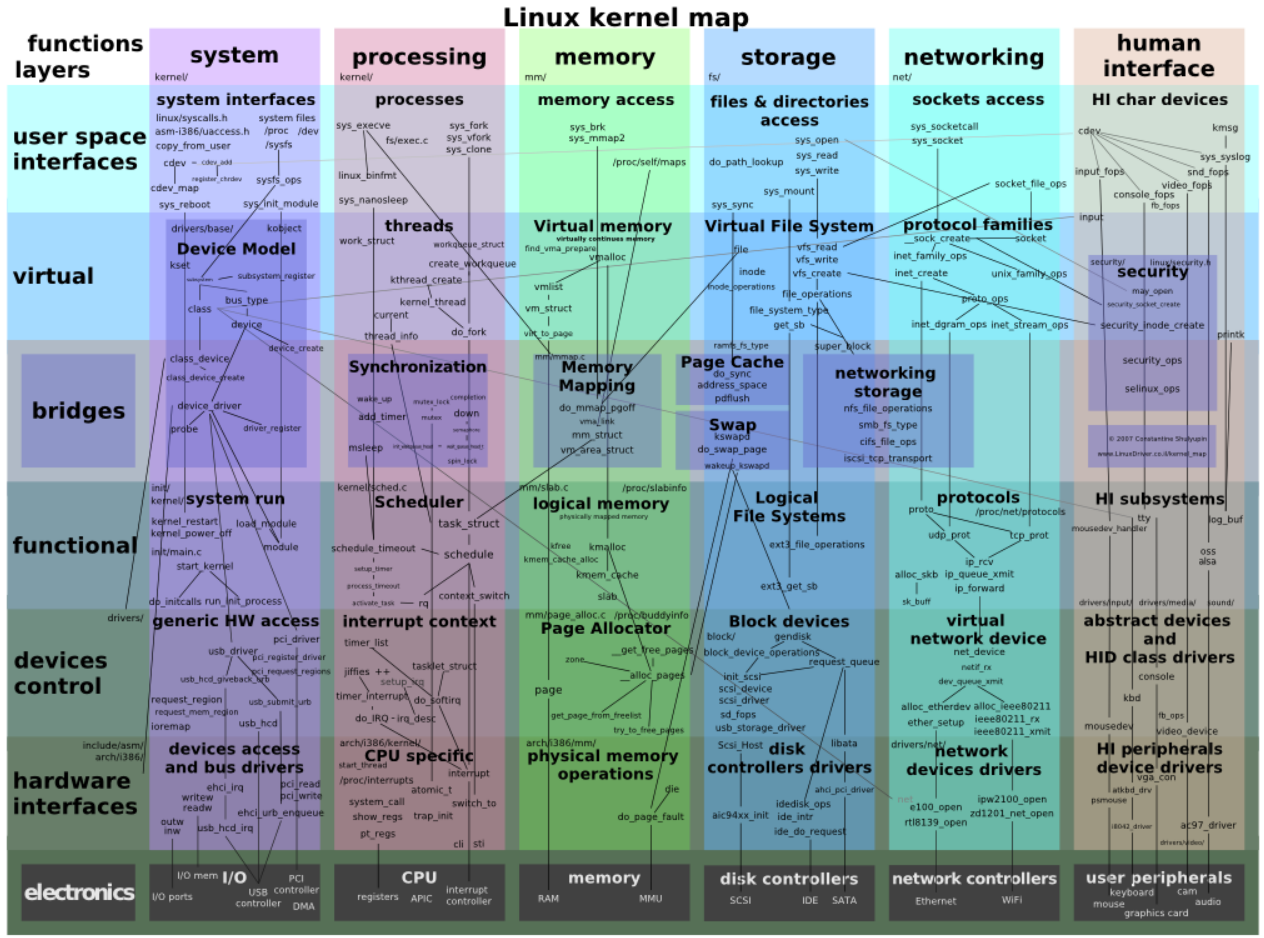
Git : *the stupid content tracker*

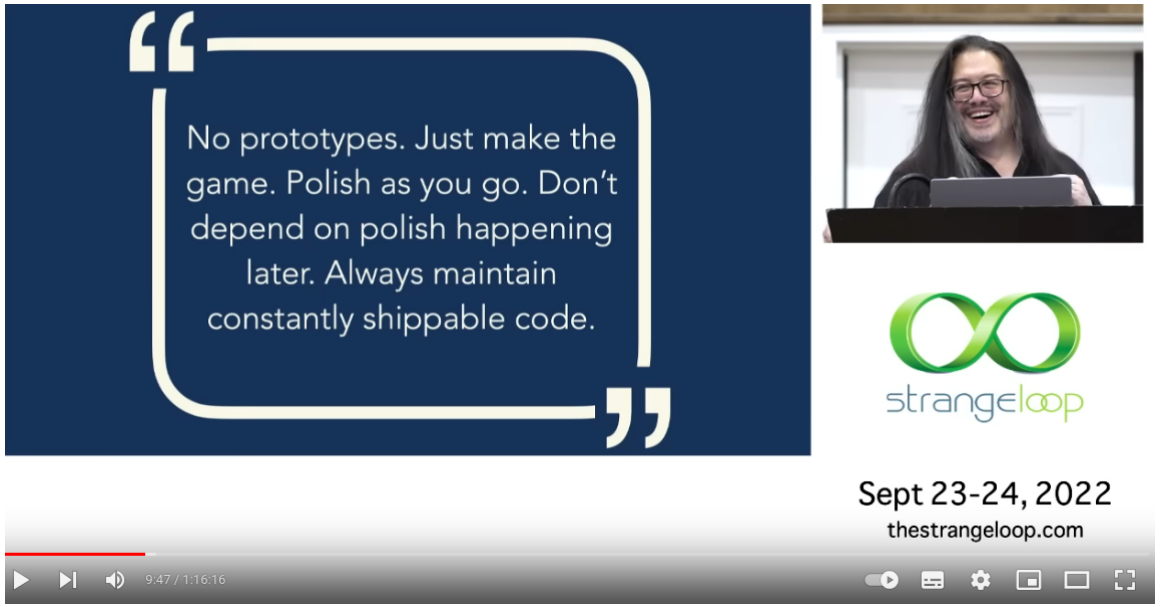
> Source: un diagramme de l'architecture du noyau Linux



La gestion de versions

Gestion de version: stocker un ensemble de fichiers en conservant la *chronologie* de toutes les *modifications* qui ont été effectuées dessus.

Existe depuis les années 70 (Logiciel *Librarian*, colonnes virtuelles sur cartes à perforer informatiques, Pacbase, etc.)



> "The Early Days of id Software: Programming Principles" by John Romero (Strange Loop 2022)

Pas toujours (*eu*) besoin d'un logiciel (recommandé aujourd'hui !)

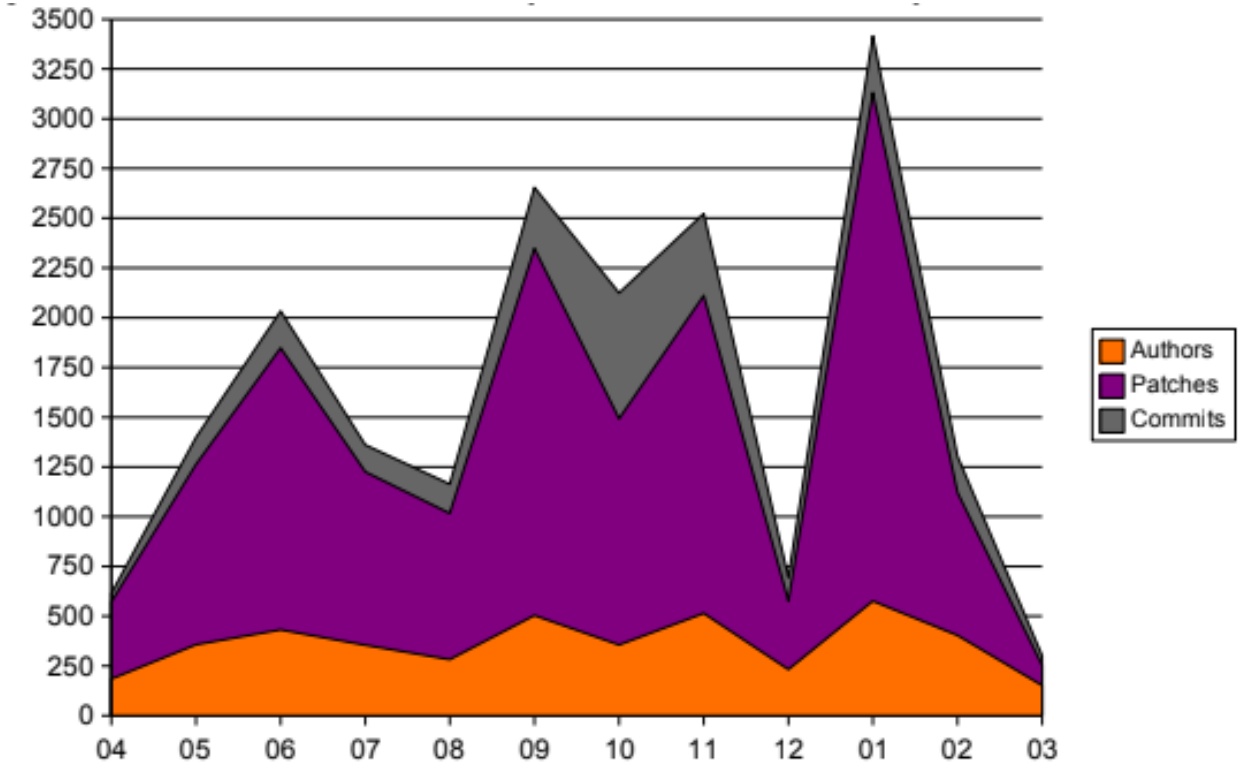
Git

- logiciel de gestion de versions (de code source) **décentralisé**
- Crée en 2005 par Linus Torvald (gauche) initialement pour la gestion du code source du noyau Linux (à la place de Bitkeeper)
- Maintenu depuis plus de 16 ans par Junio C Hamano (successeur de Torvalds, à droite) et plus de 1500 contributeurs


- Conçu pour:
  - Environnements de développement **distribués** et parallèles
  - **Grands projets**, **beaucoup de petits changements**

Pourquoi Git ?

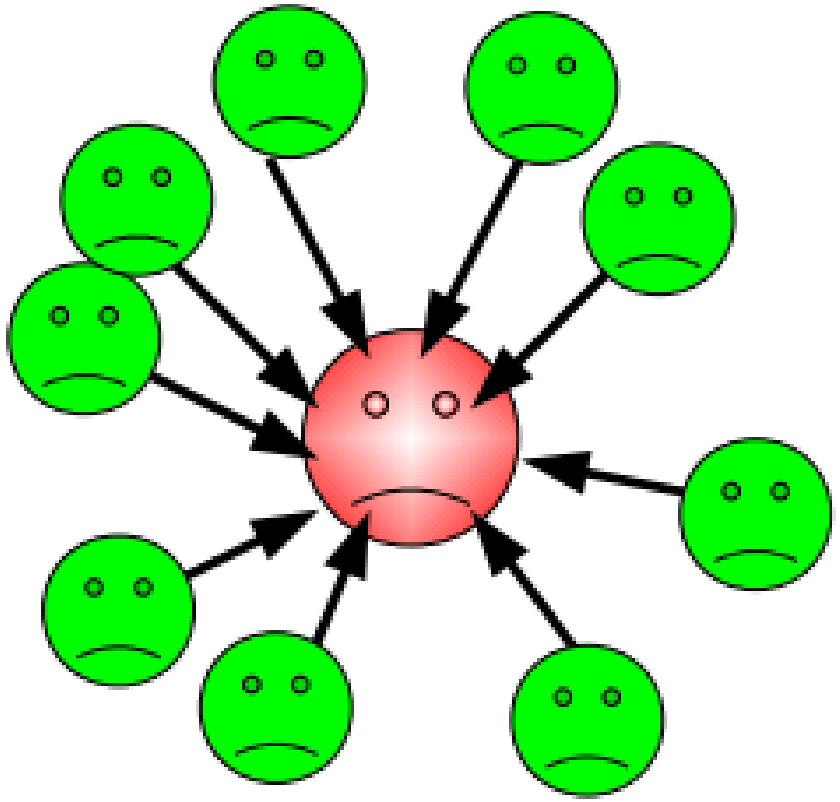
> Graphique: activité du code source du noyau Linux en 2006 sur une période de 1 an



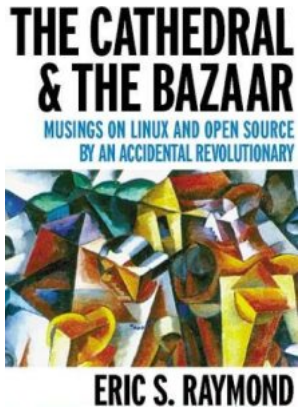
Pourquoi Git ? Problèmes des systèmes centralisés (ex: CVS)

Alternative décentralisée à Bitkeeper (licence commerciale)

- non adapté au cycle developpement/test/release dans un environnement *concurrent*
- un seul endroit gère toutes les données :
  - un noeud compte plus que les autres
  - gestion des accès et confiance, necessite un accès au noeud (*commit acces*)
  - gestion et merge des branches compliquées
  - backup
  - etc.

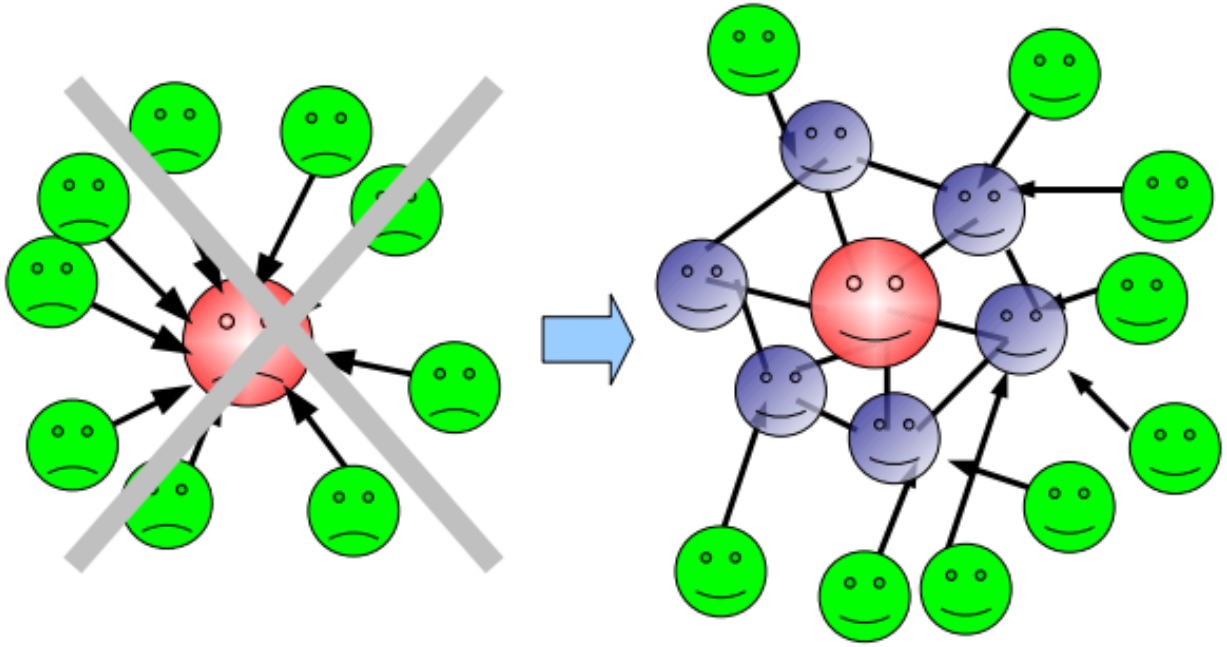


Pourquoi Git ? Bénéfices des sytèmes décentralisés et dynamique open-source



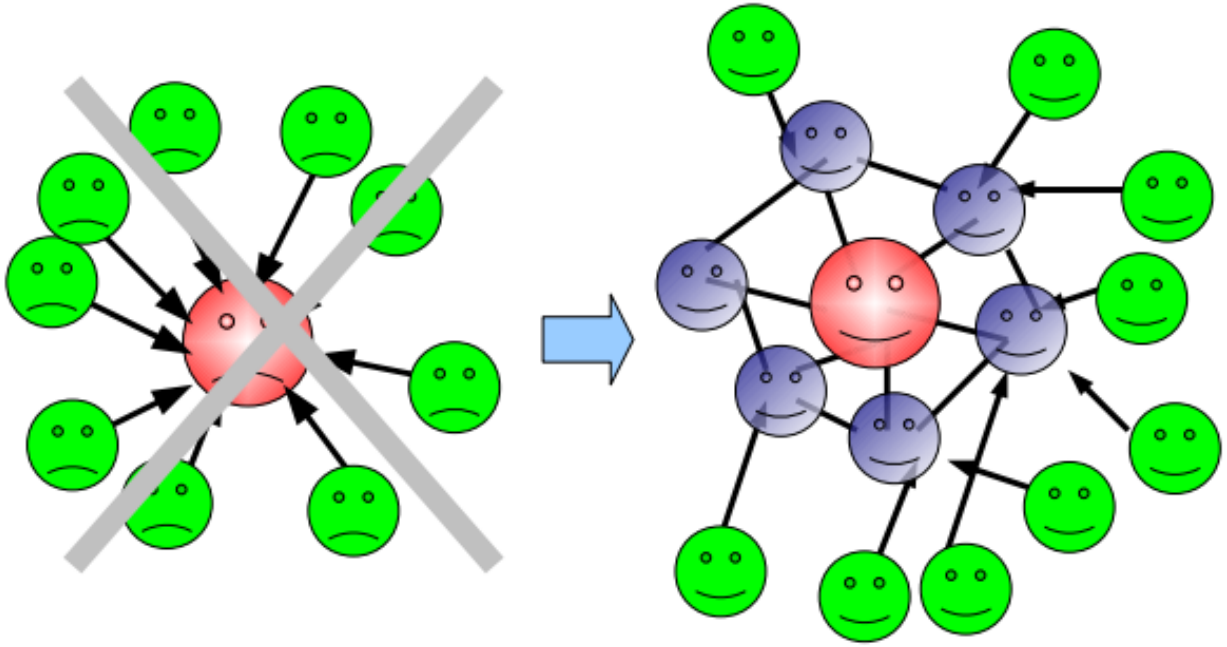
Le *bazar* : la *dynamique* des projets open-source

> Référence: La Cathédrale et le Bazar (1999), Eric S. Raymond (design DNS, implémentation/design de la couche SMTP, Fetchmail, open-source vs free software, etc.)



Pourquoi Git ? Bénéfices des sytèmes décentralisés et dynamique open-source

- chacun·e à *sa* version (ses branches)
- *network of trust* et *leader* (dynamique open-source)
- supporte livrer *tôt* et *souvent* (*ship early, ship often*)
- pas de politique d'*acces commit*



Pourquoi Git ? Bénéfices des sytèmes décentralisés et dynamique open-source

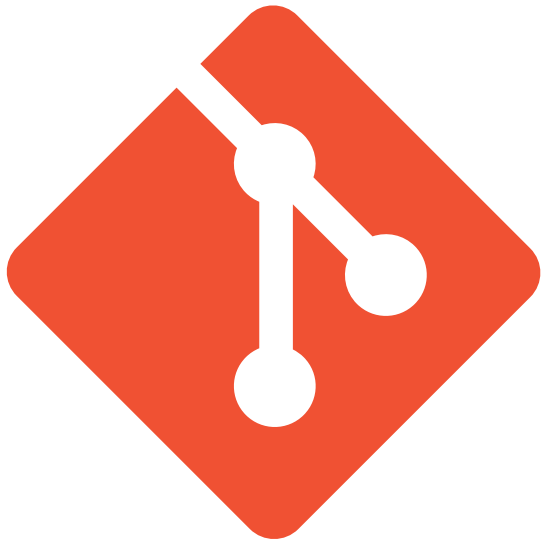
Pas de dépôt central par design, **mais par convention sociale** (réseau de confiance)

> Source: slide (et autres schémas) tirée d'une présentation de Junio Hamano donnée à Tokyo en 2006

Git aujourd'hui

- Maintenu et évolue depuis plus de 15 ans
- garantit que ce qui est retrouvé correspond exactement à ce qui y est mis
- performant, sécurisé, fiable (distribué), logiciel libre (Licence GPL)
- utilisé par grands projets open source: Noyau linux (bien sûr), Tensor Flow, React Native, Ansible, PostgreSQL, etc.
- utilisé par des grandes entreprises: Adobe, Dell, Facebook, etc.

> Le système de gestion de version à utiliser, *point barre*.



À regarder

Tech Talk (Google): Linus Torvalds on git, conférence donnée par Linus Torvald chez Google en 2007, deux ans après la création de Git. Un *classique*

Références

- Tech Talk: Linus Torvalds on git, conférence de Linus Torval sur la création de git et sa vision de systèmes de gestion de code source. Cette conférence est un classique, à voir.
- The Cathedral and the Bazaar, de Eric S. Raymond. Disponible gratuitement en ligne. Version traduite partiellement en français ici, par Sébastien Blondeel
- Celebrating 15 years of Git: An interview with Git maintainer Junio Hamano