

Sisteme de versionare software

Mihai Angheluță



Ce este un sistem de versionare?

De ce este necesar?

Instrument de dezvoltare software utilizat în gestionarea multiplelor versiuni ale fișierelor și dependentelor unei aplicații, înregistrând toate stările acestora, inclusiv modificări, autori și comentarii privind fiecare modificare.



Lucrul în echipă - permite colaborarea prin distribuirea modificărilor



Securitate - repository securizat, backup



Istoria proiectului - permite revizuirea stărilor anterioare ale proiectului



Integrare cu project trackers - modificările și comentariile vor apărea în tracker

Exemple de sisteme de versionare

- Apache Subversion (SVN) -
<https://subversion.apache.org/>
- CVS - <http://www.nongnu.org/cvs/>
- Git - <https://git-scm.com/>
- Mercurial - <https://www.mercurial-scm.org/>
- Perforce - <http://www.perforce.com/>
- Team Foundation Version Control -
<https://visualstudio.microsoft.com/>

Concepțe, cuvinte cheie

/thoughtworks

4 4 nodes/ba780c3a.39e07/func

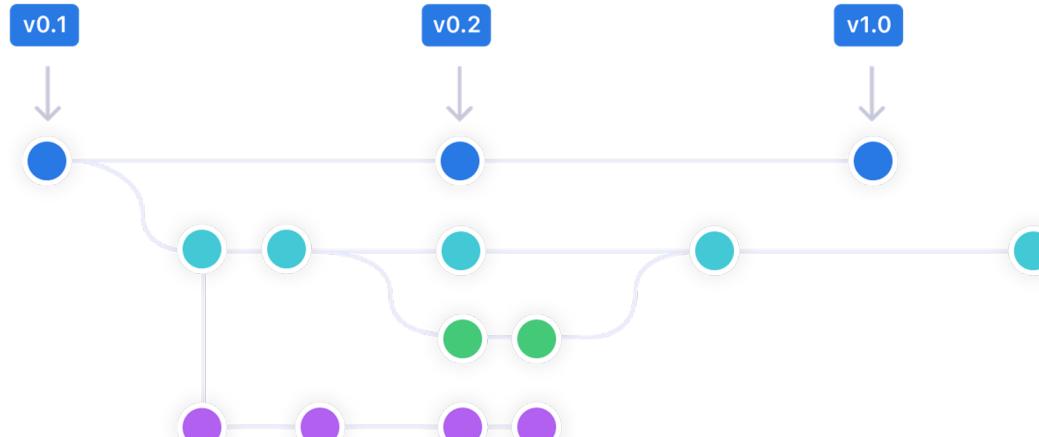
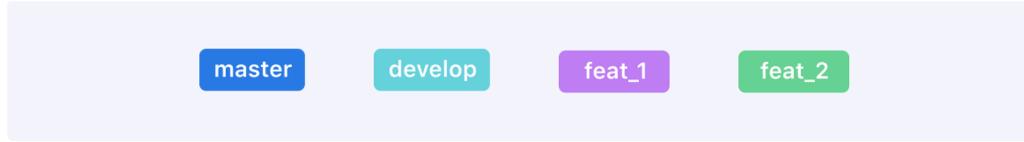
...	...	@@ -1,4 +1,4 @@
1		-if (true) {
2		- console.log('Hello world.')
	1	+if (false) {
	2	+ console.log('Goodbye world.')
3	3	}
4	4	return msg; @@

Commit

modificări efectuate asupra unor fișiere, publicate în repository

Modificare unitară a fișierelor proiectului, ce de regulă păstrează proiectul într-o stare consistentă

Commit-ul este atomic, reversibil și pe lângă modificările propriu-zise conține și informații despre autor, timestamp și comentarii.



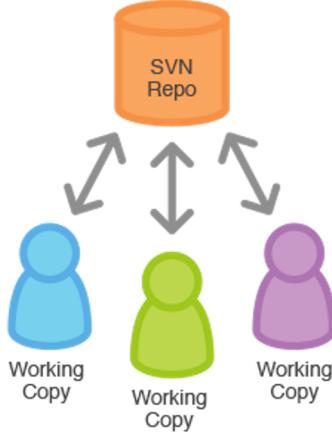
Branch

linie de dezvoltare a proiectului, ce conține o istorie proprie

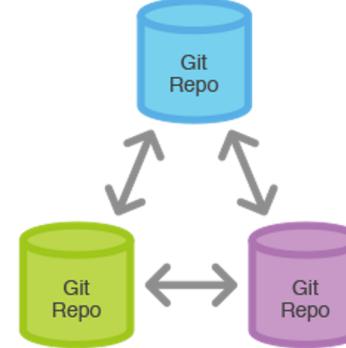
Operațiunile de fork și merge reprezintă interacțiunile dintre branch-uri

Trunk sau master prin convenție este branch-ul de bază al proiectului

Central-Repo-to-Working-Copy
Collaboration



Repo-to-Repo
Collaboration



Repository

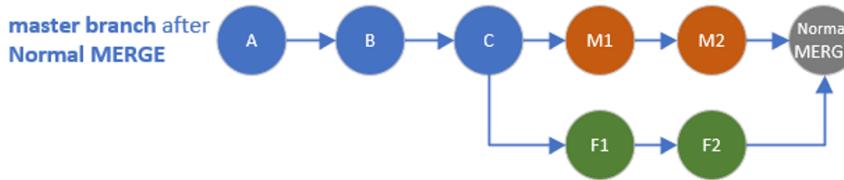
Server de fișiere (bază de date) unde sunt stocate datele proiectului software

Poate funcționa independent sau în colaborare cu alte Repository - clone

Conține întreaga istorie a proiectului, pe fiecare linie de dezvoltare (branch), modificări (commit) și contribuitori.

Alte concepte

- Revision - versiune a proiectului, ca urmare a aplicării unui commit
- Working directory - fișierele vizibile în sistemul de operare / IDE, ce corespund unui Revision curent, din Repository
- Merge - operație de combinare a modificărilor din branch-uri separate
- Checkout / fetch - operațiunea de descărcare modificări de pe repository
- Push - încărcarea modificărilor de pe un repository local pe un repository remote.



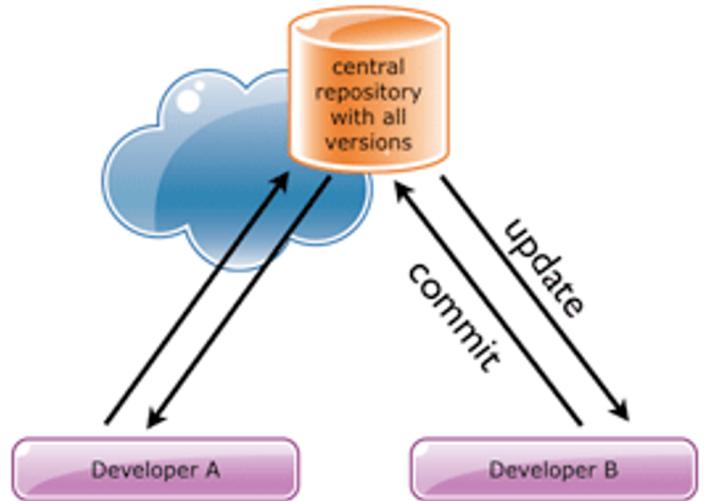
Merging branches

Unificarea liniilor de dezvoltare

Squash păstrează modificările dar omite commit-urile individuale din istoric

Rebase mută tot branch-ul astfel încât să înceapă după HEAD a branch-ului destinație

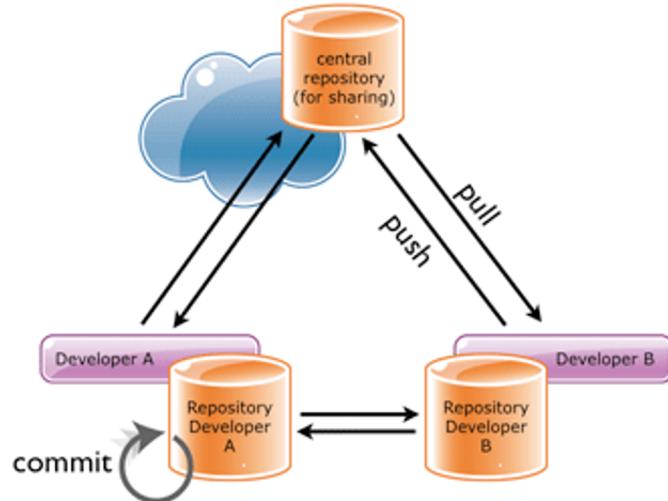
Arhitecturi ale sistemelor de versionare



Arhitectura centralizată
(exemplu SVN)



Arhitectura distribuită
(exemplu Git)





/thoughtworks

git – Source Control Management

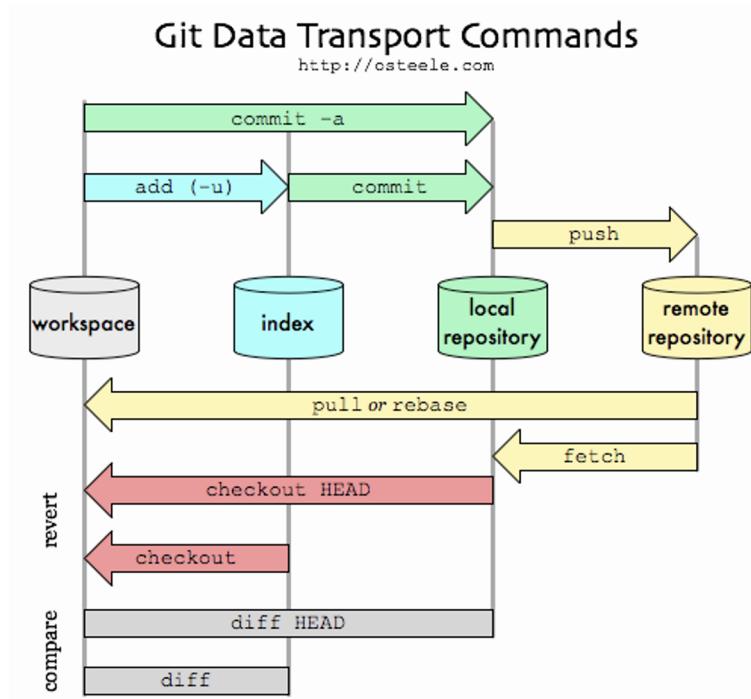
Sistem open source, descentralizat, dezvoltat de Linus Torvalds

- Salveaza patches (diferențe) pentru fiecare commit
- Local branches sunt folosite intensiv
- Operăriuni de merge, rebase, fork mult mai facile decât în SVN
- Repository principal poate fi schimbat
- Fork & pull request

<https://git-scm.com/doc>

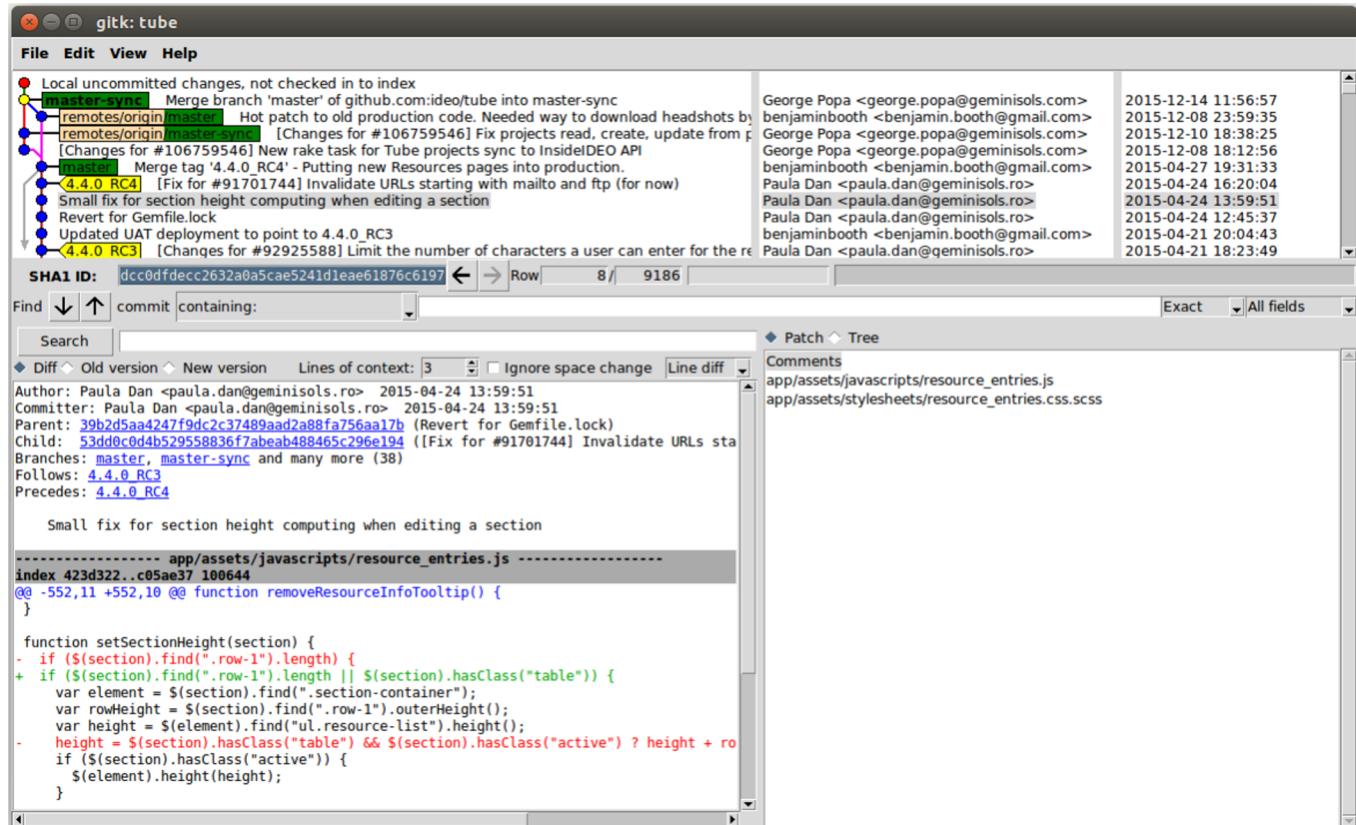
git - comenzi uzuale

- git config user.name / user.email
- git init
- git add [file1, file2...]
- git commit -m "First commit"
- git remote add origin url.git
- git push origin master
- git clone url.git
- git pull origin master
- git rebase origin master



gitk

git GUI tools



Să încercăm!

/thoughtworks



Proiect nou în GitHub

- Sunt necesare un cont pe GitHub (<https://github.com/>) și un repository nou.
- Configurarea locală a git
 - `git config --global user.name john-doe`
 - `git config --global user.email johndoe@example.com`
 - Comunicați user.name către repository Administrator pentru a vă acorda acces la proiect
- Descărcarea proiectului de test
 - `git clone git@github.com:georgpopa/computer.git`
 - `git status`

Primul commit

- Adăugarea unui fișier nou în cadrul proiectului
 - touch nume_prenume.txt
 - git status
- Urcarea fișierului nou creat pe repository-ul github.com și descărcarea fișierelor nou adăugate:
 - git add nume_prenume.txt
 - git commit -m "Upload nume_prenume.txt"
 - git pull origin master
 - git push origin master
 - git status

Conflict

/thoughtworks



Rezolvarea conflictelor

care apar inevitabil în lucrul în echipă

apar când sunt modificări simultane la aceleași linii de cod

nu există tool-uri automate care să rezolve toate conflictele

rezolvarea conflictelor implică decizii luate de echipă, specifice logicii aplicației.

The screenshot shows a merge conflict resolution interface. At the top, there's a toolbar with buttons for 'Show Conflicts Only', 'Take Incoming', 'Take Current', 'Compare', 'Accept Merge', 'Show Word Diffs', and a gear icon. Below the toolbar, a status bar indicates '1 Conflicts (1 Remaining)'. The main area is divided into two sections: 'Incoming: b1' on the left and 'Current: main' on the right. Both sections show a list of changes with line numbers (1, 2) and modification types (v1 or v2). In the 'Incoming' section, line 1 has a change 'v1 2:22'. In the 'Current' section, line 1 has a change 'v1 2:22' and line 2 has a change 'v2 2:26'. Below this, a preview window shows the resulting file content: 'Result: test.txt' with the line 'v1 2:22'. At the bottom, another editor window shows the raw conflict markers: '1 v1 2:22', '2 <<<<< HEAD', '3 v2 2:26', '4 =====', '5 v2 2:25', '6 >>>> first conflict commit', and '7'. A status bar at the bottom right says 'No issues found'.

Primul conflict

- Modificarea acelaiași fișier în două branch-uri diferite
- commit apoi pull de pe <https://github.com>
 - git add test.txt
 - git commit -m "fixme i'm broken"
 - git pull origin master
 - ### CONFLICT ### TODO: resolve!
 - git add test.txt
 - git commit -m "you're fixed, get over it"
 - git push origin master

Operații cu branch-uri

Lucrând cu git se folosesc, de regulă,
multiple branch-uri locale

Comenzi uzuale

- git checkout -b new_feature
- git rebase master
- git checkout master
- git merge new_feature
- git push origin master

Utilizarea pull requests



Multumesc

Mihai Angheluță

Lead Consultant

mihai.angheluta@thoughtworks.com

