature #2"

  enter

  delete "*"
  delete "3yNy8wQeGi.Xzj"
  edit A.java
  edit B.java
  edit E.java
  edit F.java

  svn delete "*"
  svn delete "3yNy8wQeGi.Xzj"

  svn commit -m "$TAG edited A, B, E, F, removed *, 3yNy8wQeGi.Xzj"
```

```
  log "commited changes to $BRANCH"
end
```

```
#!/bin/bash

. ci/svn/lib/head.sh --source-only
BRANCH="feature-1"
COMMIT="commit5"
TARGET="$BRANCH:$COMMIT"
. ci/svn/lib/dsl.sh --source-only

begin
  enter

  edit A.java
  edit B.java
  edit E.java
  edit F.java

  svn commit -m "$TAG edited A, B, E, F"
  log "commited changes to $BRANCH"
end
```

```
#!/bin/bash

. ci/svn/lib/head.sh --source-only
BRANCH="trunk"
COMMIT="commit6"
TARGET="$BRANCH:$COMMIT"
. ci/svn/lib/dsl.sh --source-only

begin
  enter

  svn merge ^/branches/feature-1
  log "merged feature-1 into trunk"

  delete "3yNy8wQeGi.Xzj"
  edit A.java
  edit B.java
  edit E.java
  edit F.java

  svn remove "3yNy8wQeGi.Xzj"

  svn commit -m "merged feature-1, $TAG edited A, B, E, F, removed 3yNy8wQeGi.Xzj"
  log "commited changes to $BRANCH"
end
```

```
#!/bin/bash

. ci/svn/lib/head.sh --source-only
BRANCH="feature-2"
COMMIT="commit7"
TARGET="$BRANCH:$COMMIT"
. ci/svn/lib/dsl.sh --source-only

begin
  enter

  edit A.java
  edit B.java
  edit E.java
  edit F.java

  svn commit -m "$TAG edited A, B, E, F"
  log "commited changes to $BRANCH"
end
```

```
#!/bin/bash

. ci/svn/lib/head.sh --source-only
BRANCH="feature-2"
```

```
COMMIT="commit8"
TARGET="$BRANCH:$COMMIT"
. ci/svn/lib/dsl.sh --source-only

begin
  enter

  edit "*"
  edit "3yNy8wQeGi.Xzj"
  edit A.java
  edit B.java
  edit E.java
  edit F.java

  svn add "*"
  svn add "3yNy8wQeGi.Xzj"

  svn commit -m "$TAG edited A, B, E, F, added *, 3yNy8wQeGi.Xzj"
end
```

```
#!/bin/bash

. ci/svn/lib/head.sh --source-only
BRANCH="trunk"
COMMIT="commit9"
TARGET="$BRANCH:$COMMIT"
. ci/svn/lib/dsl.sh --source-only

begin
  enter

  delete "*"
  edit A.java
  edit B.java
  edit E.java
  edit F.java

  svn remove "*"

  svn commit -m "$TAG edited A, B, E, F, removed *"
  log "commited changes to $BRANCH"
end
```

```
#!/bin/bash

. ci/svn/lib/head.sh --source-only
BRANCH="trunk"
COMMIT="commit10"
TARGET="$BRANCH:$COMMIT"
. ci/svn/lib/dsl.sh --source-only

begin
  enter

  edit "*"
  edit "67VNlR0FbP.TcV"
  edit A.java
  edit B.java
  edit E.java
  edit F.java

  svn add "*"
  svn add "67VNlR0FbP.TcV"

  svn commit -m "$TAG edited A, B, E, F, restored *, added 67VNlR0FbP.TcV"
  log "commited changes to $BRANCH"
end
```

```
#!/bin/bash

. ci/svn/lib/head.sh --source-only
BRANCH="trunk"
COMMIT="commit11"
```

```
TARGET="$BRANCH:$COMMIT"
. ci/svn/lib/dsl.sh --source-only

begin
  enter

  delete "67VNlR0FbP.TcV"
  edit   "*"
  edit   A.java
  edit   B.java
  edit   E.java
  edit   F.java

  svn remove "67VNlR0FbP.TcV"

  svn commit -m "$TAG edited A, B, E, F, *, removed 67VNlR0FbP.TcV"
  log "commited changes to $BRANCH"
end
```

```
#!/bin/bash

. ci/svn/lib/head.sh --source-only
BRANCH="feature-2"
COMMIT="commit12"
TARGET="$BRANCH:$COMMIT"
. ci/svn/lib/dsl.sh --source-only

begin
  enter

  delete "*"
  delete "3yNy8wQeGi.Xzj"
  edit A.java
  edit B.java
  edit E.java
  edit F.java

  svn remove "*"
  svn remove "3yNy8wQeGi.Xzj"

  svn commit -m "$TAG edited A, B, E, F, removed *, 3yNy8wQeGi.Xzj"
  log "commited changes to $BRANCH"
end
```

```
#!/bin/bash

. ci/svn/lib/head.sh --source-only
BRANCH="trunk"
COMMIT="commit13"
TARGET="$BRANCH:$COMMIT"
. ci/svn/lib/dsl.sh --source-only

begin
  enter

  svn merge ^/branches/feature-2
  log "merged feature-2 into trunk"

  edit A.java
  edit B.java
  edit E.java
  edit F.java

  svn commit -m "$TAG edited A, B, E, F, merged feature-2 into $BRANCH"
  log "commited changes to $BRANCH"
end
```

```
#!/bin/bash

. ci/svn/lib/head.sh --source-only
BRANCH="trunk"
COMMIT="commit14"
TARGET="$BRANCH:$COMMIT"
```

```
. ci/svn/lib/dsl.sh --source-only

begin
  enter

  edit "rvvddKJVqH.1iP"
  edit A.java
  edit B.java
  edit E.java
  edit F.java

  svn add "rvvddKJVqH.1iP"

  svn commit -m "$TAG edited A, B, E, F, added rvvddKJVqH.1iP"
  log "commited changes to $BRANCH"
end
```

```
set -e

cd $(dirname -- "$0"; )
cd ../..

VSC_NAME="svn"
REPO_NAME="semt-assigment-vcs-$VSC_NAME-repository"
SCRIPT="ci/svn"
```

```
TAG="[$VSC_NAME:$TARGET]"

log() {
  echo "$TAG $1"
}

remove() {
  rm -rf $1
  log "removed $1"
}

copy() {
  cp "$1" "$2"
  log "copied $1 to $2"
}

call() {
  bash "$SCRIPT/$1.sh"
}

begin() {
  log "started $TARGET"
}

end() {
  log "finished $TARGET"
}

enter() {
  SRC=../../../history/$COMMIT
  cd playground/$REPO_NAME/$BRANCH

  USERNAME="blue"
  if [[ $BRANCH = "trunk" ]]; then
    USERNAME="red"
  fi
  svn update --username $USERNAME
}

edit() {
  copy "$SRC/$1" "$1"
}

delete() {
  remove $1
}

```

```
branch() {
  # $1 - source branch name, e.g. "trunk" or "branches/my-branch"
  # $2 - target branch name, e.g. "my-branch"
  # $3 - message

  svn copy \
    file://$HOME/.svnrepos/$REPO_NAME/$1 \
    file://$HOME/.svnrepos/$REPO_NAME/branches/$2 \
    -m "$3"
  log $3

  cd playground/$REPO_NAME
  svn checkout \
    file://$HOME/.svnrepos/$REPO_NAME/branches/$2 $2
  log "checkout to branch $2"
  cd ../..
}
```

3 Взаимодействие с репозиторием через GIT

```
#!/bin/bash

. ci/git-r/lib/head.sh --source-only
TARGET="main"
. ci/git-r/lib/dsl.sh --source-only

begin
  call clean
  call init
  call init_users
  call r0
  call r1
  call r2
  call r3
  call r4
  call r5
  call r6
  call r7
  call r8
  call r9
  call r10
  call r11
  call r12
  call r13
  call r14

  cd ~/.gitrepo/$REPO_NAME
  mkdir logs
  cd logs
  git log > $VSC_NAME-log.txt
  git log --pretty=format:"%h %s" --graph > $VSC_NAME-graph.txt
end
```

```
#!/bin/bash

. ci/git-r/lib/head.sh --source-only
TARGET="init"
. ci/git-r/lib/dsl.sh --source-only

begin
  log "repo is $REPO_NAME"

  mkdir -p ~/.gitrepo
  cd ~/.gitrepo
  git init $REPO_NAME --bare

  git config --global pull.rebase false
  log "repository created"

end
```

```
#!/bin/bash
```

```

. ci/git-r/lib/head.sh --source-only
BRANCH="master"
COMMIT="commit0"
TARGET="$BRANCH:$COMMIT"
NAME=$ARTEM
EMAIL="Artem@itmo.ru"
. ci/git-r/lib/dsl.sh --source-only

begin
  enter

  set_name $NAME
  set_email $EMAIL

  cp $HISTORY_PATH/$COMMIT/* .
  add_all

  comm "Start of project. Added initial files."

  git push origin
end
```

```
#!/bin/bash

. ci/git-r/lib/head.sh --source-only
BRANCH="master"
COMMIT="commit1"
TARGET="$BRANCH:$COMMIT"
NAME=$ARTEM
EMAIL="Artem@itmo.ru"
. ci/git-r/lib/dsl.sh --source-only

begin
  enter

  git pull origin

  cp $HISTORY_PATH/$COMMIT/* .
  add_all

  comm "Added: bb - print class name in F.java."

  git push origin
end
```

```
#!/bin/bash

. ci/git-r/lib/head.sh --source-only
BRANCH="second_branch"
COMMIT="commit2"
TARGET="$BRANCH:$COMMIT"
NAME=$VITYA
EMAIL="Vitya@itmo.ru"
. ci/git-r/lib/dsl.sh --source-only

begin
  enter

  set_name $NAME
  set_email $EMAIL

  git pull origin

  git checkout -b second_branch

  cp $HISTORY_PATH/$COMMIT/* .
  add_all

  comm "Created second branch, files are in the same state as in r0 commit."

  git push origin second_branch
end
```

```
#!/bin/bash

. ci/git-r/lib/head.sh --source-only
BRANCH="master"
COMMIT="commit3"
TARGET="$BRANCH:$COMMIT"
NAME=$ARTEM
EMAIL="Artem@itmo.ru"
. ci/git-r/lib/dsl.sh --source-only

begin
  enter

  git pull origin

  cp $HISTORY_PATH/$COMMIT/* .
  add_all

  comm "Added: pp fuction - returns Object in F class, * file - contains chinese, 3
    yNy8wQeGi.Xzj file - contains binary something "

  git push origin
end
```

```
#!/bin/bash

. ci/git-r/lib/head.sh --source-only
BRANCH="third_branch"
COMMIT="commit4"
TARGET="$BRANCH:$COMMIT"
NAME=$VITYA
EMAIL="Vitya@itmo.ru"
. ci/git-r/lib/dsl.sh --source-only

begin
  enter

  git checkout master

  git pull origin

  git checkout -b third_branch

  cp $HISTORY_PATH/$COMMIT/* .

  git rm "*" -f
  git rm 3yNy8wQeGi.Xzj -f

  add_all

  comm "Created third branch, files are in the same state as in r0 commit., files * and
    3yNy8wQeGi.Xzj were removed"

  git push origin third_branch
end
```

```
#!/bin/bash

. ci/git-r/lib/head.sh --source-only
BRANCH="second_branch"
COMMIT="commit5"
TARGET="$BRANCH:$COMMIT"
NAME=$VITYA
EMAIL="Vitya@itmo.ru"
. ci/git-r/lib/dsl.sh --source-only

begin
  enter

  git pull origin master

  git checkout second_branch

```

```
  cp $HISTORY_PATH/$COMMIT/* .
  add_all

  comm "Added: bb - print class name in F.java."

  git push origin second_branch
end
```

```
#!/bin/bash

. ci/git-r/lib/head.sh --source-only
BRANCH="master"
COMMIT="commit6"
TARGET="$BRANCH:$COMMIT"
NAME=$ARTEM
EMAIL="Artem@itmo.ru"
. ci/git-r/lib/dsl.sh --source-only

begin
  enter

  git pull origin

  git checkout second_branch

  git checkout third_branch

  git checkout master

  git merge second_branch -m "Second branch does not contain any new features"
  cp $HISTORY_PATH/$COMMIT/* .
  add_all

  git rm 3yNy8wQeGi.Xzj -f

  comm "Added: pp fuction - returns Object in F class, * file - contains chinese, 3
    yNy8wQeGi.Xzj file - contains binary something, interfaces A,B,E turned into classes
    "

  git push origin
end
```

```
#!/bin/bash

. ci/git-r/lib/head.sh --source-only
BRANCH="third-branch"
COMMIT="commit7"
TARGET="$BRANCH:$COMMIT"
NAME=$VITYA
EMAIL="Vitya@itmo.ru"
. ci/git-r/lib/dsl.sh --source-only

begin
  enter

  git pull origin master

  git checkout third_branch

  cp $HISTORY_PATH/$COMMIT/* .
  add_all

  comm "Added: bb function - returns Object in F.java"

  git push origin third_branch
end
```

```
#!/bin/bash

. ci/git-r/lib/head.sh --source-only
BRANCH="third-branch"
COMMIT="commit8"
TARGET="$BRANCH:$COMMIT"
```

```
NAME=$VITYA
EMAIL="Vitya@itmo.ru"
. ci/git-r/lib/dsl.sh --source-only

begin
  enter

  git pull origin third_branch

  cp $HISTORY_PATH/$COMMIT/* .
  add_all

  comm "Added: pp fuction - returns Object in F class, * file - contains chinese, 3
    yNy8wQeGi.Xzj file - contains binary something"

  git push origin third_branch
end
```

```
#!/bin/bash

. ci/git-r/lib/head.sh --source-only
BRANCH="master"
COMMIT="commit9"
TARGET="$BRANCH:$COMMIT"
NAME=$ARTEM
EMAIL="Artem@itmo.ru"
. ci/git-r/lib/dsl.sh --source-only

begin
  enter

  git pull origin

  git checkout master

  cp $HISTORY_PATH/$COMMIT/* .
  add_all

  git rm "*" -f

  comm "Added: nn fuction - returns Object in F class, * file - was removed, classes A,
    B, E got one new method each "

  git push origin
end
```

```
#!/bin/bash

. ci/git-r/lib/head.sh --source-only
BRANCH="master"
COMMIT="commit10"
TARGET="$BRANCH:$COMMIT"
NAME="Artem"
EMAIL="Artem@itmo.ru"
. ci/git-r/lib/dsl.sh --source-only

begin
  enter

  git pull origin

  cp $HISTORY_PATH/$COMMIT/* .
  add_all

  comm "Added: A, B, E, F classes got one new method each, * and 67VNIR0FbP.TcV files
    were created."

  git push origin
end
```

```
#!/bin/bash

. ci/git-r/lib/head.sh --source-only
```

```
BRANCH="master"
COMMIT="commit11"
TARGET="$BRANCH:$COMMIT"
NAME=$ARTEM
EMAIL="Artem@itmo.ru"
. ci/git-r/lib/dsl.sh --source-only

begin
  enter

  git pull origin

  cp $HISTORY_PATH/$COMMIT/* .
  add_all

  git rm 67VNlROFbP.TcV -f

  comm "Added: A, B, E, F classes got one new method each, 67VNlROFbP.TcV file - removed
    ."

  git push origin
end
```

```
#!/bin/bash

. ci/git-r/lib/head.sh --source-only
BRANCH="third-branch"
COMMIT="commit12"
TARGET="$BRANCH:$COMMIT"
NAME=$VITYA
EMAIL="Vitya@itmo.ru"
. ci/git-r/lib/dsl.sh --source-only

begin
  enter

  git checkout third_branch

  cp $HISTORY_PATH/$COMMIT/* .
  add_all

  git rm "*" -f
  git rm 3yNy8wQeGi.Xzj -f

  comm "Added: mm function - returns Object in F.java, * and 3yNy8wQeGi.Xzj files were
    removed."

  git push origin third_branch
end
```

```
#!/bin/bash

. ci/git-r/lib/head.sh --source-only
BRANCH="master"
COMMIT="commit13"
TARGET="$BRANCH:$COMMIT"
NAME=$ARTEM
EMAIL="Artem@itmo.ru"
. ci/git-r/lib/dsl.sh --source-only

begin
  enter

  git pull origin

  git checkout third_branch

  git checkout master

  {
    git merge third_branch
  }||{
    git checkout second_branch F.java
```

```
    git rm "*" -f
  }

  cp $HISTORY_PATH/$COMMIT/* .

  add_all

  comm "Merged master and third branch. Added new functions in B, E, F"

  git push origin

end
```

```
#!/bin/bash

. ci/git-r/lib/head.sh --source-only
BRANCH="master"
COMMIT="commit14"
TARGET="$BRANCH:$COMMIT"
NAME=$ARTEM
EMAIL="Artem@itmo.ru"
. ci/git-r/lib/dsl.sh --source-only

begin
  enter

  git pull origin

  cp $HISTORY_PATH/$COMMIT/* .
  add_all

  comm "Added: A, B, E, F classes got one new method each, rvvddKJVqH.1ip  file - added.
    "

  git push origin
end
```

```
set -e

cd $(dirname -- "$0"; )
cd ../..
HISTORY_PATH=$(pwd)/history
USERS_REPO=~/.user_repo
ARTEM="Artem"
VITYA="Vitya"

VSC_NAME="git-r"
REPO_NAME="semt-assigment-vcs-$VSC_NAME-repository"
SCRIPT="ci/git-r"
URL=git+ssh://s337054@se.ifmo.ru:2222/home/studs/s337054/srv/git/project.git
```

```
TAG="[$VSC_NAME:$TARGET]"

log() {
  echo "$TAG $1"
}

remove() {
  rm -rf $1
  log "removed $1"
}

add_all(){
    git add .
    log "added files from $TAG"
}

call() {
  bash "$SCRIPT/$1.sh"
}

begin() {
  log "started $TARGET"
```

```
}

end() {
  log "finished $TARGET"
}

set_name() {
  git config --local user.name "$1"
}

set_email() {
  git config --local user.email "$2"
}

enter() {
  cd $USERS_REPO/$NAME
}

comm() {
    git commit -m "$1"
}
```

4 Вывод

Выполнив данную лабораторную работу мы научились использовать базовые функции таких известных систем контроля версий как git и svn. Оба пакета программного обеспечения предоставляют весь необходимый функционал для удобного использования, но управляются пользователем по-разному, что делает одну СКВ предпочтительнее другой в зависимости от сложившейся ситуации. Например, интерфейс системы контроля версий git немного проще svn и предлагает более лаконичную (не факт) схему работы с репозиторием. А svn в свою очередь может предложить очень удобный механизм частичного монтажа поддиректорий репозитория, такая функциональность может быть полезна, когда мы имеем дело с монорепозиториями и не хотим видеть сразу все содержимое большого проекта.