

GIT - Fast Version Control System

Martin Vejmelka <vejmema1@fel.cvut.cz>

30. září 2010

1 Základní koncepty verzovacích systémů

Komponenty SCM

- Pracovní kopie (working tree)
- Informace ukládané v repozitáři

Operace SCM

Decentralizace

- Centralizovaný model SCM
- Decentralizovaný model SCM

2 Historie verzovacího systému GIT

3 Repository

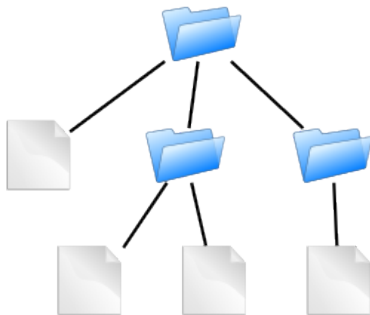
- Struktura
- Objekty

4 Používání verzovacího systému GIT

Komponenty SCM

Pracovní kopie (working tree)

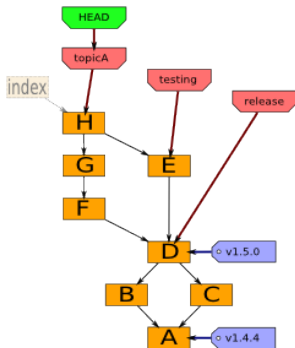
- adresáře
- soubory



Komponenty SCM

Informace ukládané v repozitáři

- soubory (files)
- commity (commits)
- následnost (ancestry) - DAG



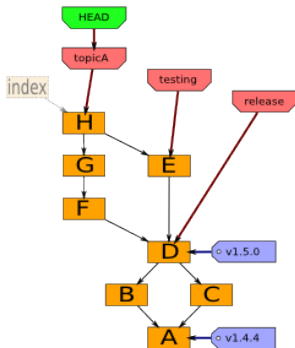
Komponenty SCM

Informace ukládané v repozitáři

- soubory (files)
- commity (commits)
- následnost (ancestry) - DAG

Reference

- tagy
- větve



Komponenty SCM

Informace ukládané v repozitáři

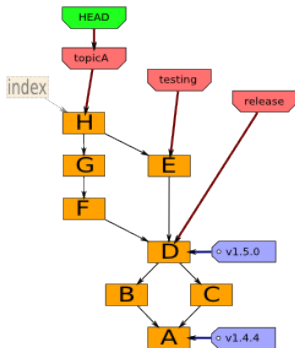
- soubory (files)
- commity (commits)
- následnost (ancestry) - DAG

Reference

- tagy
- větve

HEAD (hlava)

- aktuální checkout
- ukazuje na větev
- někdy může být v „detached“ stavu



Komponenty SCM

Informace ukládané v repozitáři

- soubory (files)
- commity (commits)
- následnost (ancestry) - DAG

Reference

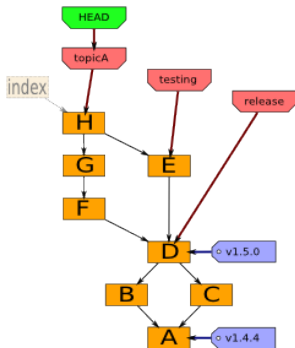
- tagy
- větve

HEAD (hlava)

- aktuální checkout
- ukazuje na větev
- někdy může být v „detached“ stavu

Index

- „staging area“
- co *má* být commitováno



Základní operace

- init - inicializace repozitáře
- checkout - získá pracovní kopie
- switch branch - přepínání mezi větvemi

Operace SCM

Základní operace

- init - inicializace repozitáře
- checkout - získání pracovní kopie
- switch branch - přepínání mezi větvemi

Modifikace repository

- add, delete, rename
- commit

Operace SCM

Základní operace

- init - inicializace repozitáře
- checkout - získá pracovní kopie
- switch branch - přepínání mezi větvemi

Modifikace repository

- add, delete, rename
- commit

Získání informací

- status
- diff
- log

Základní operace

- init - inicializace repozitáře
- checkout - získání pracovní kopie
- switch branch - přepínání mezi větvemi

Modifikace repository

- add, delete, rename
- commit

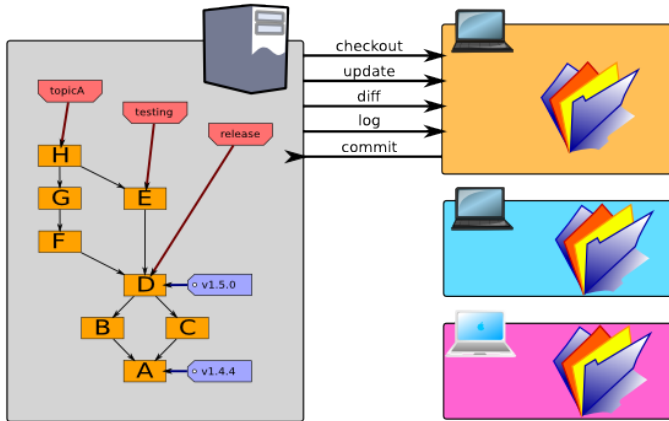
Získání informací

- status
- diff
- log

Práce s referencemi

- tag - přidání/odebrání tagu
- branch - vytvoření/modifikace/odstranění větve

Centralizovaný model SCM

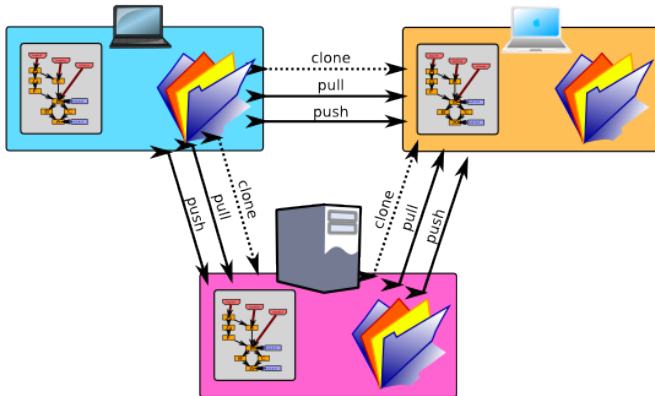


- operace vyžadují přítomnost **serveru**
 - single point of failure
 - bottleneck

Nové operace pro decentralizované SCM

- clone
- pull, fetch
- push

Decentralizovaný model SCM



- kdokoliv může být serverem

K čemu je decentralizace dobrá?

- neintrusivní micro-commity
- operace mimo síť (detached operation)
- neexistuje „single point of failure“
- zálohy jsou triviální

Historie verzovacího systému GIT

- 2002
 - Linus Torvalds používá BitKeeper pro verzování Linuxu
 - BitKeeper se zlepšuje
 - vývoj Linuxu je škálovatelnější

Historie verzovacího systému GIT

- 2002
 - Linus Torvalds používá BitKeeper pro verzování Linuxu
 - BitKeeper se zlepšuje
 - vývoj Linuxu je škálovatelnější
- 6. Dubna 2005
 - BitMover ruší svobodnou licenci
 - Linus píše vlastní SCM → GIT

Historie verzovacího systému GIT

- 2002
 - Linus Torvalds používá BitKeeper pro verzování Linuxu
 - BitKeeper se zlepšuje
 - vývoj Linuxu je škálovatelnější
- 6. Dubna 2005
 - BitMover ruší svobodnou licenci
 - Linus píše vlastní SCM → GIT
- 18. Dubna 2005
 - GIT zvládá merge

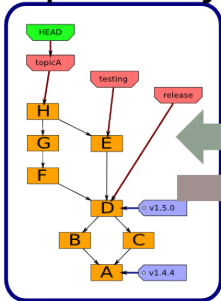
Historie verzovacího systému GIT

- 2002
 - Linus Torvalds používá BitKeeper pro verzování Linuxu
 - BitKeeper se zlepšuje
 - vývoj Linuxu je škálovatelnější
- 6. Dubna 2005
 - BitMover ruší svobodnou licenci
 - Linus píše vlastní SCM → GIT
- 18. Dubna 2005
 - GIT zvládá merge
- 16. Června 2005
 - GIT je oficiálně použit pro verzování Linuxu

Historie verzovacího systému GIT

- 2002
 - Linus Torvalds používá BitKeeper pro verzování Linuxu
 - BitKeeper se zlepšuje
 - vývoj Linuxu je škálovatelnější
- 6. Dubna 2005
 - BitMover ruší svobodnou licenci
 - Linus píše vlastní SCM → GIT
- 18. Dubna 2005
 - GIT zvládá merge
- 16. Června 2005
 - GIT je oficiálně použit pro verzování Linuxu
- 14. února 2007
 - vydán GIT 1.5.0
 - pokus o rozšíření GITu tak, aby se jednalo o široce použitelný nástroj

repository



index

```
100644 20b024 0 bar
100644 1d52a6 0 baz
100644 20b024 0 sub/fiz
100644 43dbe0 0 sub/foo
```

work tree

```
-- bar
|-- baz
|-- sub
|   |-- fiz
|   -- foo
```

Struktura

```
.git
|-- HEAD                current checkout reference
|-- config              repo private config
|-- description         repo description
|-- hooks
|   '-- ...            hooking scripts
|-- index              changes to commit
|-- info
|   |-- exclude        repo private
|   '-- refs           refs?
|-- logs
|   '-- ...            "reflog" data
|-- objects
|   |-- XX
|   |   '-- ...       loose objects
|   |-- info
|   |   '-- packs     info about packs
|   '-- pack
|       '-- ...       packs and indexes
'-- refs
    |-- heads
    |   '-- master     master branch
    '-- tags
        '-- ...       tags
```

Objekty

```
.git/objects
|-- 23
|   '-- d4bd826aba9e29aaace9411cc175b784edc399
|-- 76
|   '-- 49f82d40a98b1ba59057798e47aab2a99a11d3
|-- c4
|   '-- aaefaa8a48ad4ad379dc1002b78f1a3e4ceabc
|-- e7
|   '-- 4be61128eef713459ca4e32398d689fe80864e
|-- info
|   '-- packs
'-- pack
    |-- pack-b7b026b1a0b0f193db9dea0b0d7367d25d3a68cc.idx
    '-- pack-b7b026b1a0b0f193db9dea0b0d7367d25d3a68cc.pack
```

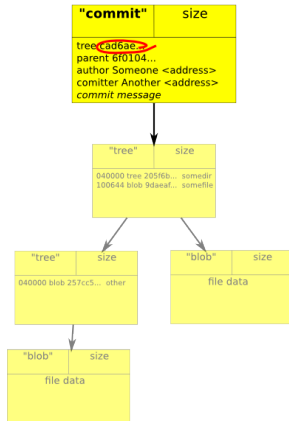
Objekty

4 typy:

- bloby
- stromy
- commity
- tagy

vlastnosti objektů:

- jsou adresovány obsahem (sha1)
- jsou neměnné (jakmile jsou jednou vytvořeny, již nejsou dále modifikovány)



Používání verzovacího systému GIT

Rozhraní systému:

- command line client
- gui rozhraní
- podpora na windows → msysgit
- gitk, gitg, tig, qgit. . .
- webové rozhraní → gitweb

Kde získat nápovědu:

- oficiální web systému GIT (adresa dále v referencích)
- git help (v podstatě manuálová stránka)
- . . .

Typický scénář použití (1/3)

```
# vytvoření čistého repozitáře
git init

# nebo naklonujeme existující repozitář
git clone git://someserver/somepath.git

# --- provedeme změny v repozitáři

# zjistíme, co všechno se změnilo
git status

# přidáme do indexu změněné soubory
git add somefile.c
git add src/*.c

# commitujeme změny
git commit -m 'Added some new C files.'
```

Typický scénář použití (2/3)

```
# nyní chceme vyvíjet experimentální feature
# vytvoříme novou větev a přepneme se do ní
git branch experiment
git checkout experiment

# nebo můžeme totéž udělat pomocí jediného příkazu
git checkout -b experiment

# --- provedeme nějakou práci ve větvi experiment (a commitujeme ji)

# následně se vrátíme do větve master
git checkout master





# experiment byl úspěšný - změny z experimentální větve
# chceme mít v hlavní vývojové větvi
git merge experiment
```

Typický scénář použití (3/3)

```
# podíváme se, jaké jsou v repozitáři commity  
git log
```

```
# podíváme se na rozdíly oproti předchozí verzi  
git diff HEAD^ HEAD
```

Reference

-  oficiální stránky verzovacího systému GIT, <http://git-scm.com/>
-  informace o verzovacím systému git na Wikipedii,
http://en.wikipedia.org/wiki/Git_%28software%29
-  referenční příručka systému GIT, <http://gitref.org/>
-  prezentace „Git the basics“, Bart Trojanowski,
[git://git.jukie.net/intro-to-git.git/](http://git.jukie.net/intro-to-git.git/)