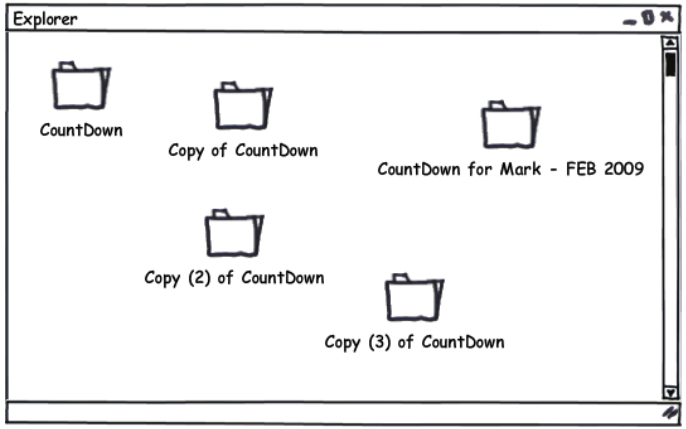
# Version Control System using Git

“

- Maintaining group Projects
- Patches are mostly sent via email
- Difficult to roll back
- Almost impossible to maintain if the number of people working in the project is large

- Testing new unstable features

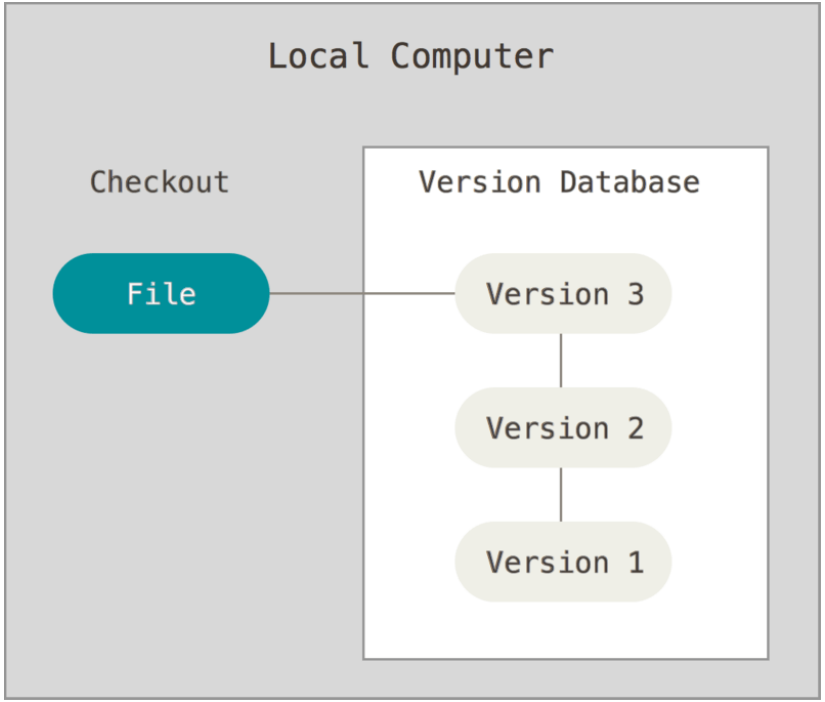


# 2.

## Version Control System

# Version Control: What is it?

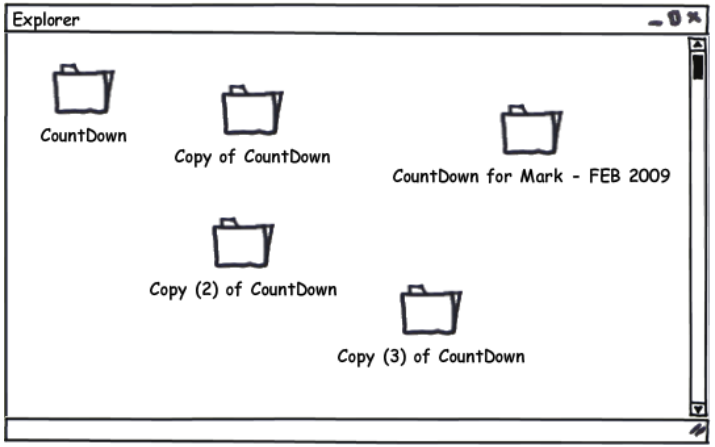
- A method for recalling versions of a codebase
- Keeping a record of changes
- Who did what and when in the system
- Save yourself when things inevitably go wrong



# Version Control: Why?

## Individual

- Back-ups of the project
- Create a “checkpoint” in the project at any stage: Fearlessly modify code
- Tagging: Mark certain point in time
- Branching: Release versions and continue development



# Version Control: Why?

Team

- Everything in “Individual”
- Allow multiple developer to work on the same codebase
- Merge changes across same files: handle conflicts
- Check who made which change: blame/praise

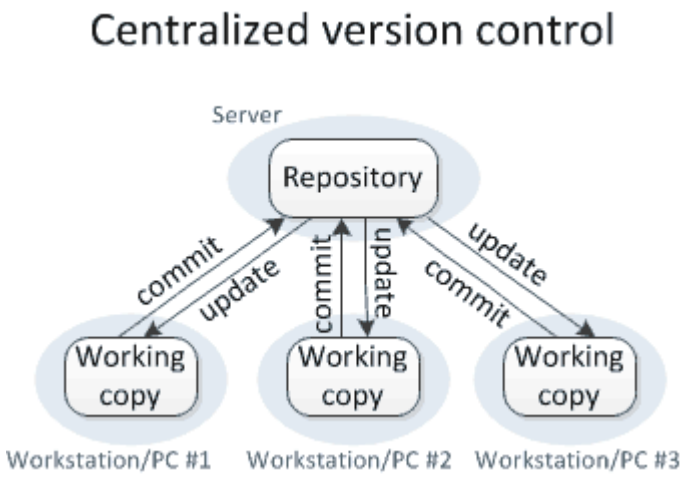
# Version Control: Types

- Centralised VCS
- Distributed VCS

# Centralised VCS

- A **single** authoritative data source (repository)
- Check-outs and check-ins are done with reference to this central repository

# Centralised VCS



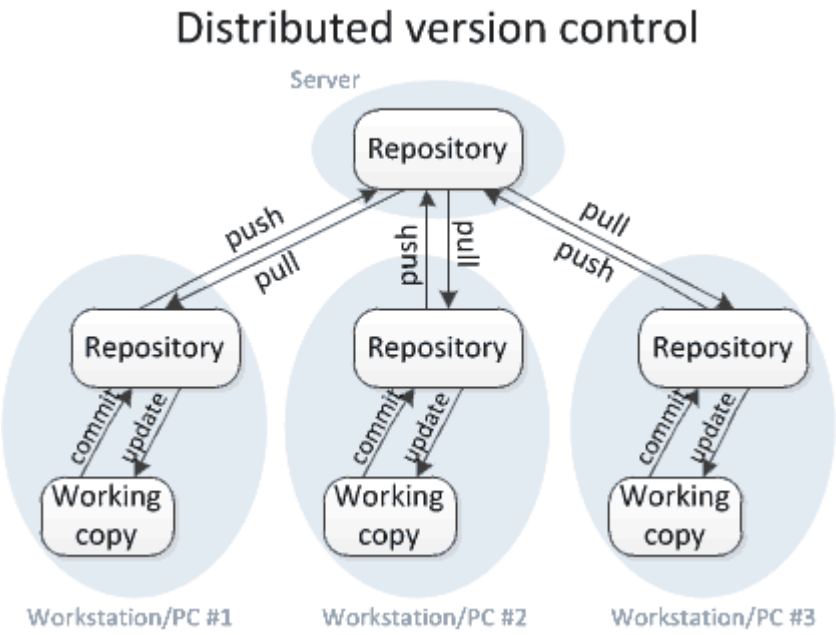
# Centralised VCS

Examples:

- Concurrent Version System (CVS)
- Subversion (SVN)

- No single repository is authoritative
- Data can be checked in and out from any repository

# Distributed VCS




# Distributed VCS

Examples

- Git
- Mercurial

# Distributed VCS

# 3.

## Git

--everything-is-local

- Free, open source
- Fully distributed
- Handle small files very effectively
- Tracks contents, not files
- Data = Snapshot
- No network
- Three stages

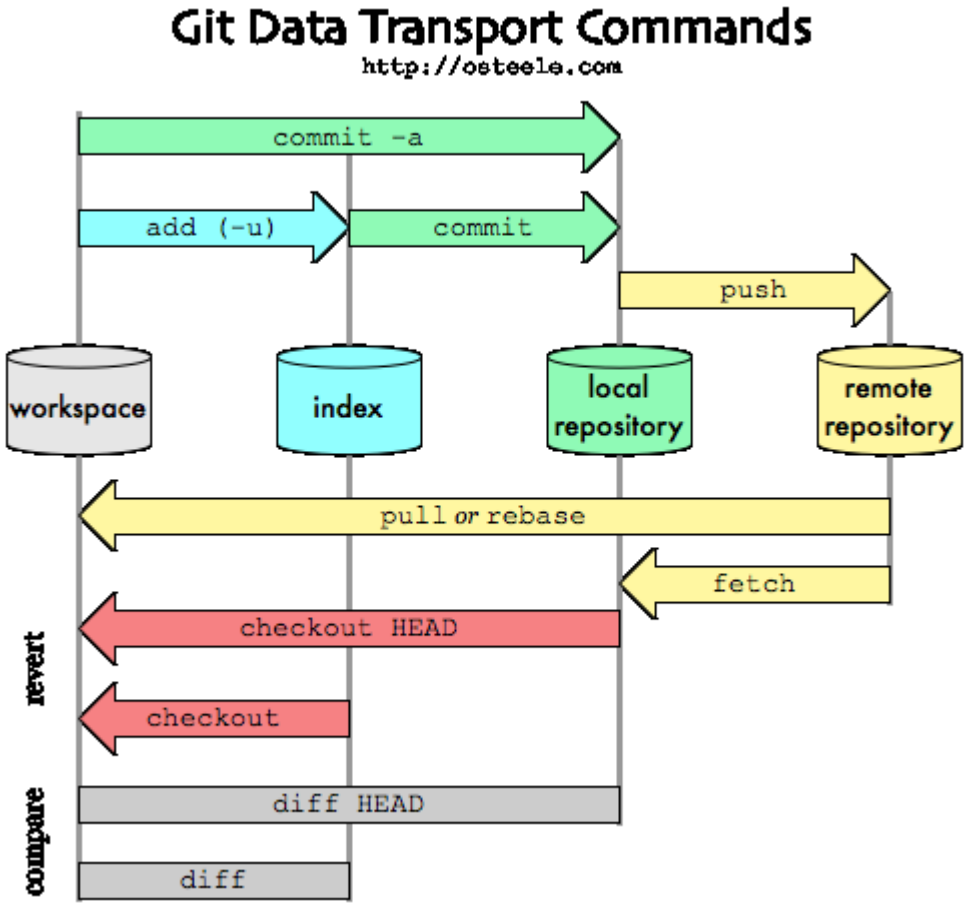
- Created by Linus Torvalds in less than 2 weeks
- Currently maintained by Junio C Hamano

# Git: Stages

Three stages:

- Working directory
- Staging directory
- Git directory (repository)

# Git: Development



# Git: Development

Setup

- git init
- git clone <remote-url>

# Configuring GIT

## STEP BY STEP

- git config –global user.name "FirstName LastName"
- git config –global user.email "yourname@maine.edu"
- git config –global color.ui "auto"
  - Mac/linux
    - git config – global core.editor "nano –w"
  - Windows
    - git config –global core.editor "'c:/program files (x86)/Notepad++/notepad++.exe' –multiInst –notabbar –nosession noPlugin"
- git config --list

# GIT

## STEP BY STEP

- Go to folder for git repository
- Clone remote repository
- git clone https://github.com/npd2020/electronics.git

Go to folder electronics

Create folder "*first_second your name* (without space)"

Greate (or copy) file in out folder

git add .

git commit –m "*name of commit*"

**git pull --rebase**

**git push** **//user: npd2020** **paswd:** standardmodel123

git status // get status of repository

git log // see history of commit

Wifi: stone , password: 31415926

git status // get status of repository

- git log //  see history of commit

# Git: Development

## Commit logs

git log

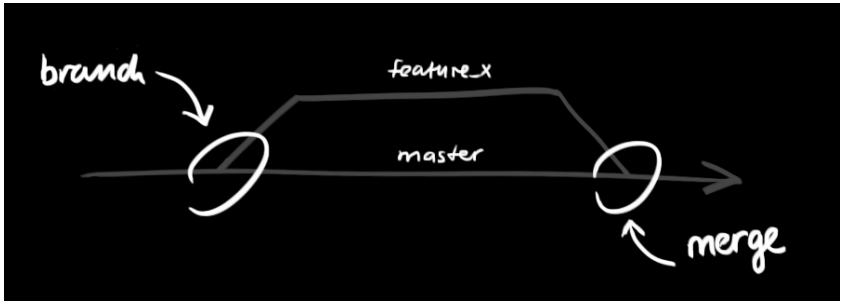
File Edit View Search Terminal Help

```
commit 115a041a1773a15968a059f8eef610a2b18d4cca
Author: mouri11 <rohit.das950@gmail.com>
Date:   Sun May 28 15:19:21 2017 +0530

    Same as before

    Signed-off-by: mouri11 <rohit.das950@gmail.com>

commit 65e6907394f3b74ce854d283b29dccae4b28aedf
Author: mouri11 <rohit.das950@gmail.com>
Date:   Sun May 28 15:14:12 2017 +0530

    Penultimate changes, I guess...

    Signed-off-by: mouri11 <rohit.das950@gmail.com>

commit b1c1027b8fac7d1c65e63da329838be172f496ce
Author: mouri11 <rohit.das950@gmail.com>
Date:   Sun May 28 14:28:53 2017 +0530

    Still changing

    Signed-off-by: mouri11 <rohit.das950@gmail.com>
:
```

# Git: Development

Branches

- git checkout –b <branch-name>



# Git: Development

View changes

- git diff

# Git: Development

## View changes

git diff

git diff 80x24

```
diff --git a/README.md b/README.md
index 24ff7ab..f121786 100644
--- a/README.md
+++ b/README.md
@@ -1 +1,4 @@
 LaTeX files
+
+Made by:
+* Rohit Das <rohit.das950@gmail.com>
(END)
```

# Git: Development

Update staging area

- git add <files>

Add file **contents** to the index

# Git: Development

Create “snapshots” of your codebase

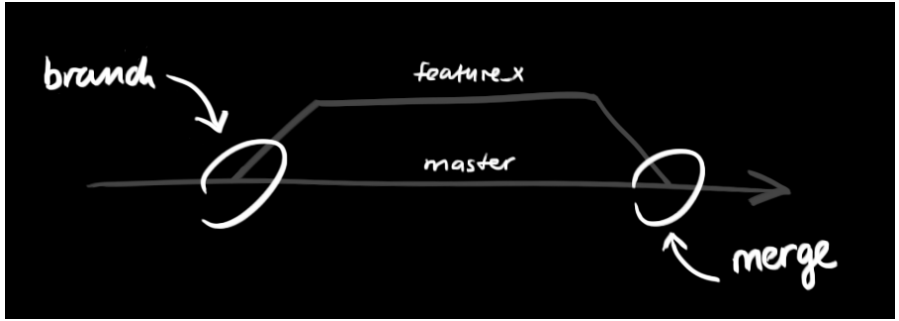
- git commit

Records changes to the repository

# Git: Development

Merge other branches

- git merge



# Git: Development

Make patches

- git format-patch --stdout > fix.patch

Patch created as “fix.patch”
Prepare patches for email submission
Send patch via mail

# Git: Development

## Make patches

```
git format-patch master --stdout
git format-patch master --stdout 80x24
From 027c42c2436f5c06077619e2338d82d2baacd526 Mon Sep 17 00:00:00 2001
From: mouri11 <rohit.das950@gmail.com>
Date: Mon, 29 May 2017 11:17:58 +0530
Subject: [PATCH] readme

Signed-off-by: mouri11 <rohit.das950@gmail.com>
---
 README.md | 3 +++
 1 file changed, 3 insertions(+)

diff --git a/README.md b/README.md
index 24ff7ab..f121786 100644
--- a/README.md
+++ b/README.md
@@ -1 +1,4 @@
 LaTeX files
+
+Made by:
+* Rohit Das <rohit.das950@gmail.com>
--
2.7.4

(END)
```

# Git: Development

Applying patches

- git apply < fix.patch

Applies changes from the patch

# Result?

- Much efficient workflow
- Creating and merging branches are very easy and fast

# Result?

The development process of the Linux kernel is maintained using Git

The Linux kernel development process has:

- Over 2000 individual contributors per year
- Grows by nearly 300,000 lines per year