<epam>

VCS concept

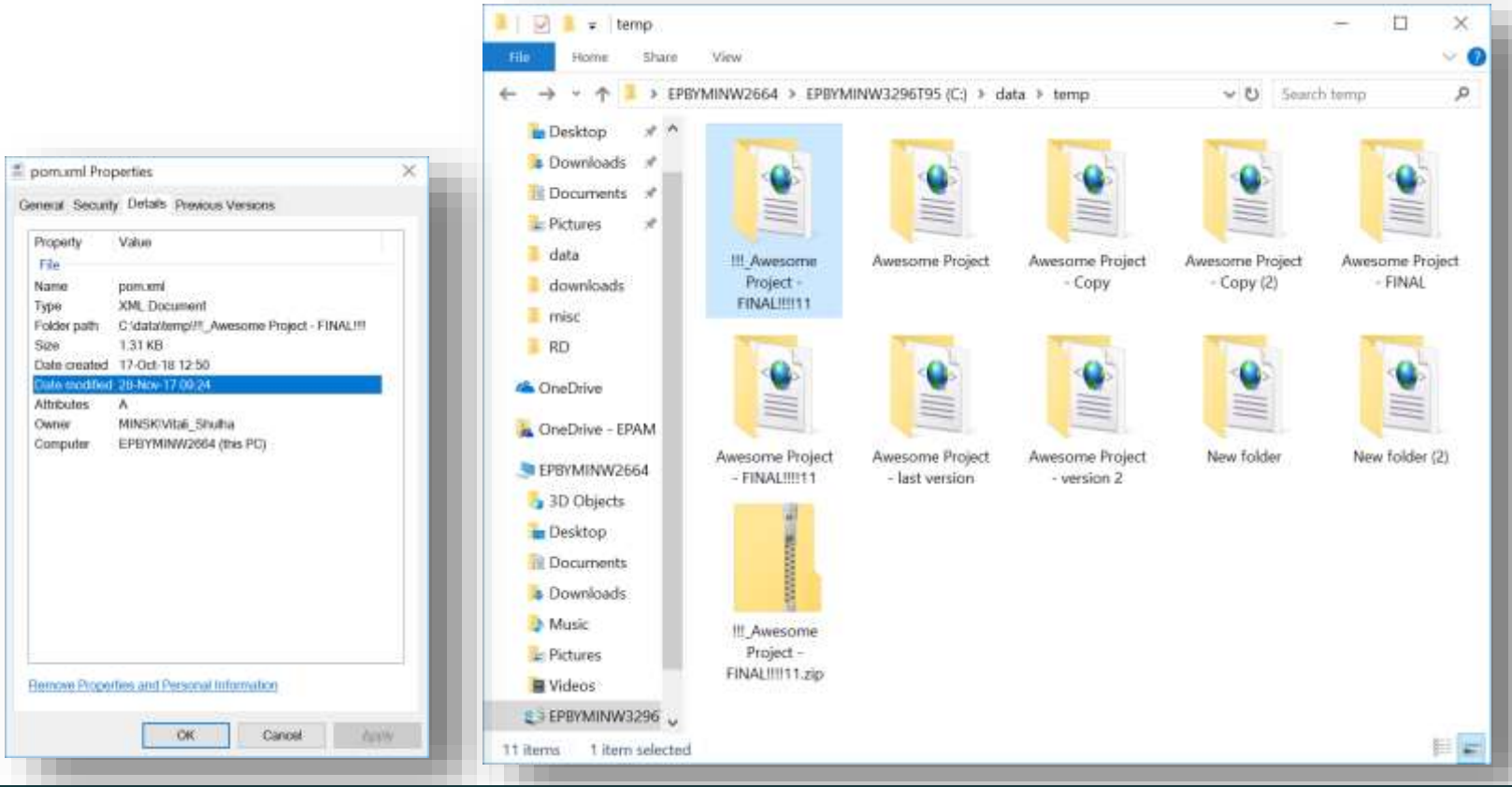
Version Control with Git. DevTestOps training.

TRAINING

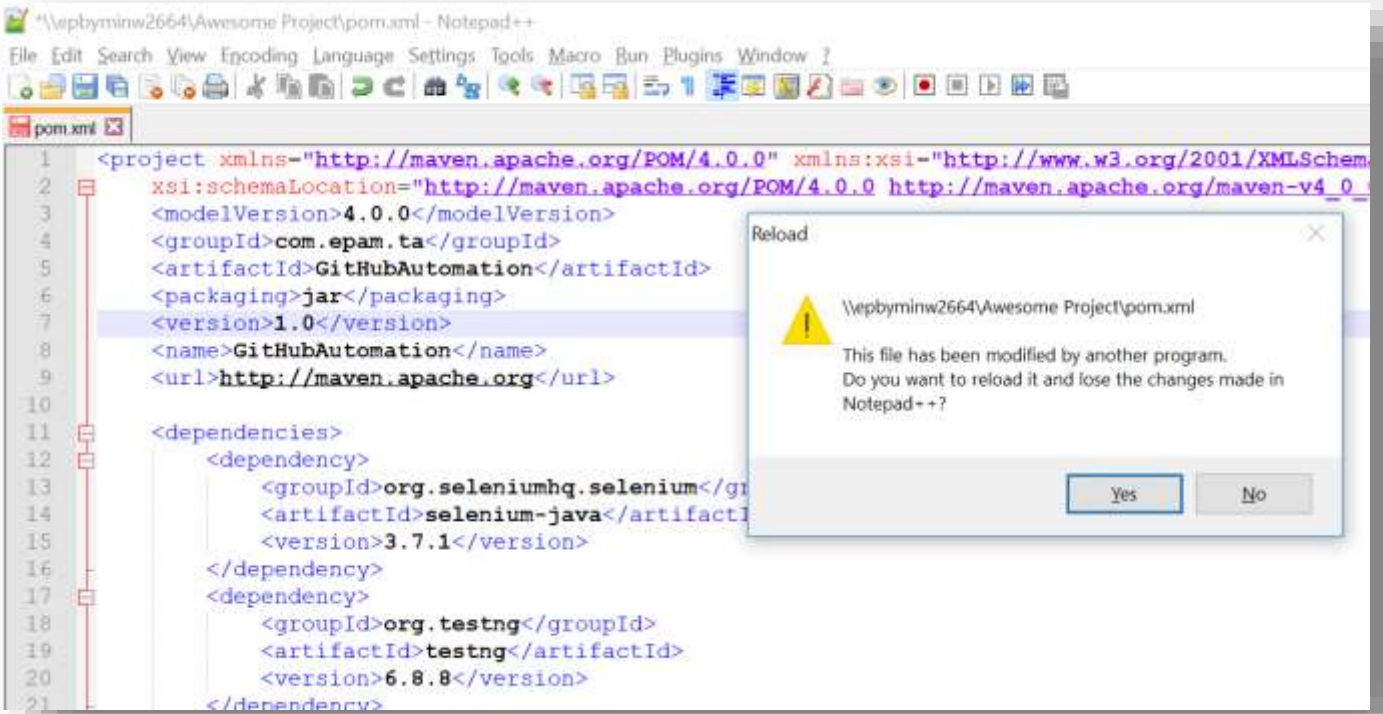
CENTER

<epam>

Standalone work. Level 1 - beginner



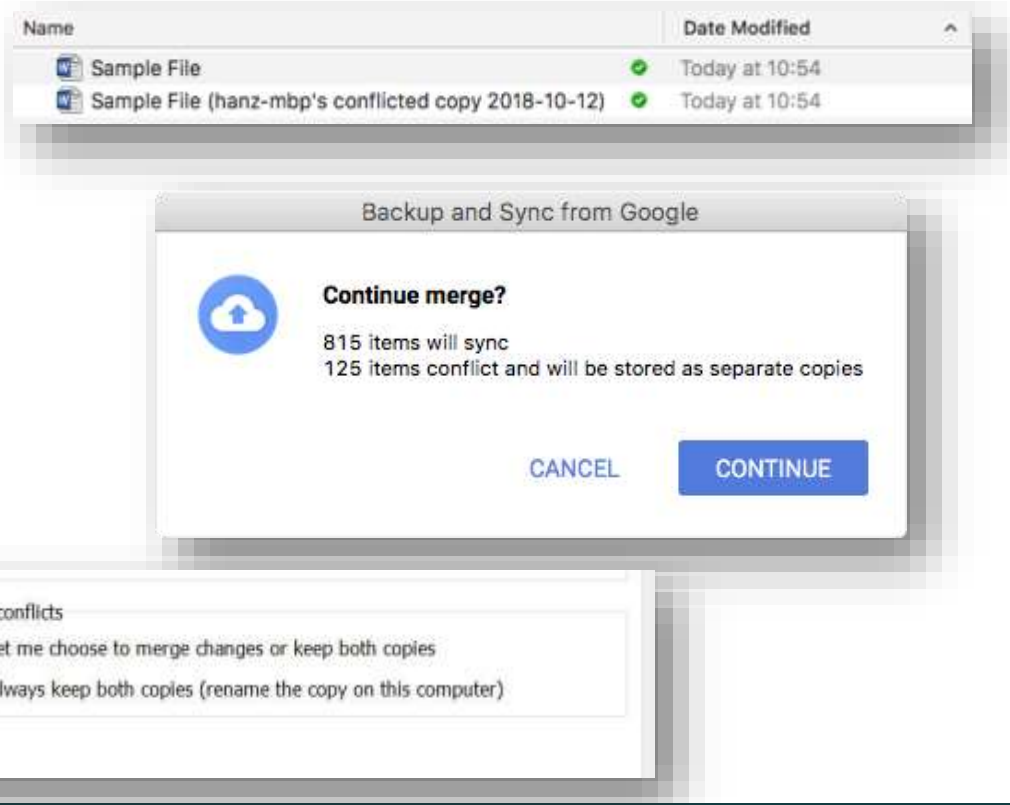
Team work. Level 2 – network share



Standalone/Team work. Level 3 - cloud

Dropbox

OneDrive



VCS goals

1. BACKUP AND RESTORE
2. SYNCHRONIZATION
3. UNDO
4. TRACK CHANGES AND OWNERSHIP
5. SANDBOXING
6. BRANCHING

<epam>

Version control types

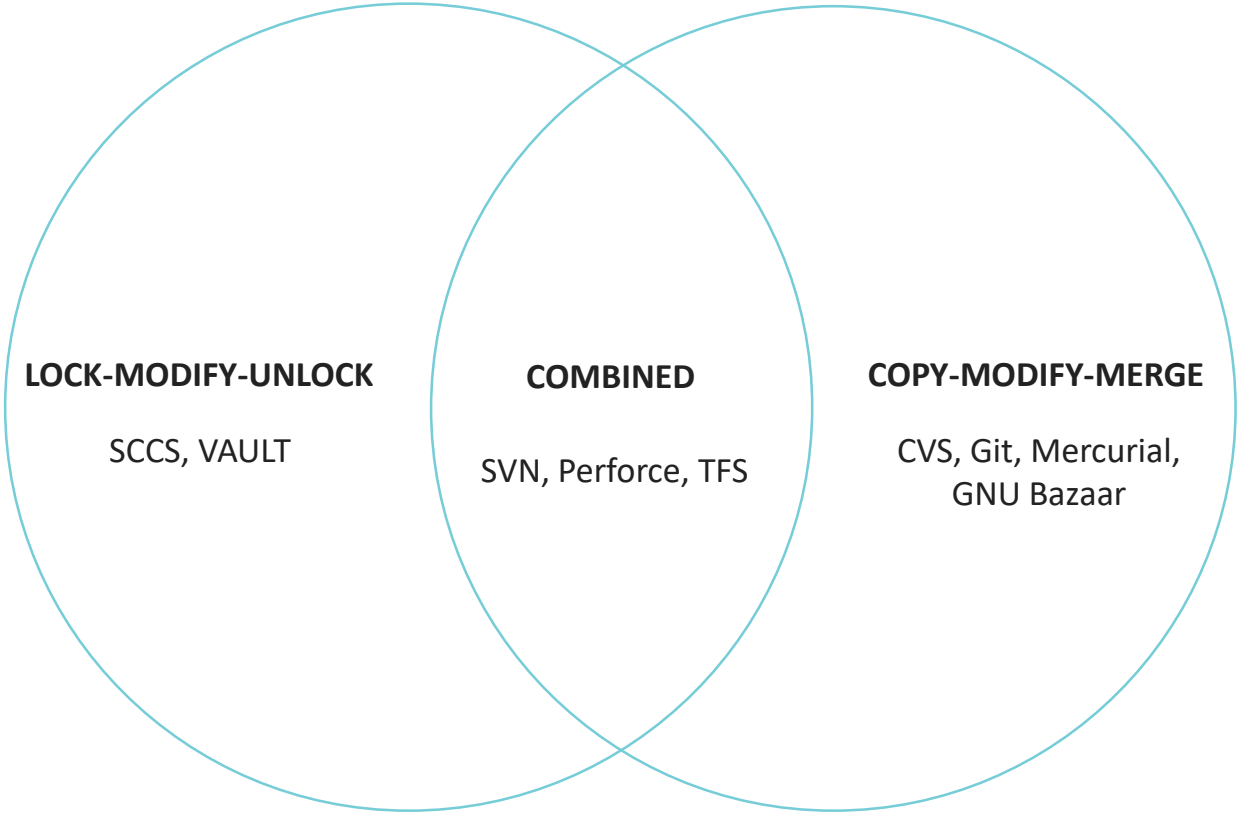
Version Control with Git. DevTestOps training.

TRAINING

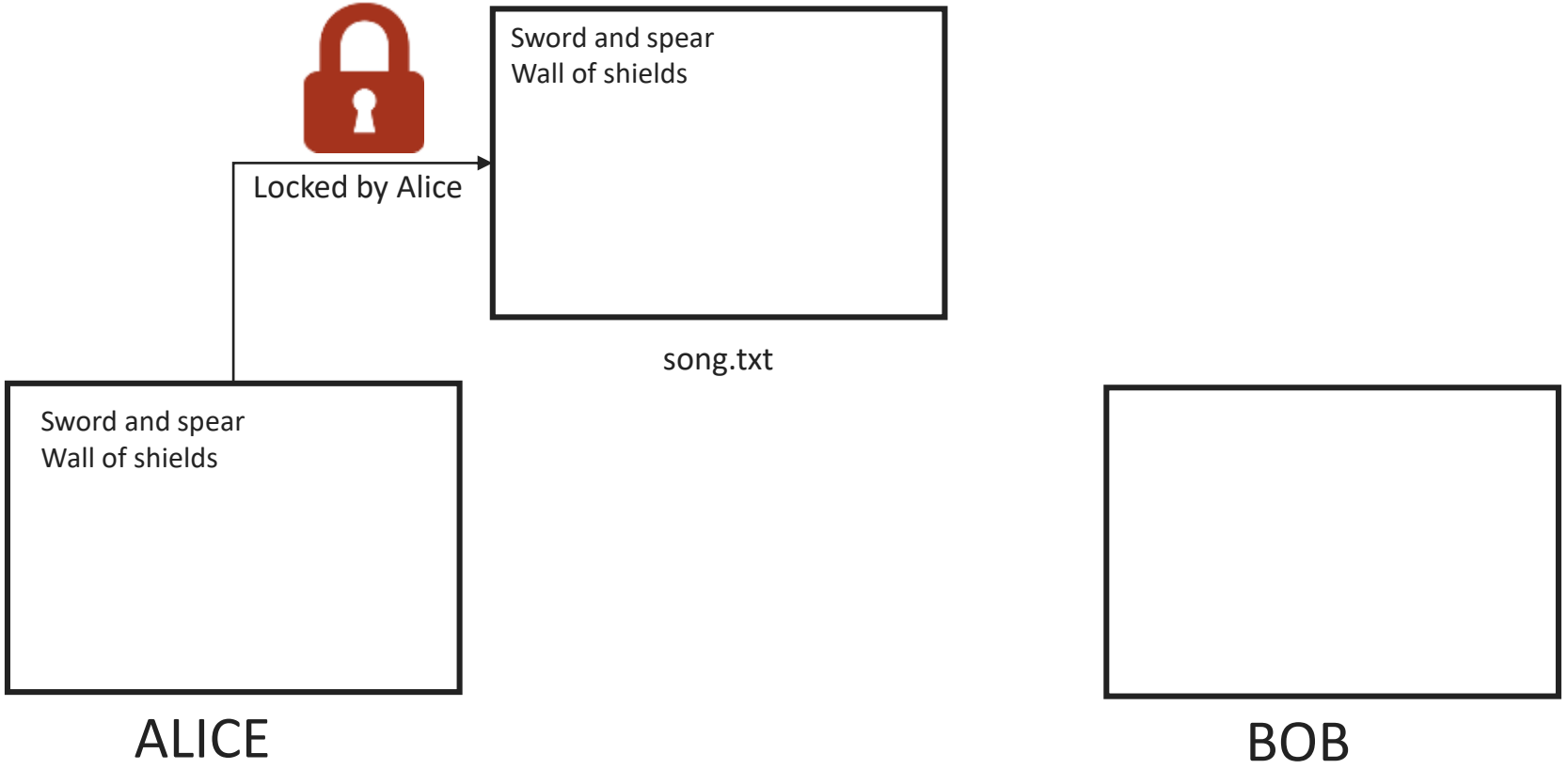
CENTER

<epam>

VCS types


LOCK-MODIFY-UNLOCK
SCCS, VAULT
COMBINED
SVN, Perforce, TFS
COPY-MODIFY-MERGE
CVS, Git, Mercurial,
GNU Bazaar

Lock-modify-unlock strategy



Lock-modify-unlock strategy

Sword and spear
Wall of shields

song.txt

Bob updates his local copy

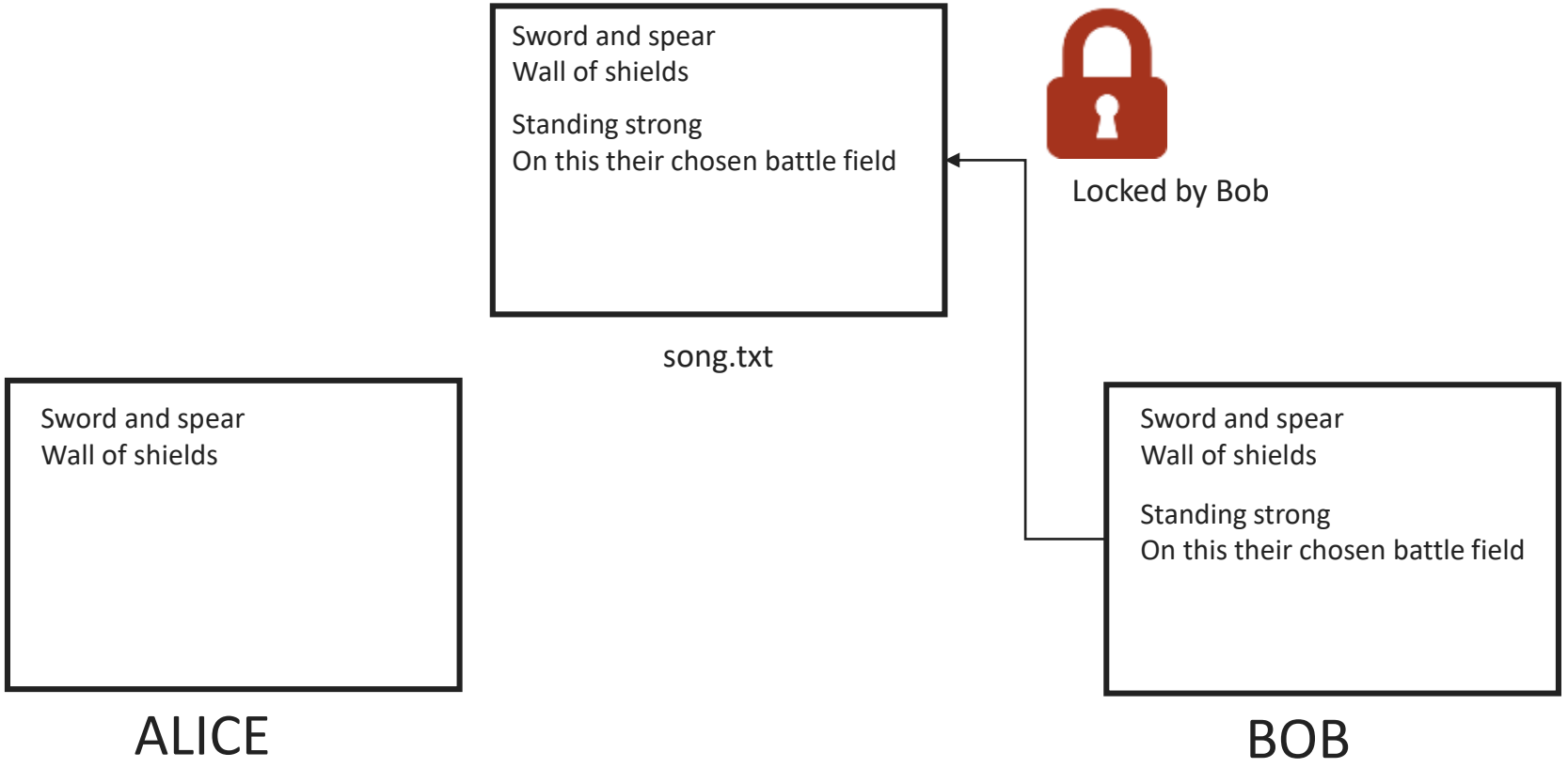
Sword and spear
Wall of shields

ALICE

Sword and spear
Wall of shields

BOB

Lock-modify-unlock strategy



Lock-modify-unlock strategy

Sword and spear
Wall of shields

Standing strong
On this their chosen battle field

song.txt

Unlocked by Bob

Sword and spear
Wall of shields

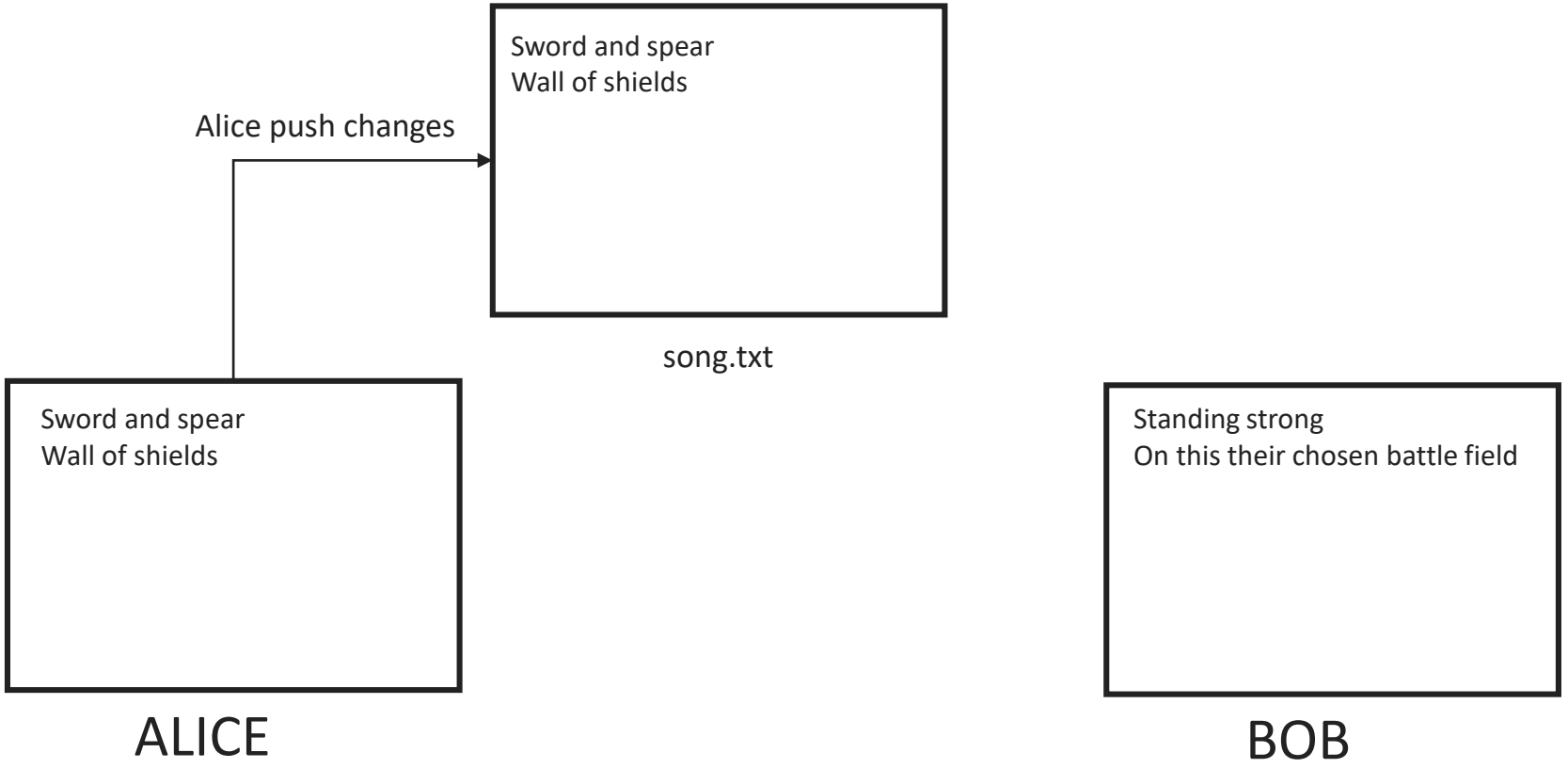
ALICE

Sword and spear
Wall of shields

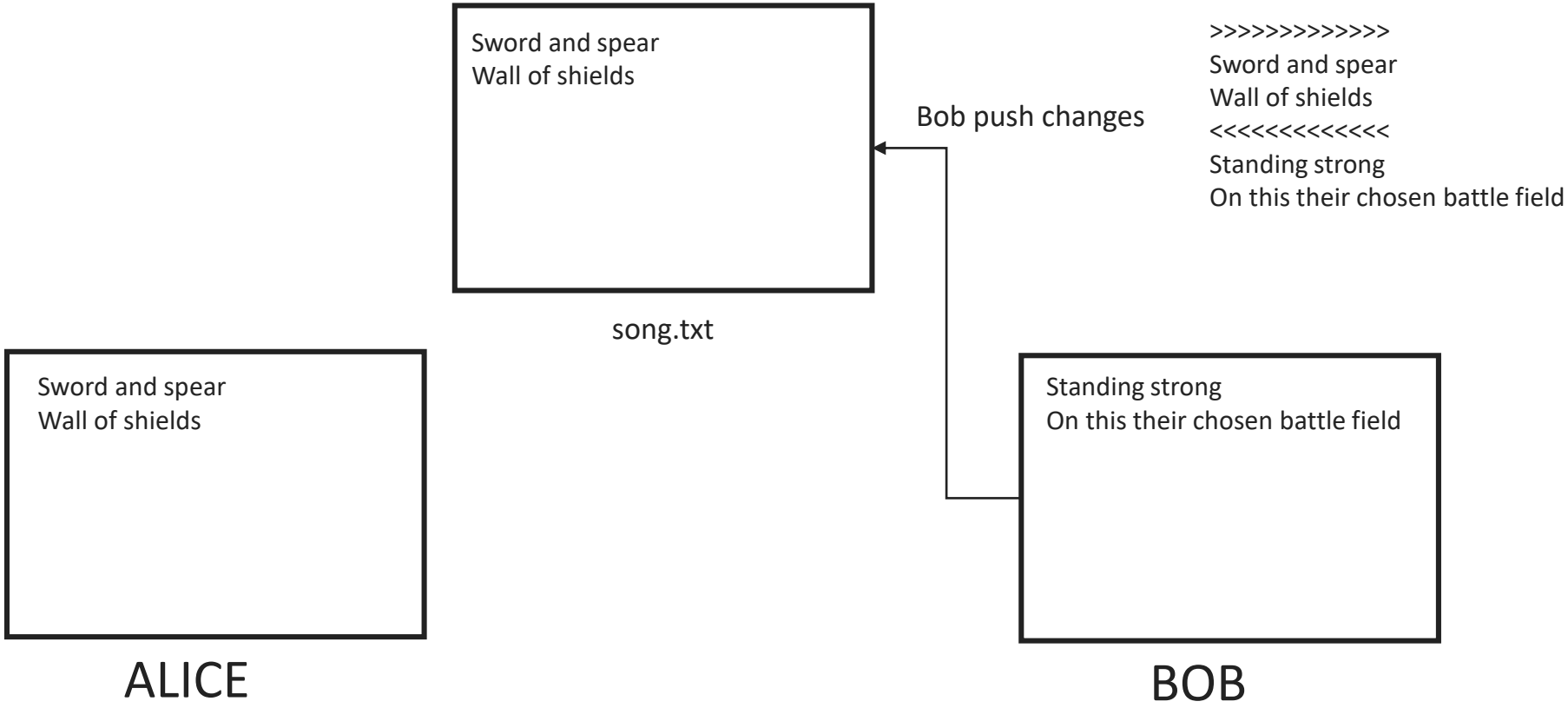
Standing strong
On this their chosen battle field

BOB

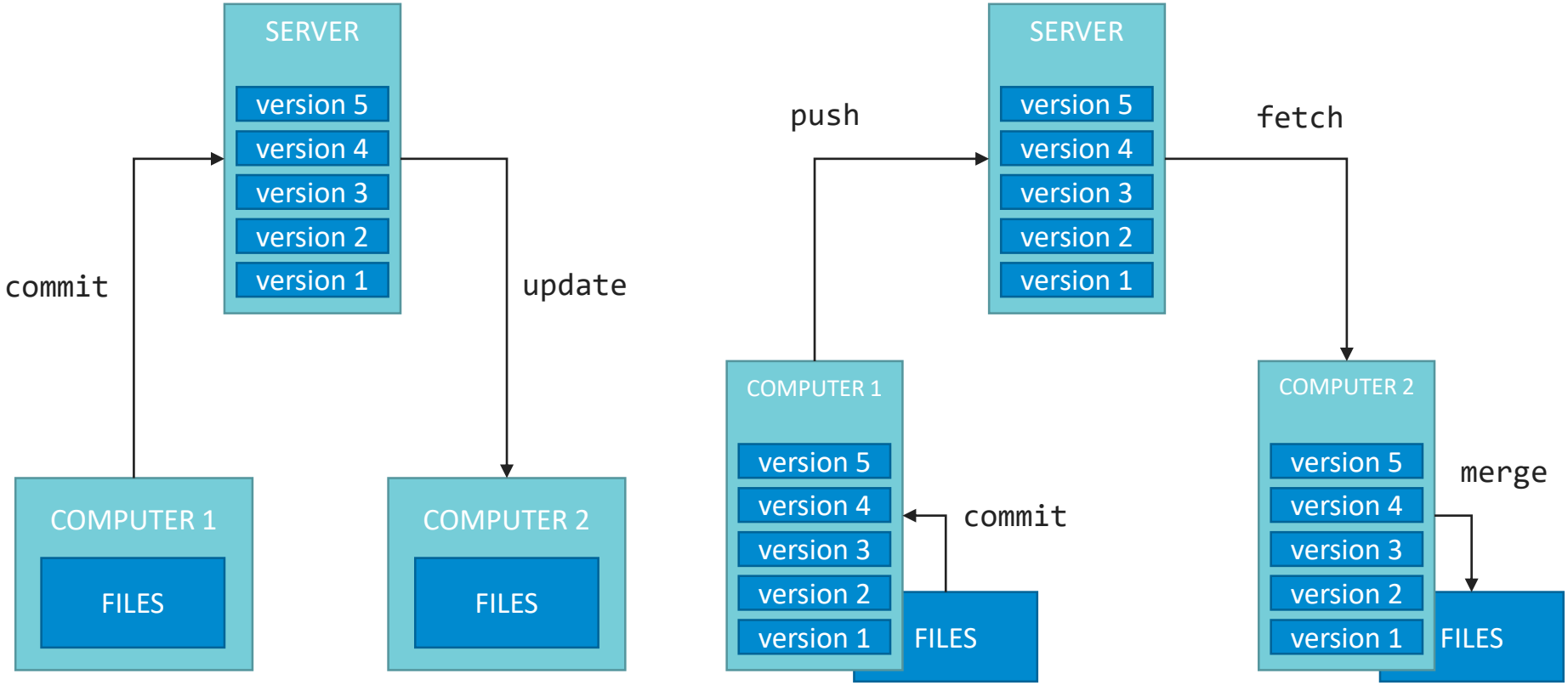
Copy-modify-merge strategy



Copy-modify-merge strategy



Centralized vs Distributed



<epam>

Why Git

Version Control with Git. DevTestOps training.

TRAINING

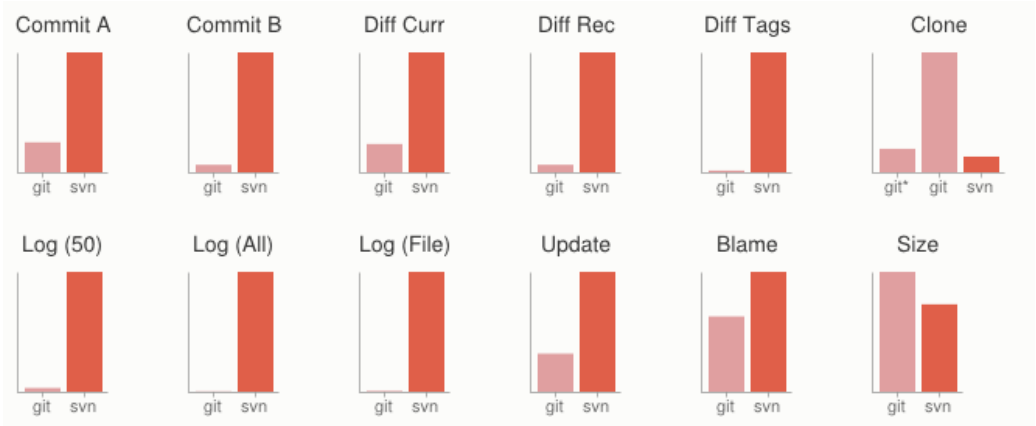
CENTER

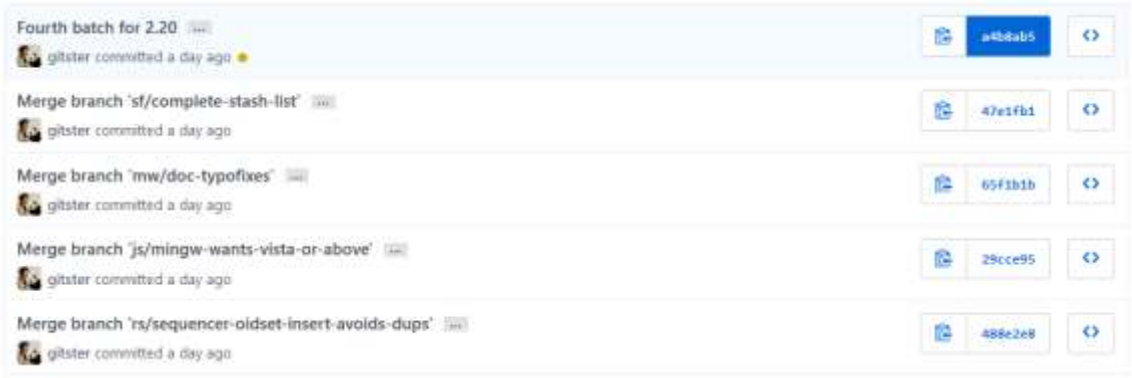
<epam>

Why Git

- Git is released under the GNU General Public License version 2.0, which is an open source license. The Git project chose to use GPLv2 to guarantee your freedom to share and change free software - to make sure the software is free for all its users.

Git benefits





<epam>

Download, install, configure

Version Control with Git. DevTestOps training.

TRAINING

CENTER

<epam>

Download, install, configure

DOWNLOAD & INSTALL

- Download binary from here: http://git-scm.com/downloads
- Follow the steps using the default options

CONFIGURE

- Generate SSH key pair

```
ssh-keygen -t rsa –C “vitali_shulha@epam.com"
```

- Send public key to repository owner or upload to your profile
- Configure username and email

```
git config --global user.name “Vitali Shulha“
git config --global user.email “vitali_Shulha@epam.com"
```

<epam>

Create a github repo and clone it

Version Control with Git. DevTestOps training.

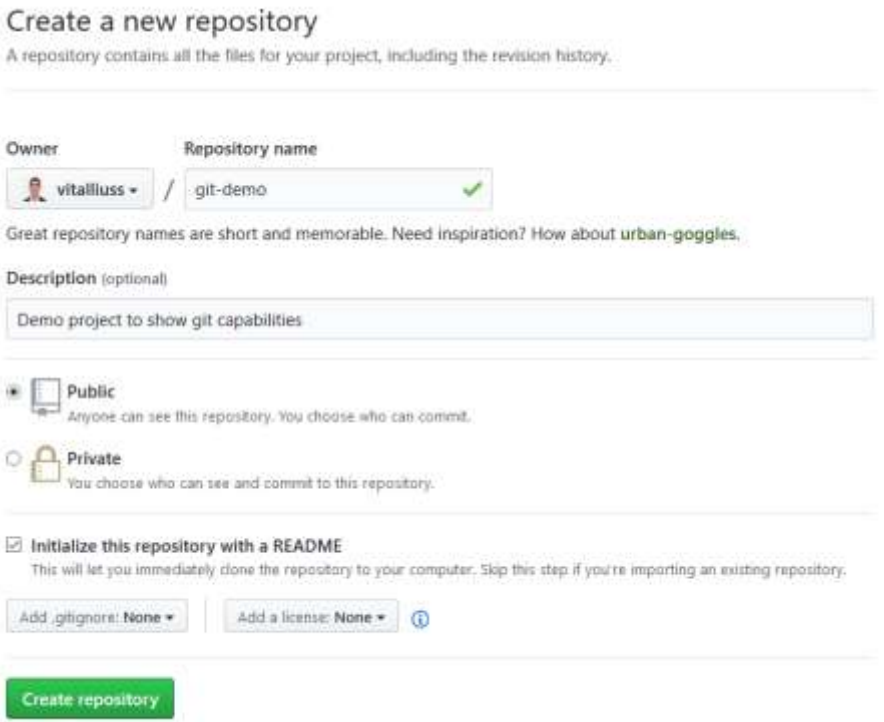
TRAINING

CENTER

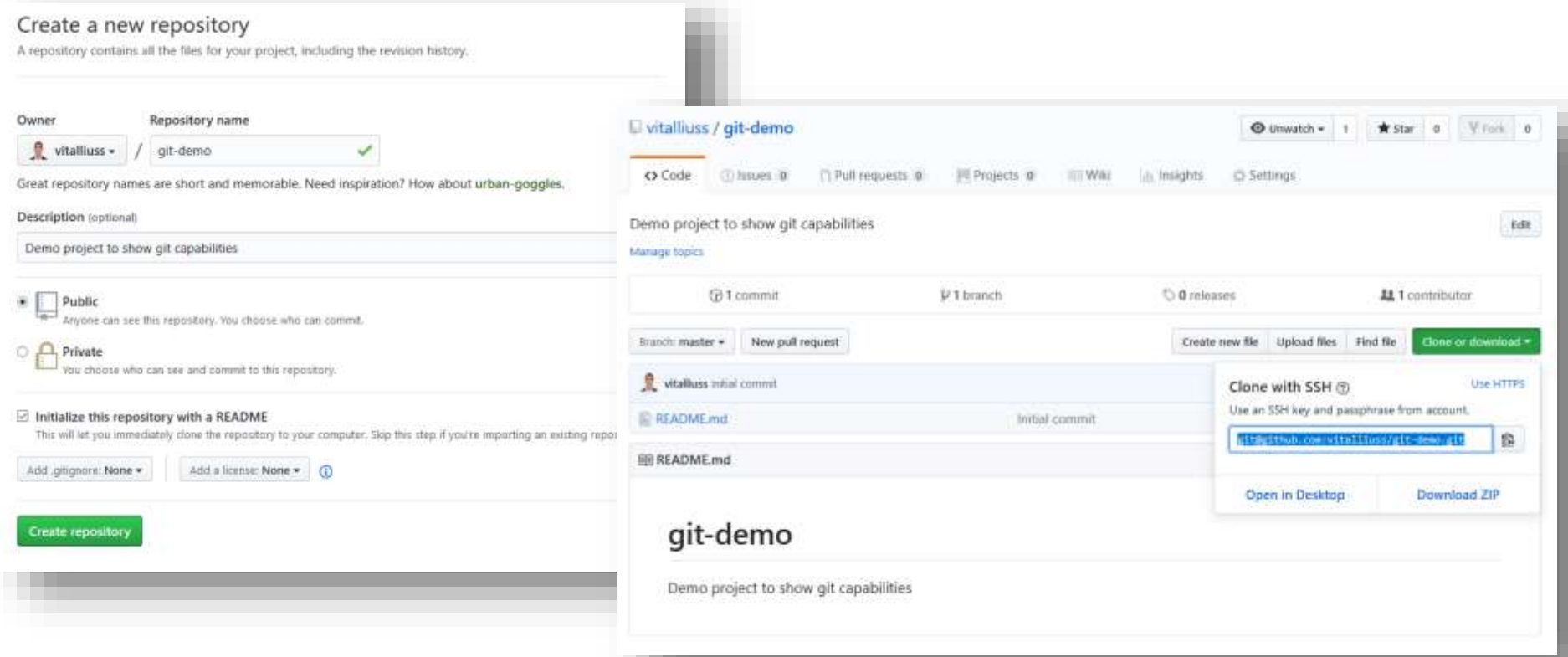
<epam>

Create github repository and clone it

- Simple as it can be
- Don’t forget to select “Initialize this repository with a README”



Create github repository



Cloning the repository

Select folder

Check content

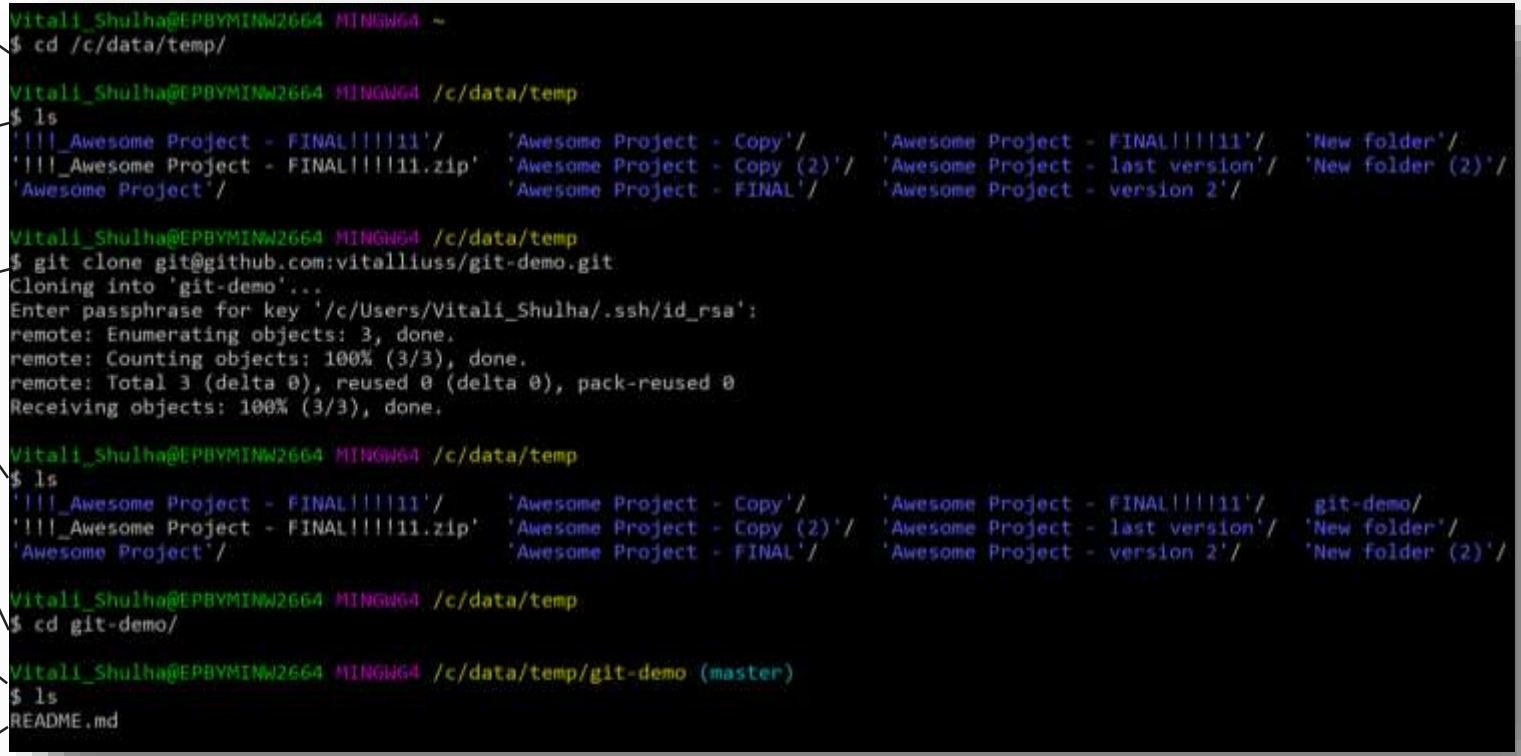
Clone repo

Ensure it’s done

Go to target dir

See master branch

See content



<epam>

Commit and push

Version Control with Git. DevTestOps training.

TRAINING

CENTER

<epam>

Making a first commit

Write line in text file

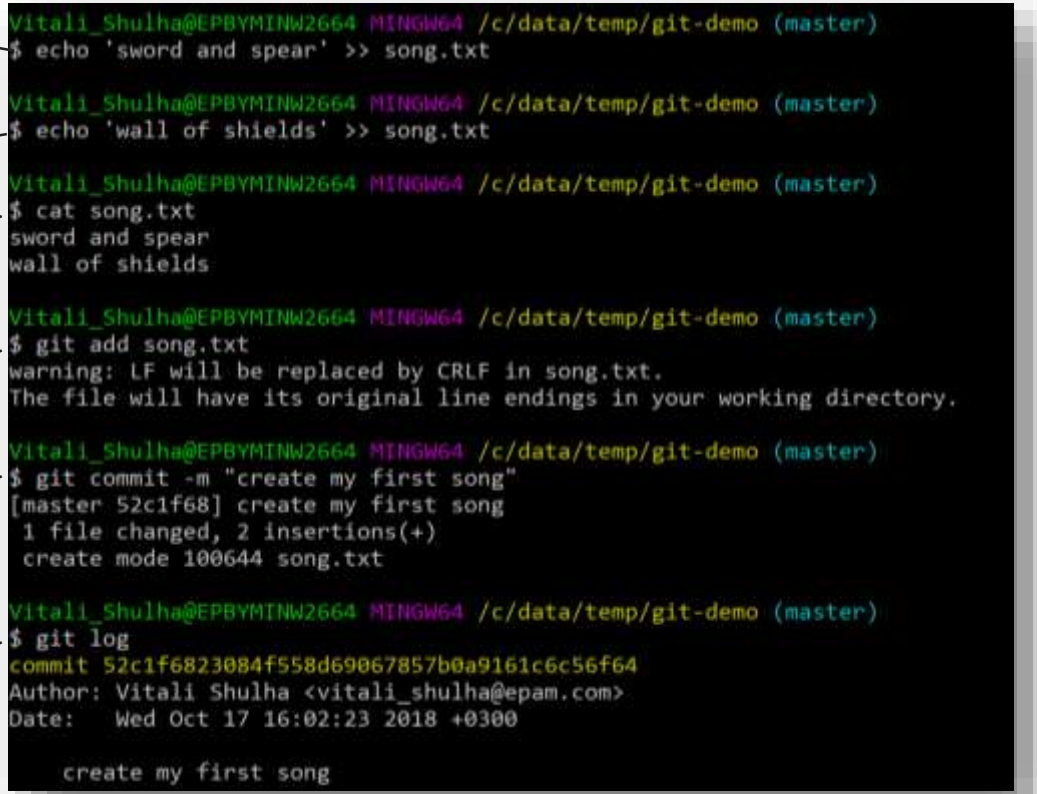
Write another line

Check the result

Add file to staging area

Commit the staged files

See the commit done



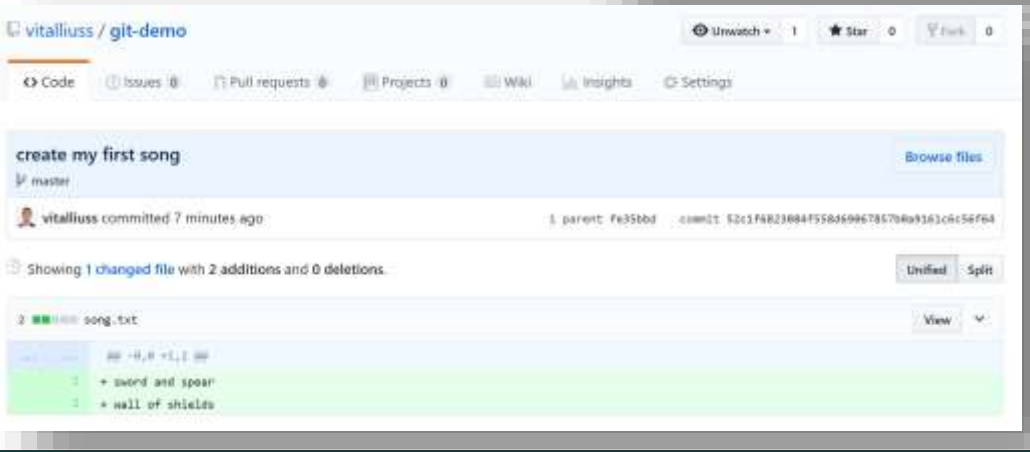
Push in to github

Check link to github.com

Push the commits to remote

```
Vitali_Shulha@EPBYMINW2664 MINGW64 /c/data/temp/git-demo (master)
$ git remote -v
origin  git@github.com:vitalliuss/git-demo.git (fetch)
origin  git@github.com:vitalliuss/git-demo.git (push)

Vitali_Shulha@EPBYMINW2664 MINGW64 /c/data/temp/git-demo (master)
$ git push
Enter passphrase for key '/c/Users/Vitali_Shulha/.ssh/id_rsa':
Counting objects: 3, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 320 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To github.com:vitalliuss/git-demo.git
   fe35bbd..52c1f68  master -> master
```



<epam>

Pull from remote

Version Control with Git. DevTestOps training.

TRAINING

CENTER

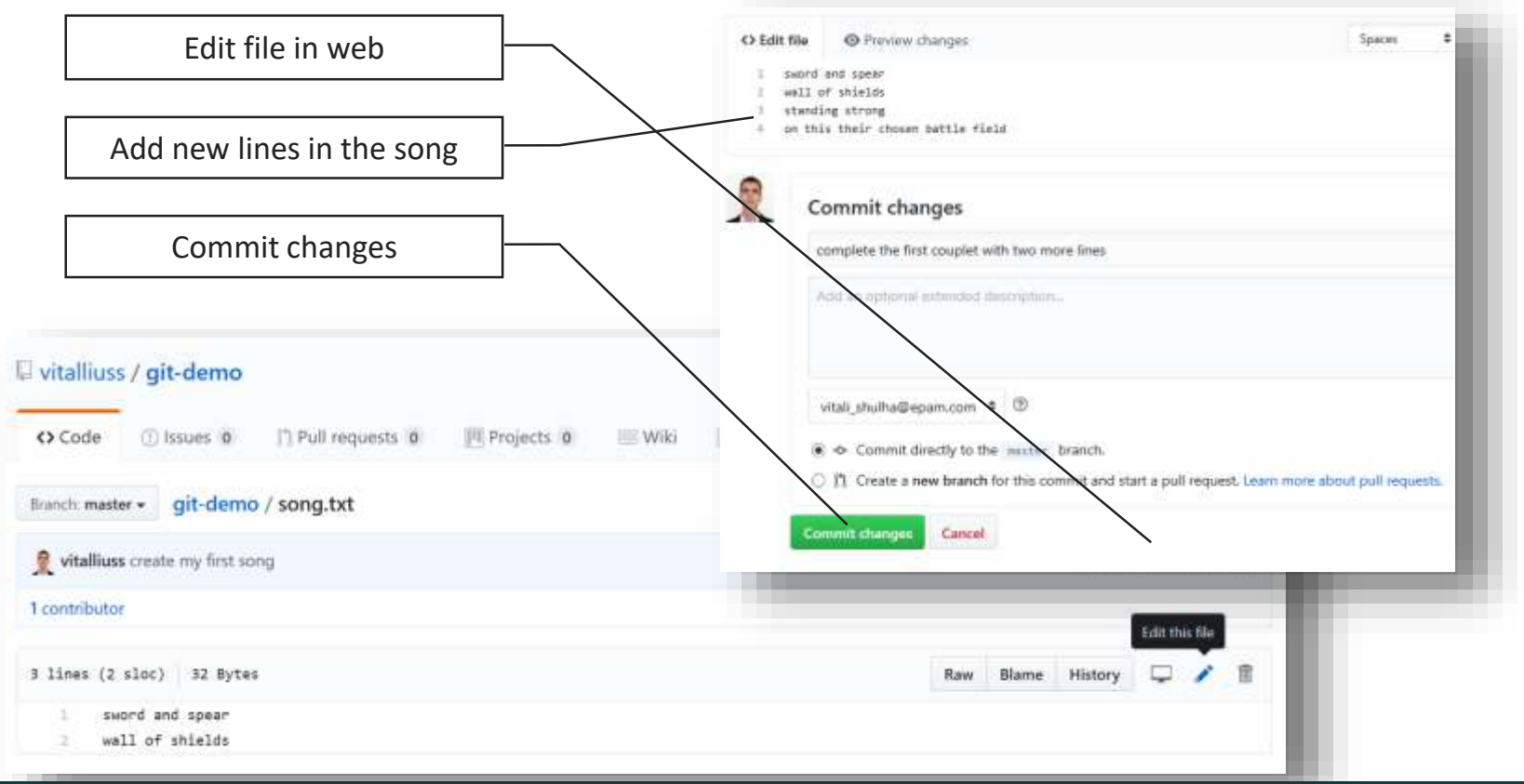
<epam>

Create commit in github and pull it

Edit file in web

Add new lines in the song

Commit changes



Create commit in github and pull it

Ensure the commit done

Pull code from github

See the changes arrived

```
Vitali_Shulha@EPBYMINW2664 MINGW64 /c/data/temp/git-demo (master)
$ git pull
Enter passphrase for key '/c/Users/Vitali_Shulha/.ssh/id_rsa':
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From github.com:vitalliuss/git-demo
   52c1f68..9258d31  master     -> origin/master
Updating 52c1f68..9258d31
Fast-forward
 song.txt | 2 ++
 1 file changed, 2 insertions(+)

Vitali_Shulha@EPBYMINW2664 MINGW64 /c/data/temp/git-demo (master)
$ cat song.txt
sword and spear
wall of shields
standing strong
on this their chosen battle field
```

<epam>

Git GUI & gitk

Version Control with Git. DevTestOps training.

TRAINING

CENTER

<epam>

<epam>

Undoing changes

Version Control with Git. DevTestOps training.

TRAINING

CENTER

<epam>

Undoing changes

Working directory

```
git checkout -- file.txt
git checkout .
git clean -xdf
```

Staging area (Index)

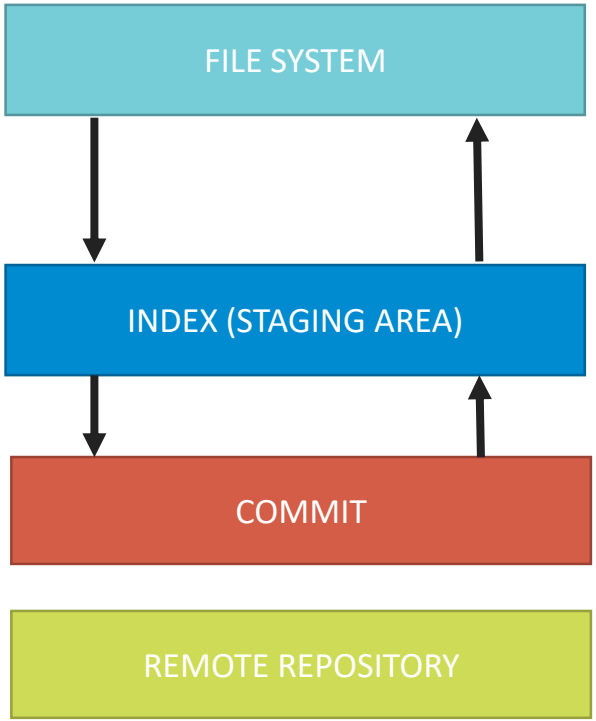
```
git reset -- file.txt
```

Local branch

```
git reset HEAD^^ (HEAD~2)
git commit --amend -m “commit message”
```

Remote repository

```
git revert <sha1>
```



<epam>

Git revert

Version Control with Git. DevTestOps training.

TRAINING

CENTER

<epam>

Undoing changes

Working directory

```
git checkout -- file.txt
git checkout .
git clean -xdf
```

Staging area (Index)

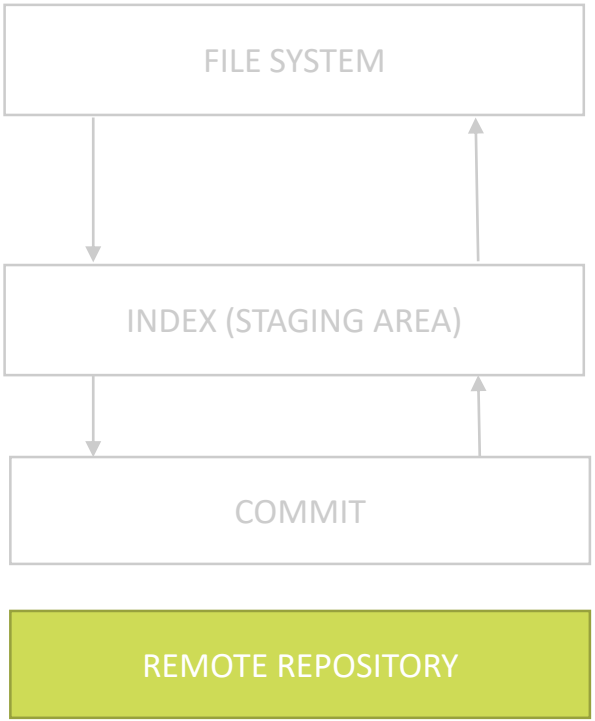
```
git reset -- file.txt
```

Local branch

```
git reset HEAD^^ (HEAD~2)
git commit --amend -m “commit message”
```

Remote repository

```
git revert <sha1>
```



<epam>

Git reset

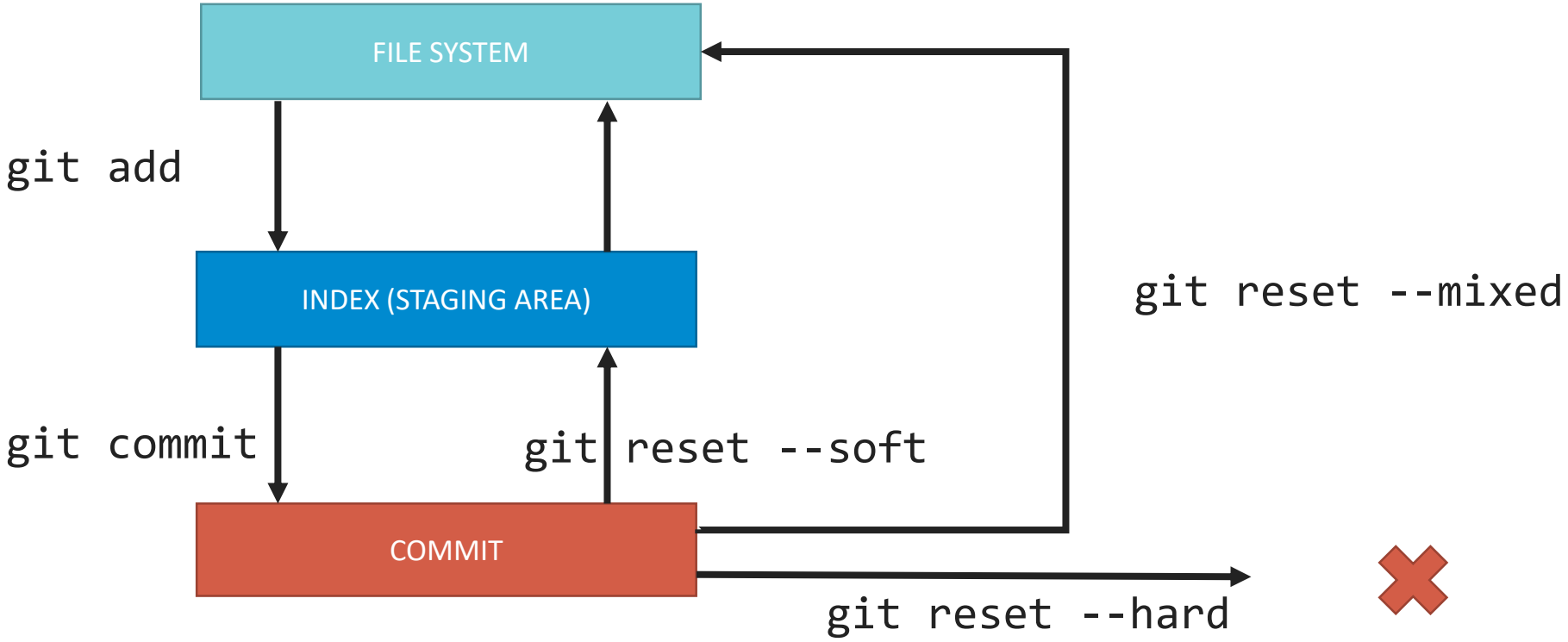
Version Control with Git. DevTestOps training.

TRAINING

CENTER

<epam>

Git reset


FILE SYSTEM
git add
INDEX (STAGING AREA)
git commit
git reset --soft
COMMIT
git reset --mixed
git reset --hard

<epam>

.gitignore

Version Control with Git. DevTestOps training.

TRAINING

CENTER

<epam>

.gitignore

```
# no .log files
*.log

# but do track error.log, even though you're
ignoring .log files above
!error.log

# only ignore the TODO file in the current
directory, not subdir/TODO
/TODO

# ignore all files in the build/ directory
build/

# ignore doc/notes.txt, but not doc/server/arch.txt
doc/*.txt

# ignore all .pdf files in the doc/ directory
doc/**/*.pdf
```

<epam>

Branching and merge

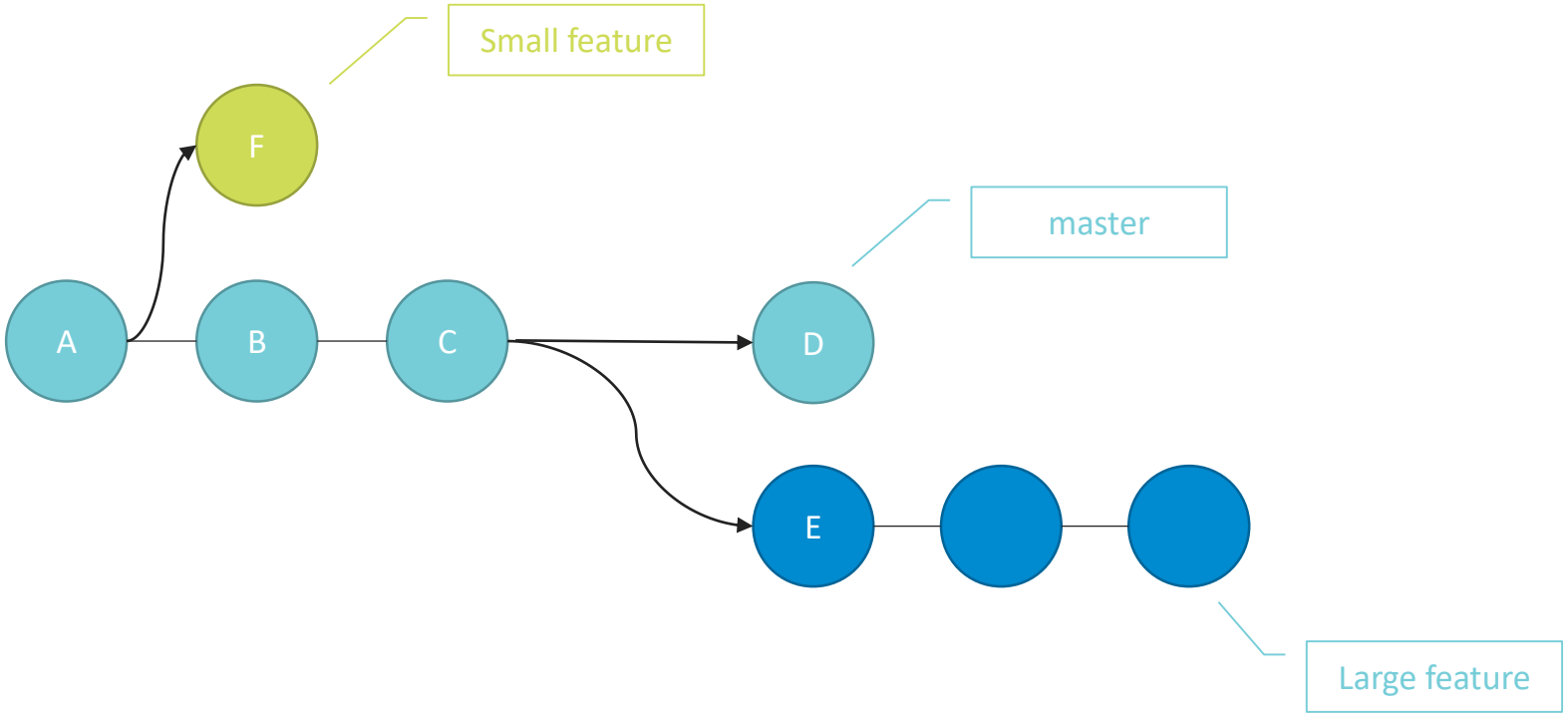
Version Control with Git. DevTestOps training.

TRAINING

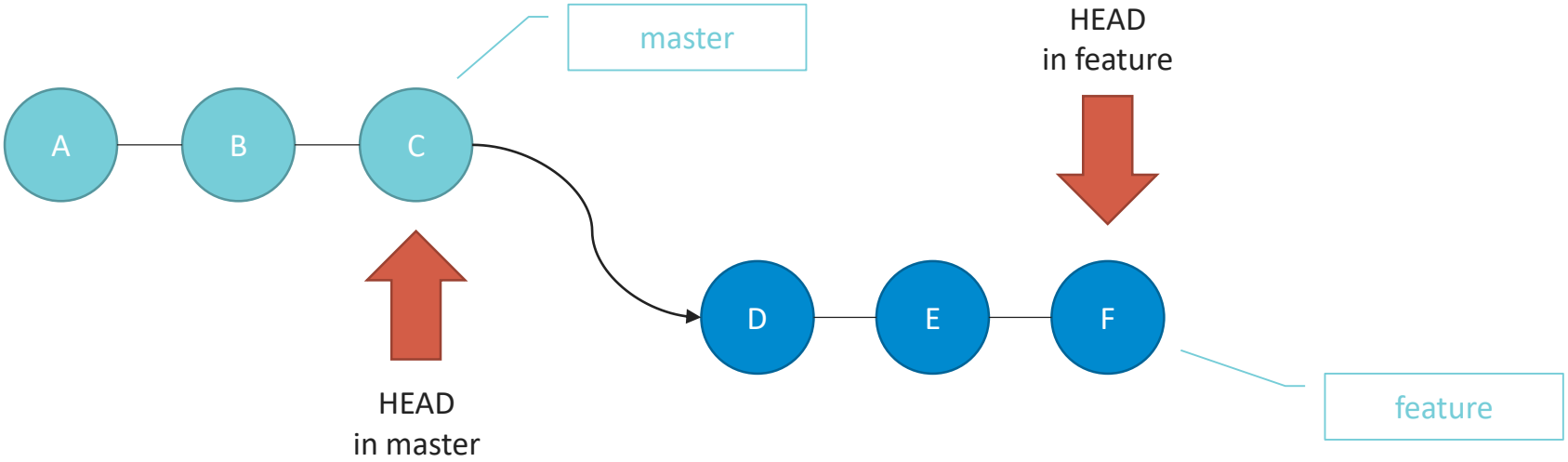
CENTER

<epam>

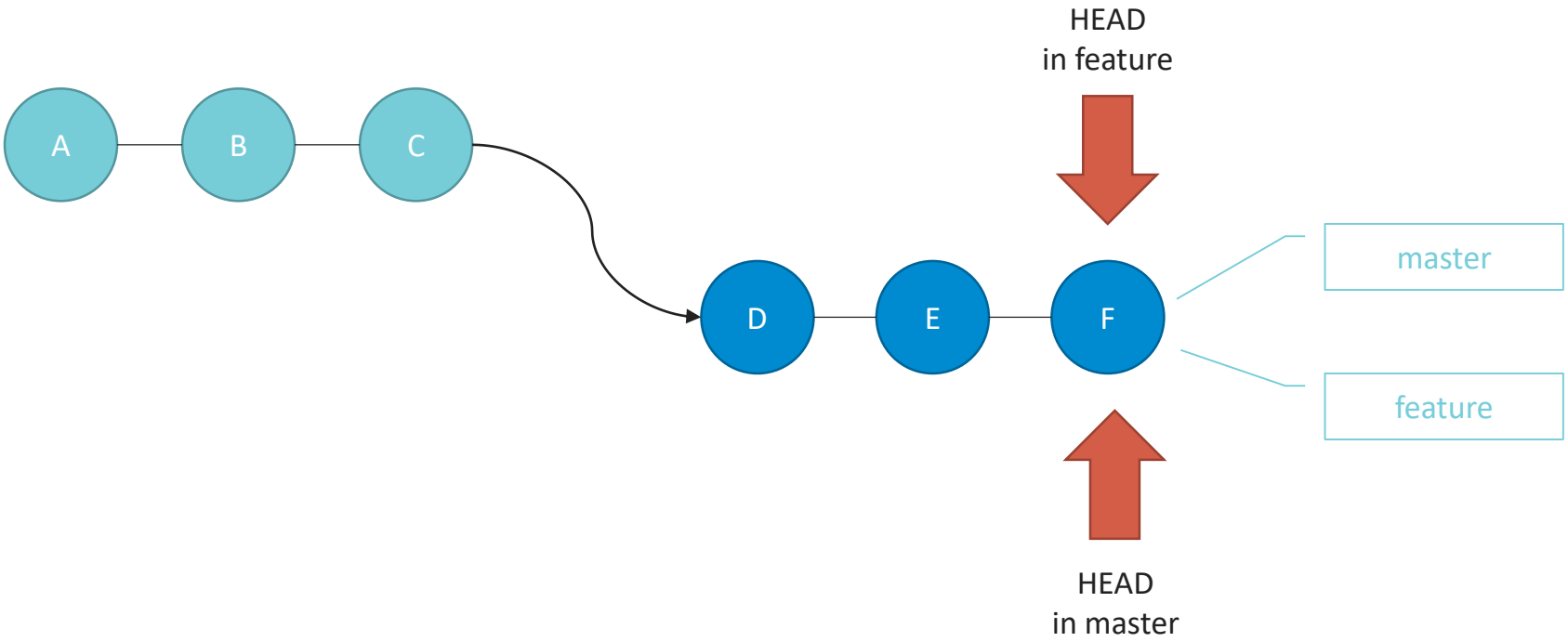
Branch concept



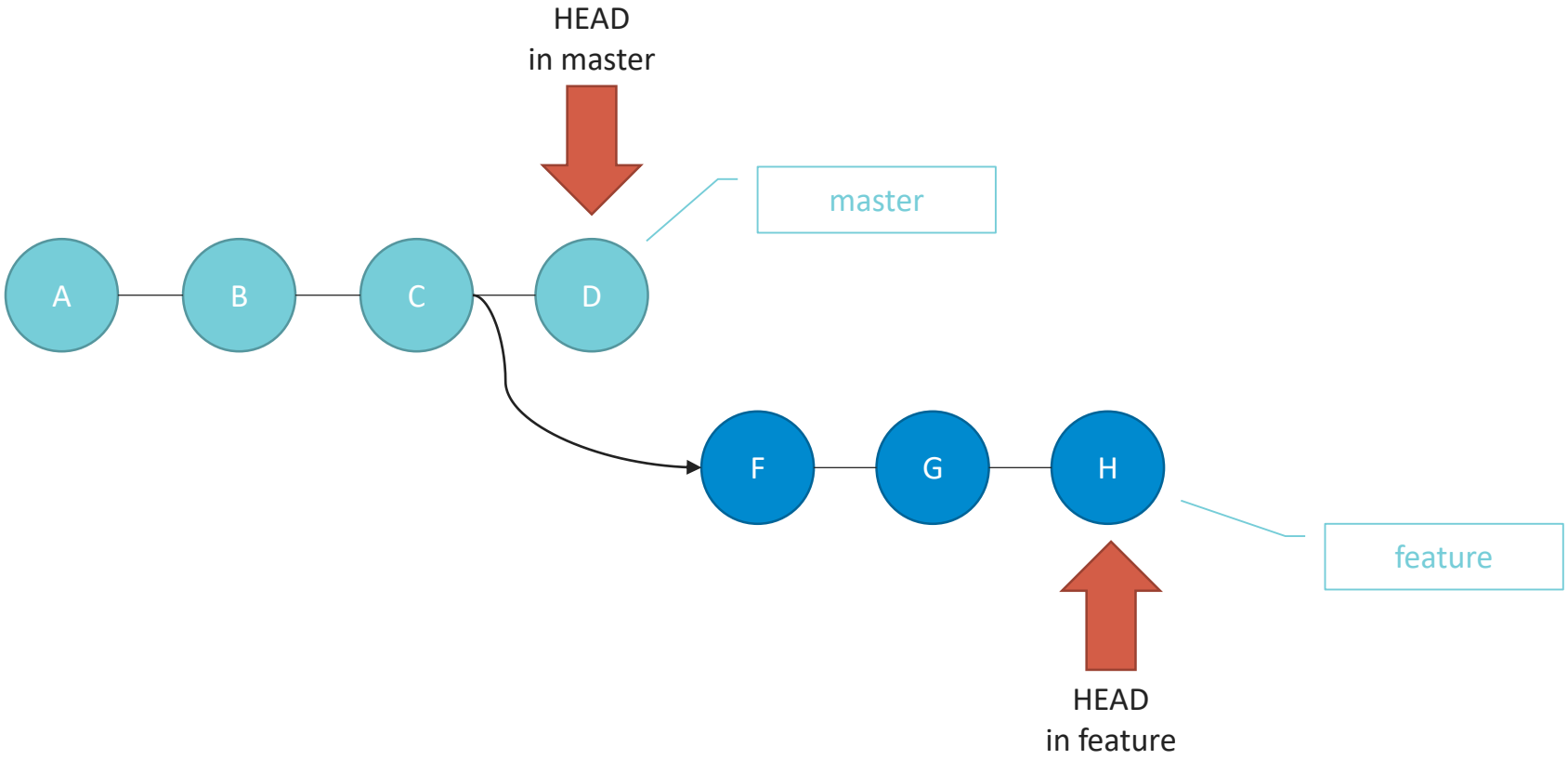
Fast-forward merge



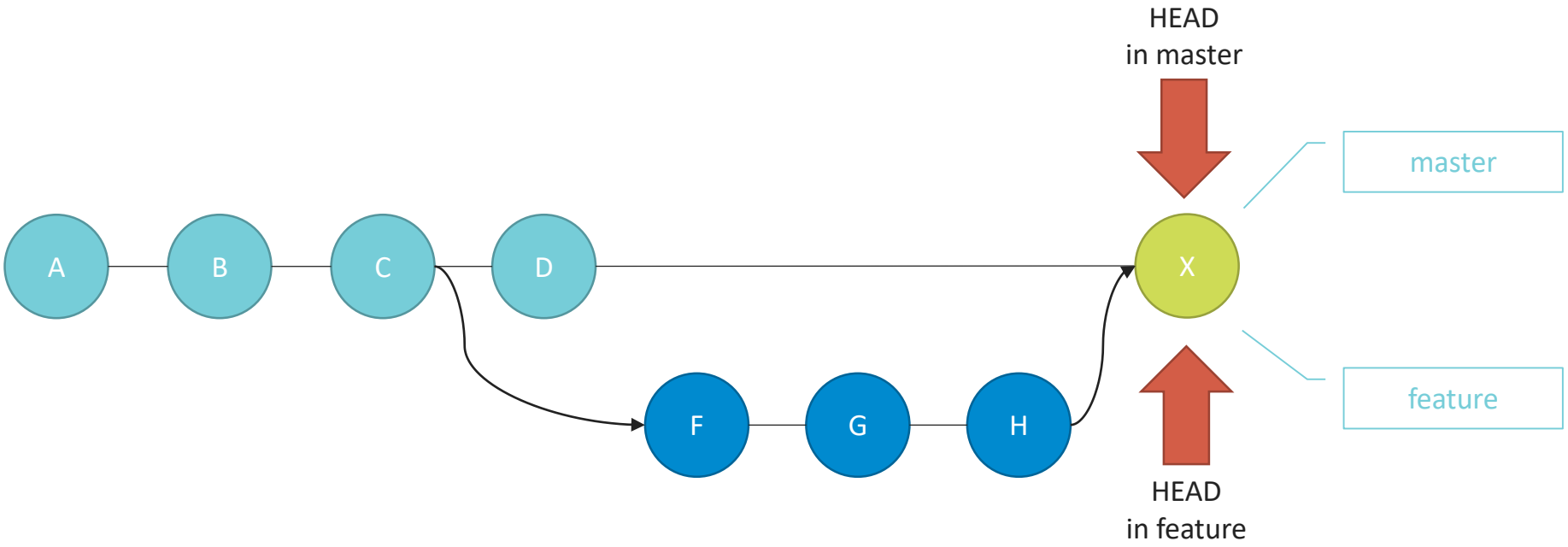
Fast-forward merge



Non fast-forward merge



Non fast-forward merge



<epam>

Conflict solving

Version Control with Git. DevTestOps training.

TRAINING

CENTER

<epam>

Conflicts solving

SOLVE CONFLICT

Abort merge

```
git merge --abort
```

Resolve by selecting version

```
git checkout --Xours --Xtheirs
```

Resolve manually

```
git diff
```

Undo merge

```
git revert 09fe472
```

User merge tool

AVOID CONFLICT

- Short commits
- No edits to whitespaces
- Merge often

<epam>

Rebase

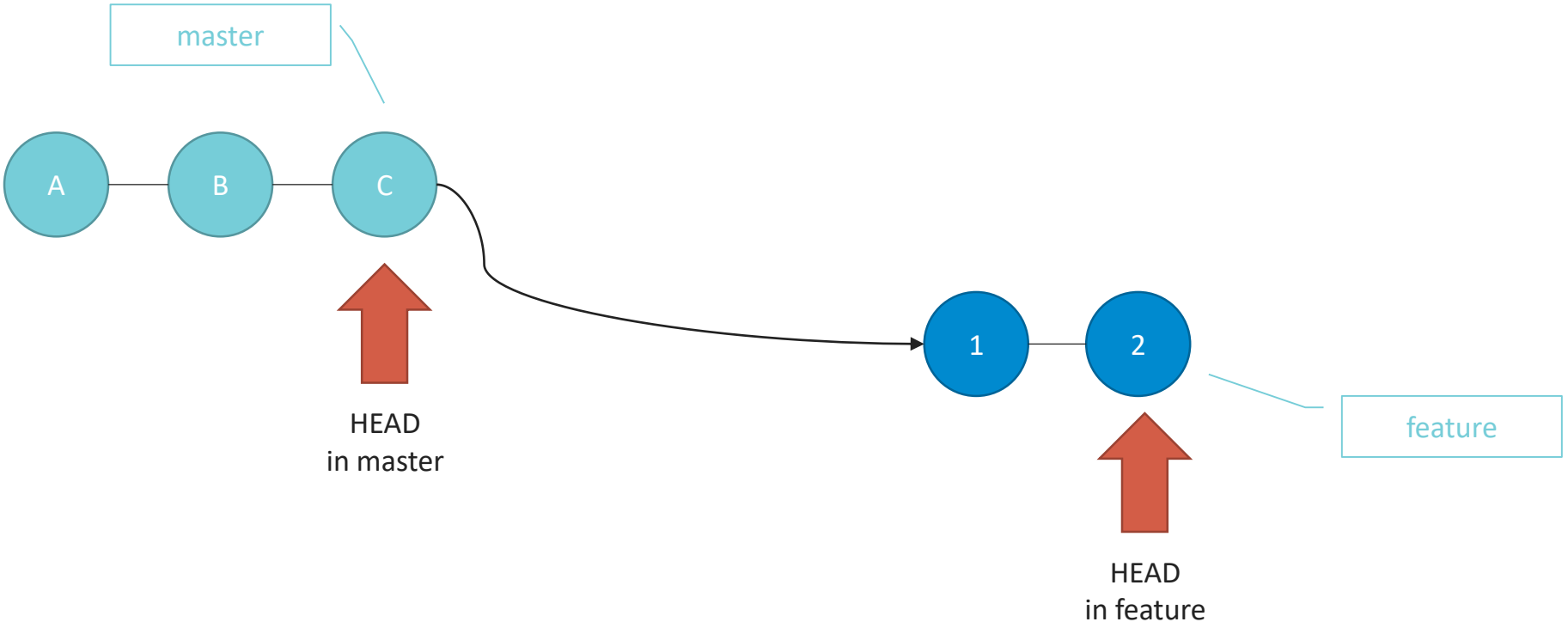
Version Control with Git. DevTestOps training.

TRAINING

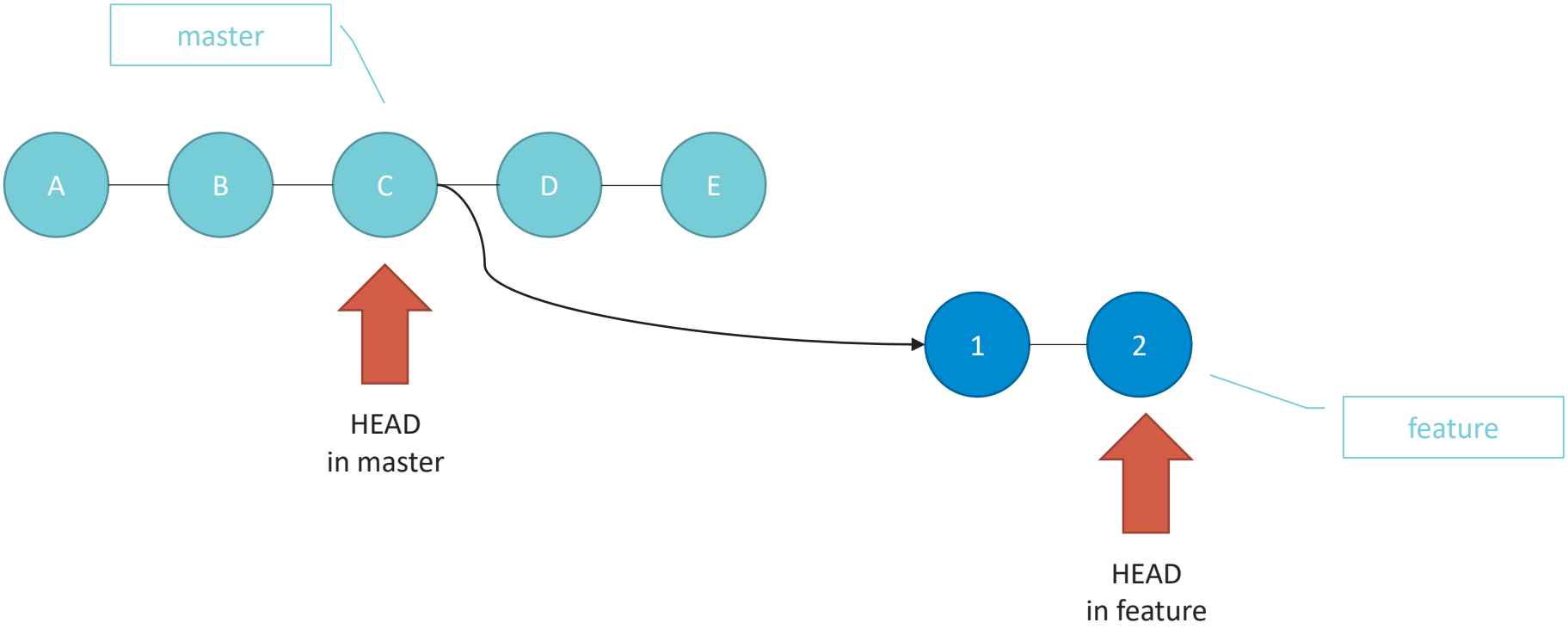
CENTER

<epam>

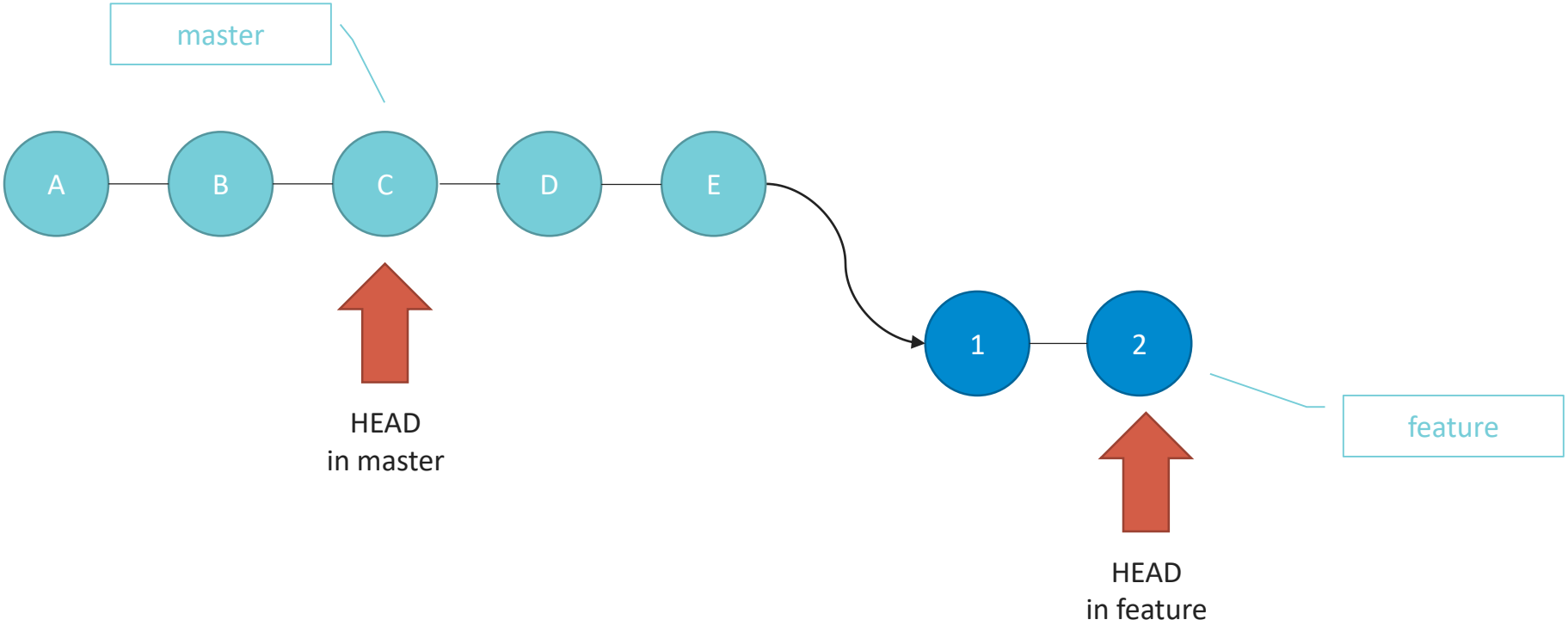
Rebase



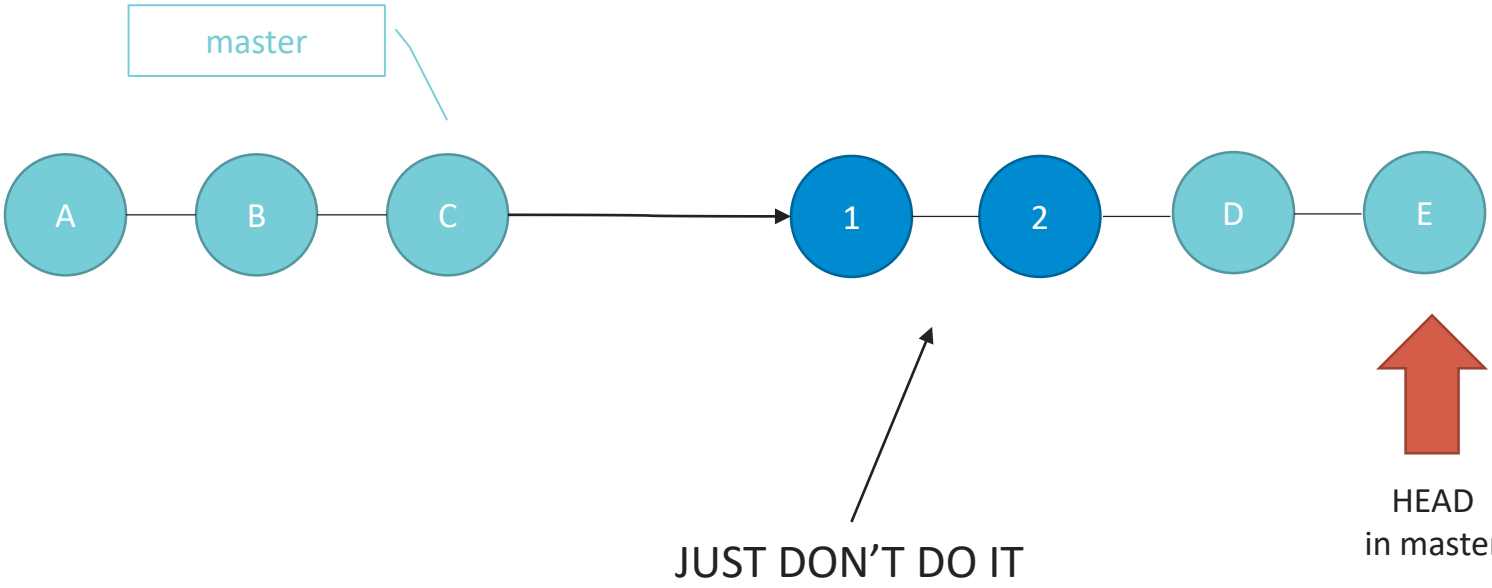
Rebase



Rebase



Golden rule of Rebase



<epam>

Cherry-pick

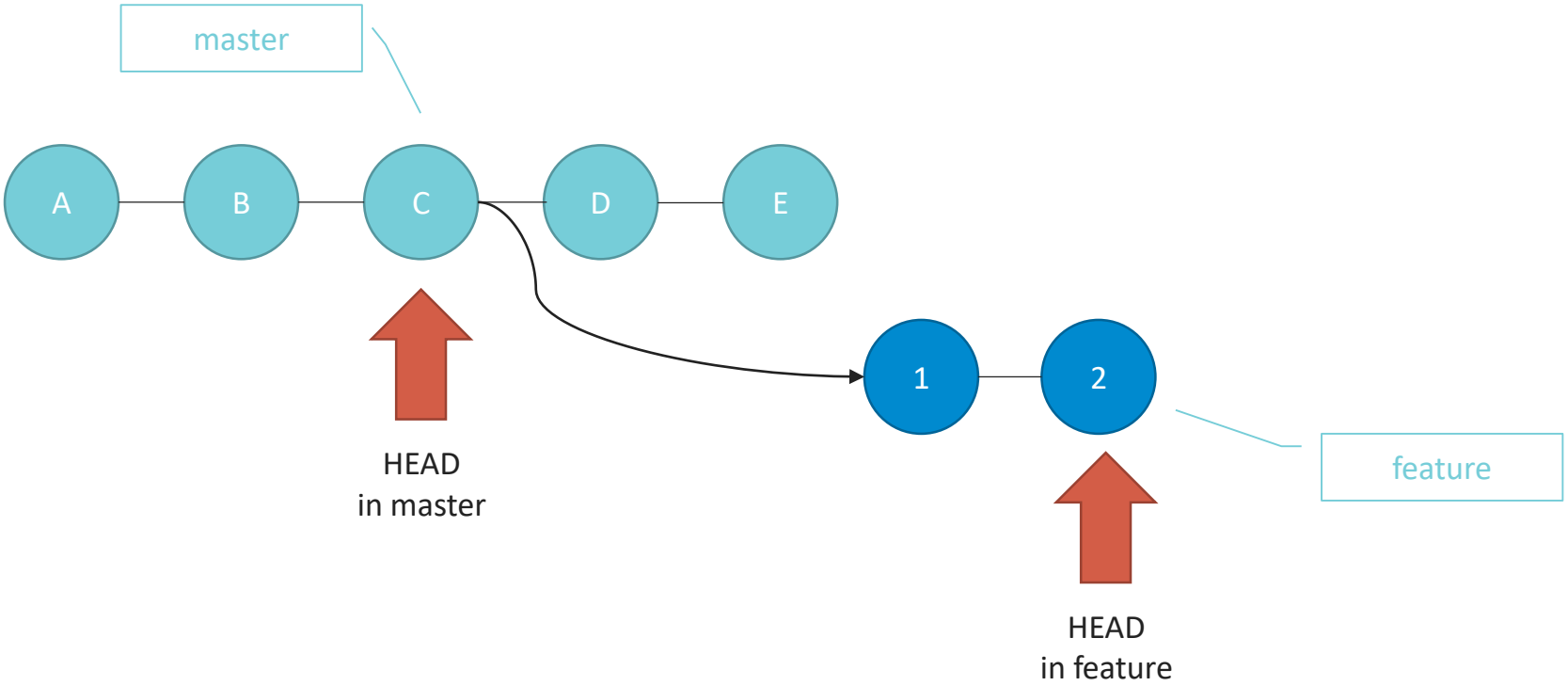
Version Control with Git. DevTestOps training.

TRAINING

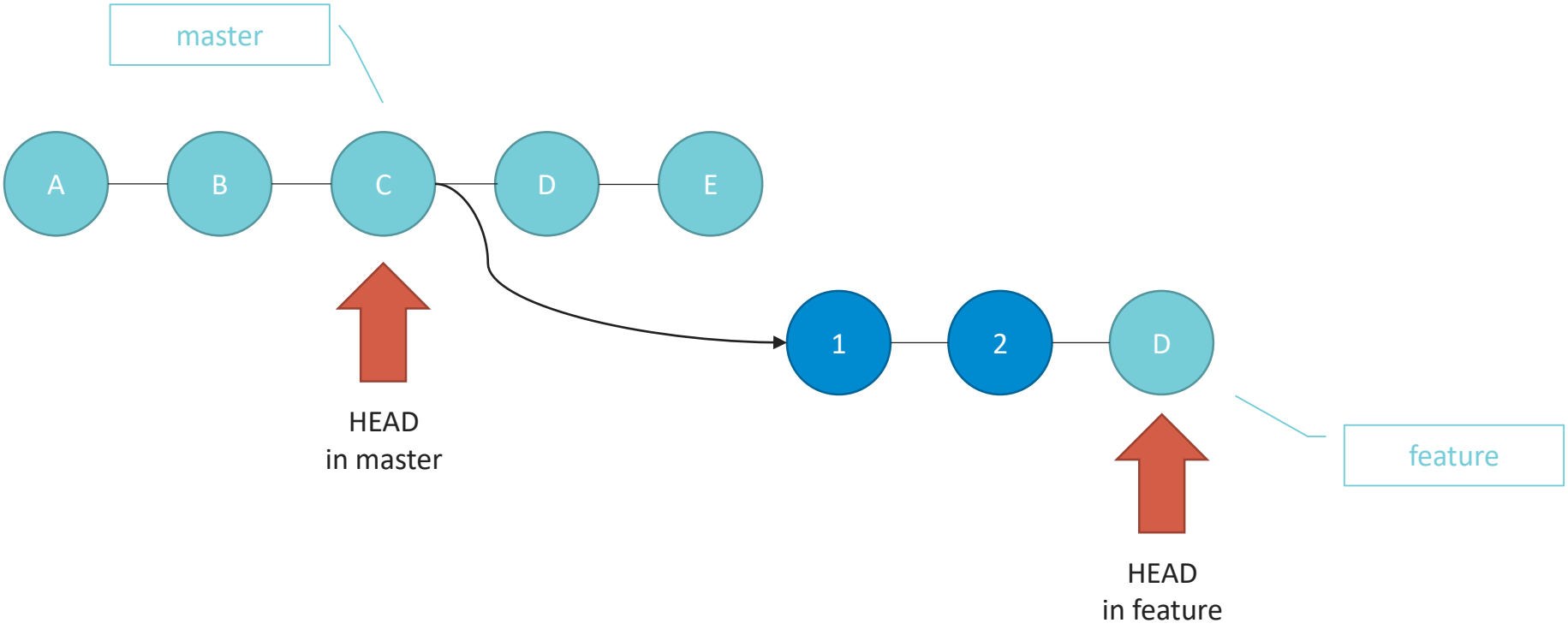
CENTER

<epam>

Cherry pick



Cherry-pick



<epam>

Tags

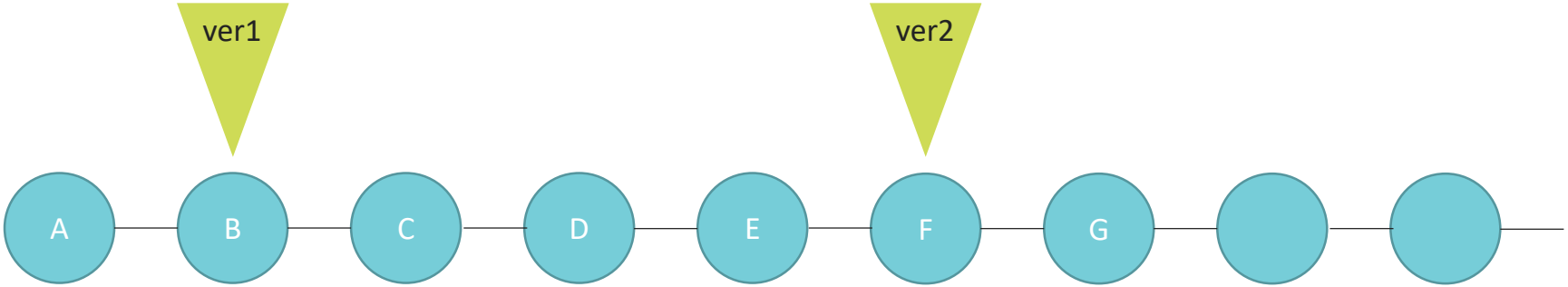
Version Control with Git. DevTestOps training.

TRAINING

CENTER

<epam>

Tags



Mark commit with tag

```
git tag ver1
```

View tags

```
git tag –list
```

Push

```
git push --tags
```

Check it out

```
git checkout ver1
```

<epam>

Stashing

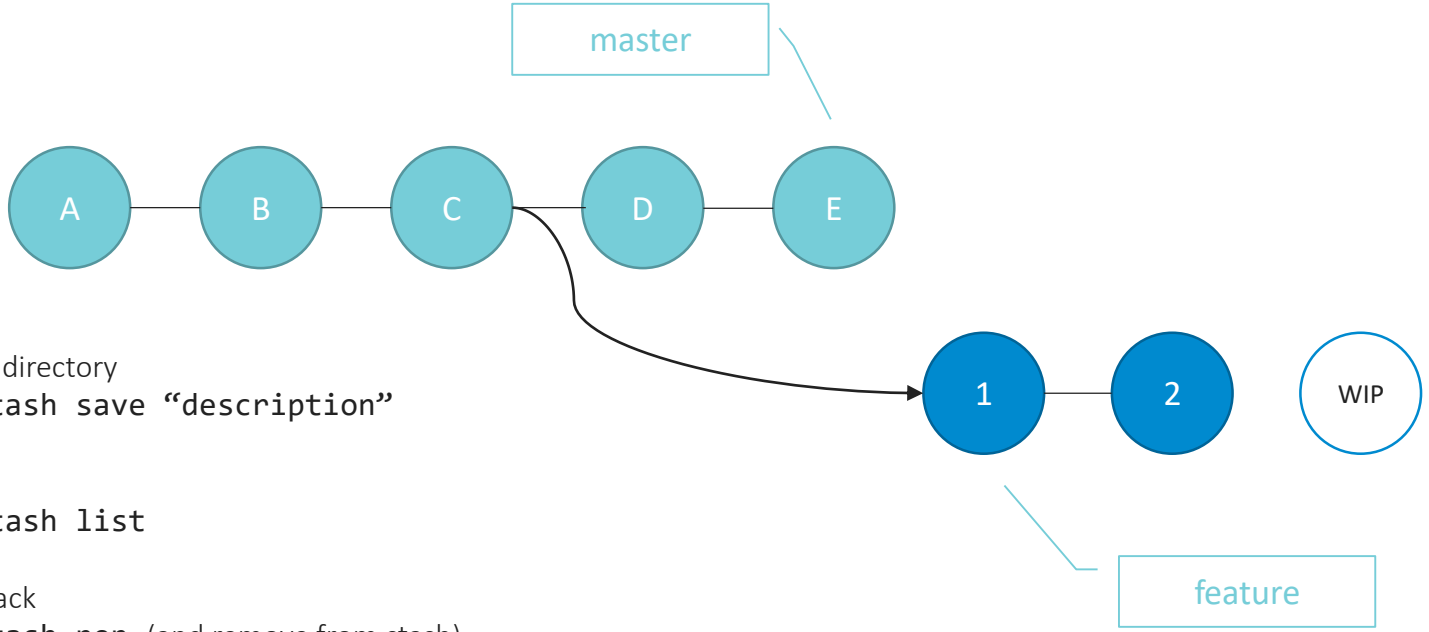
Version Control with Git. DevTestOps training.

TRAINING

CENTER

<epam>

Stash



Save working directory

```
git stash save “description”
```

View stashes

```
git stash list
```

Bring them back

```
git stash pop  (and remove from stash)
git stash apply  (leave in stash)
```

Remove

```
git stash drop  (clear)
```

<epam>

Remotes

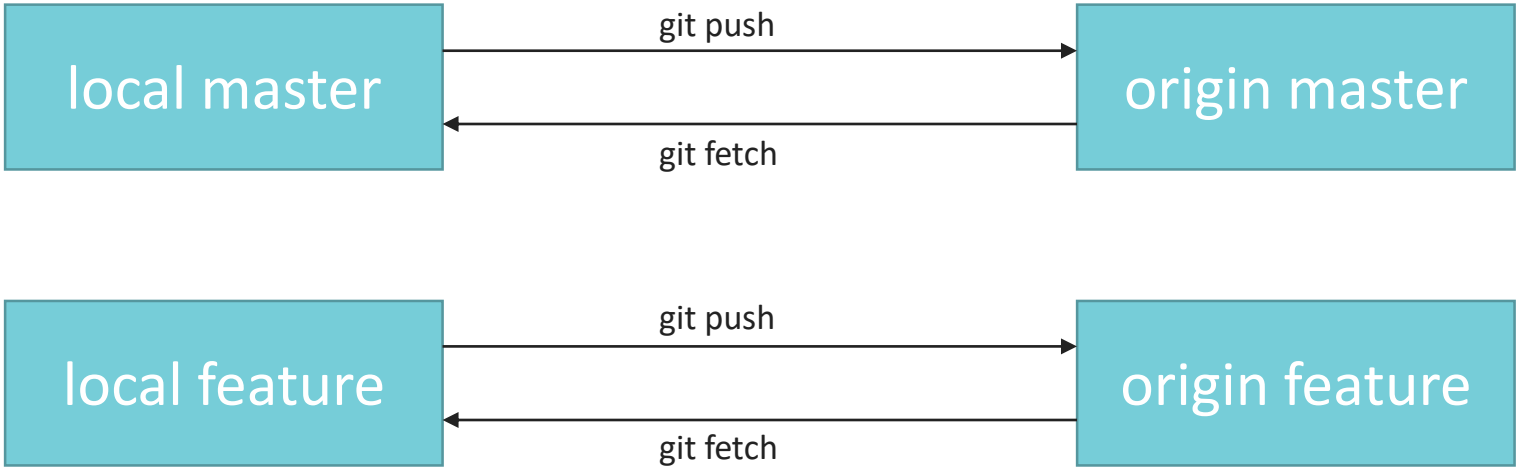
Version Control with Git. DevTestOps training.

TRAINING

CENTER

<epam>

Remotes



Add

```
git remote add <name> <url>
git remote add origin git@github.com:user/repo.git
```

View

```
git remote –v
git remote show <name>
```

<epam>

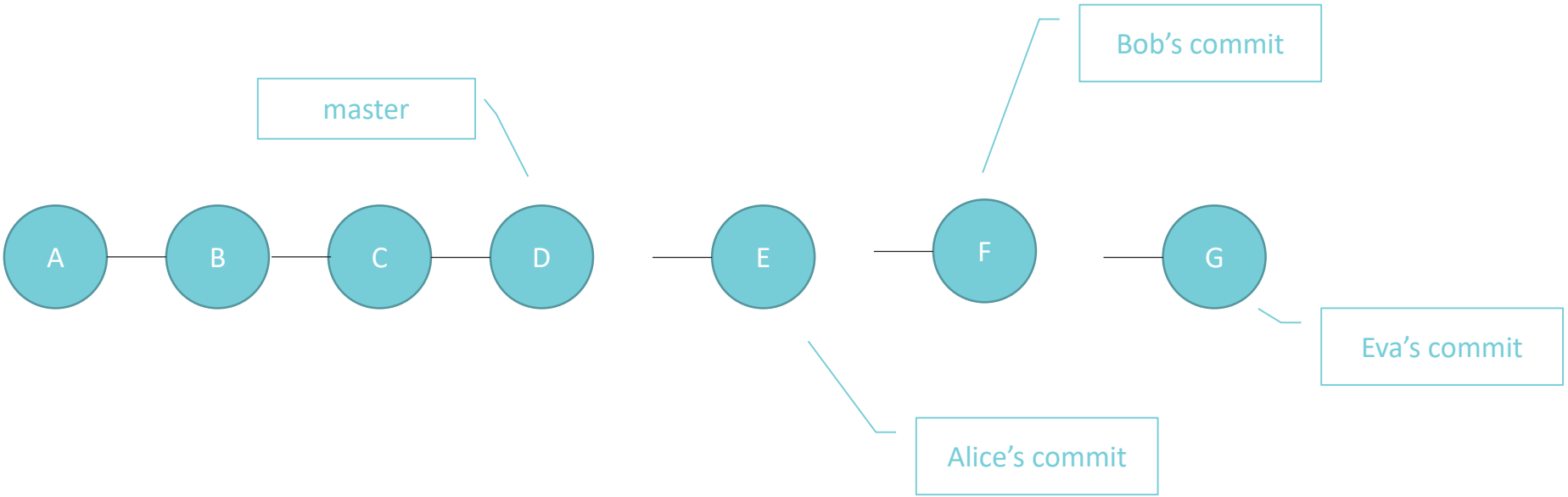
Branching strategies

Version Control with Git. DevTestOps training.

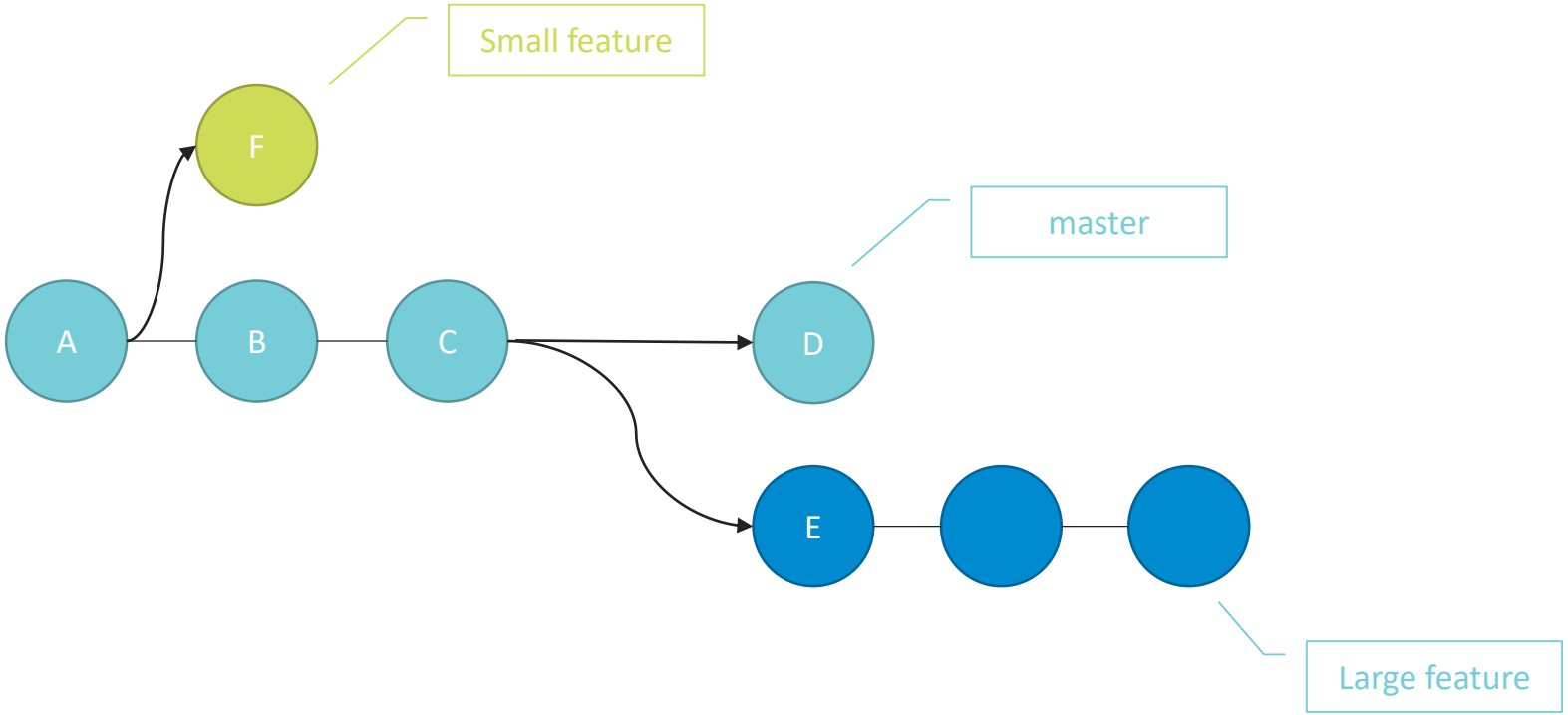
TRAINING CENTER

<epam>

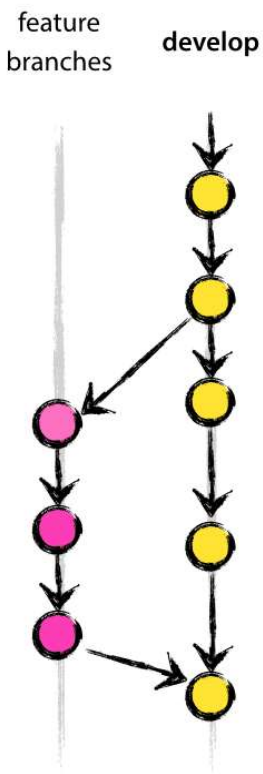
Centralized strategy

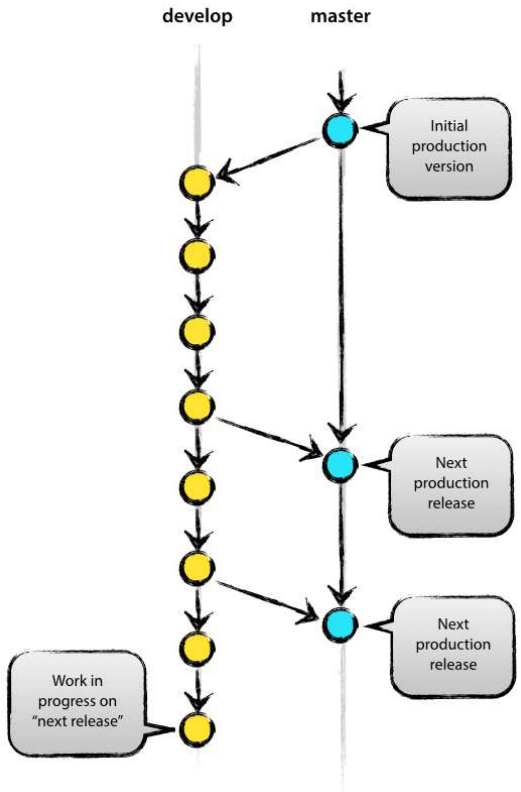


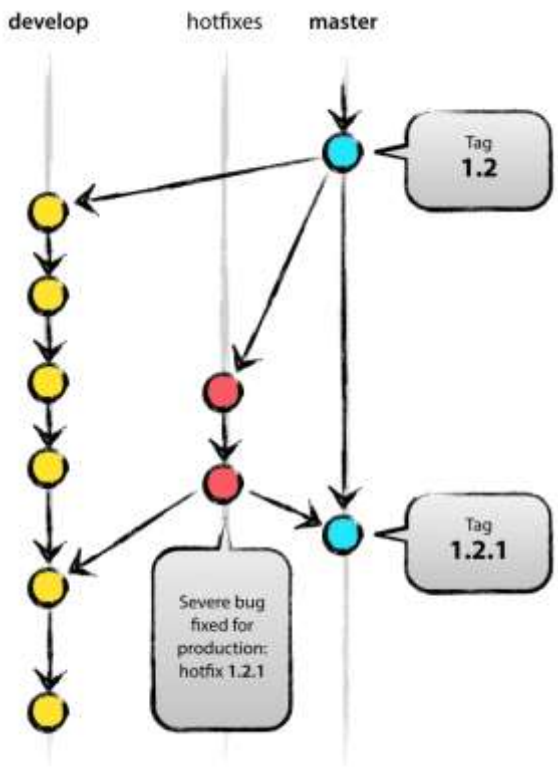
Feature-branch workflow



Gitflow

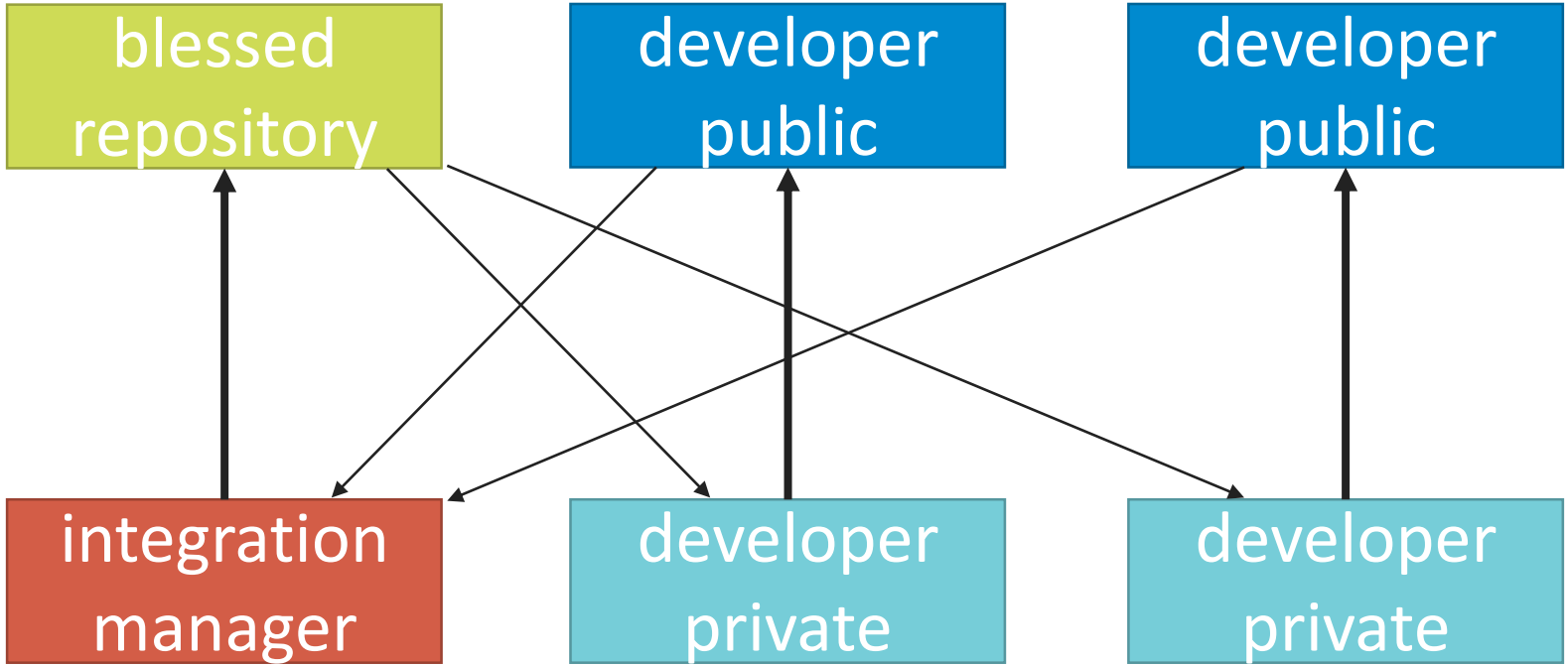




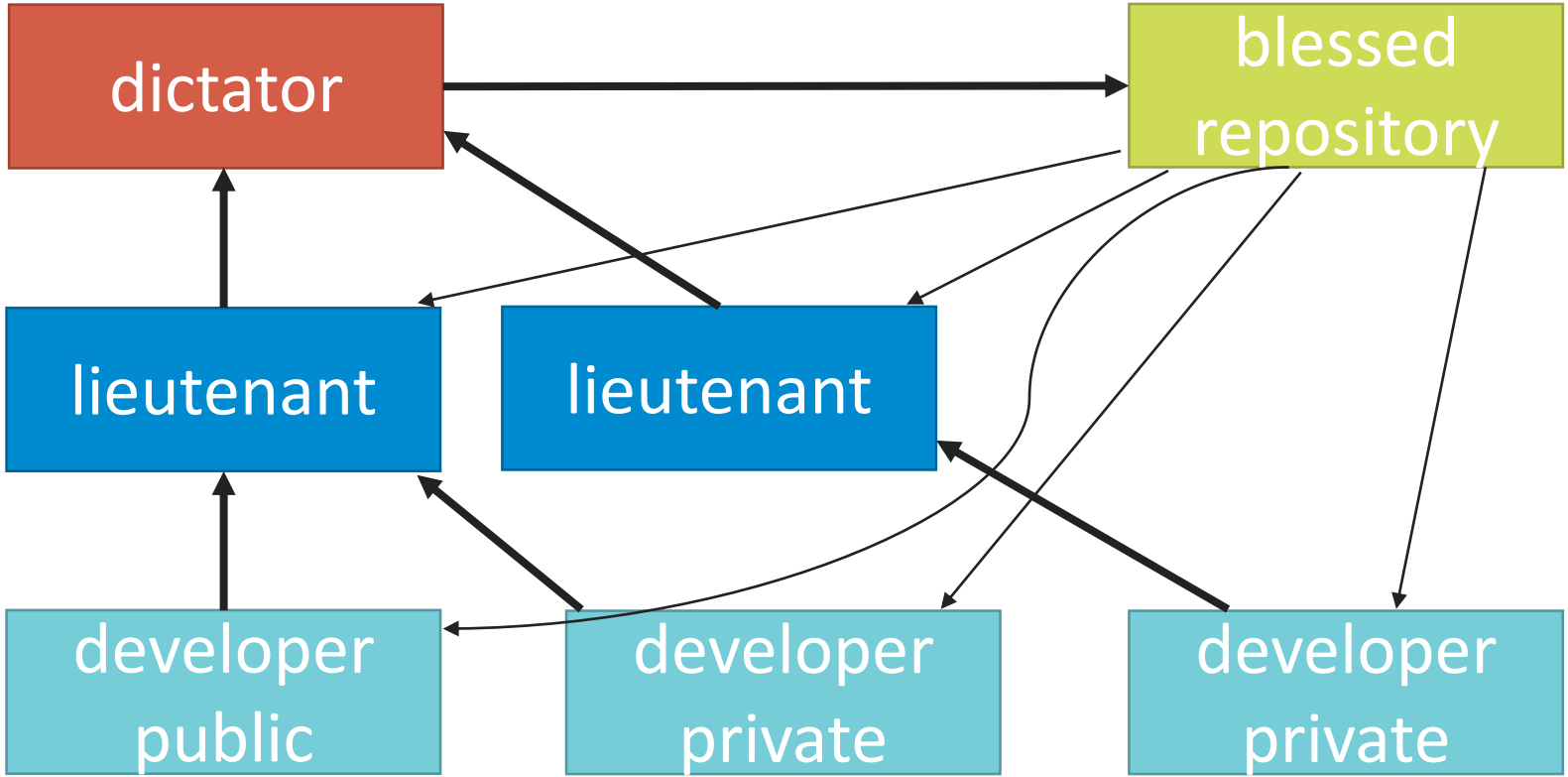


Original post by Vincent Driessen: https://nvie.com/posts/a-successful-git-branching-model/

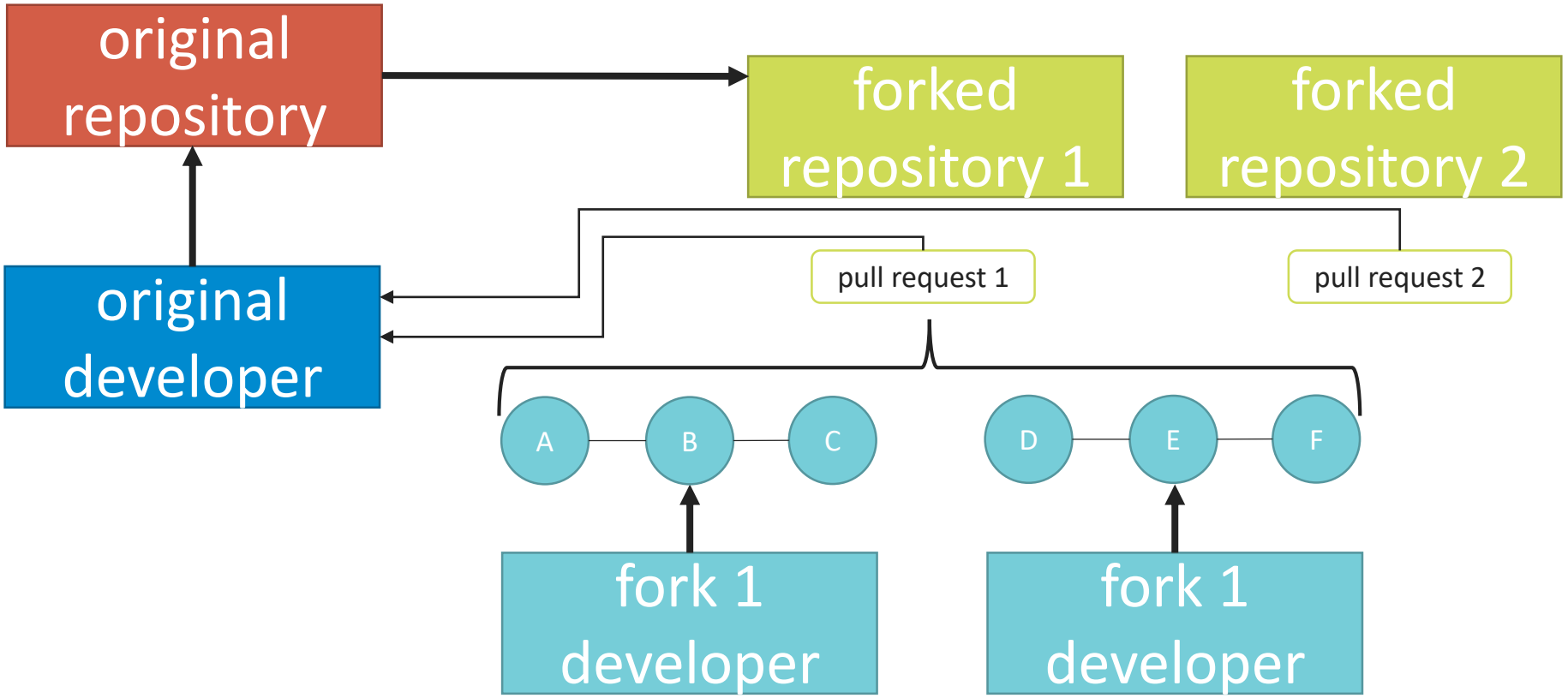
Integration manager workflow



Dictator and Lieutenants workflow



Forking workflow



<epam>

Inside .git folder

Version Control with Git. DevTestOps training.

TRAINING

CENTER

<epam>

Commit, tree, blob

- `git show -s --pretty=raw d2d84`
- `git ls-tree 629eb`
- `git show abb54`

<epam>

Extras

Version Control with Git. DevTestOps training.

TRAINING

CENTER

<epam>

Extras

EXTRAS

- `git config -- global user.name “Vitali Shulha”`
- `git config -- global user.email “vitali_shulha@epam.com”`
- `git config --global core.editor "'C:/Program Files (x86)/Notepad++/notepad++.exe‘”`
- `git blame`
- `git bisect`
- `git log --pretty=oneline`
- `git log --pretty=format:"%h %s" –graph`
- `git config --global alias.last 'log -1 HEAD‘`
- `git last`
- `git log master..experiment`
- `git filter-branch --tree-filter 'rm -f passwords.txt' HEAD`
- `git rerere`
- `git submodule`

READ MORE

- **Pro Git** by Scott Chacon and Ben Straub
- **Version Control with Git** by Jon Loeliger, Matthew McCullough