[](https://git-scm.com/)--distributed-even-if-your-workflow-isnt

Haut du formulaire



Bas du formulaire

* [**About**](https://git-scm.com/about)
* [**Documentation**](https://git-scm.com/doc)
  + [Reference](https://git-scm.com/docs)
  + [Book](https://git-scm.com/book)
  + [Videos](https://git-scm.com/videos)
  + [External Links](https://git-scm.com/doc/ext)
* [**Downloads**](https://git-scm.com/downloads)
* [**Community**](https://git-scm.com/community)

This book is available in [English](https://git-scm.com/book/en).

Full translation available in

|  |
| --- |
| [български език](https://git-scm.com/book/bg), |
| [Español](https://git-scm.com/book/es), |
| [Français](https://git-scm.com/book/fr), |
| [Ελληνικά](https://git-scm.com/book/gr), |
| [日本語](https://git-scm.com/book/ja), |
| [한국어](https://git-scm.com/book/ko), |
| [Nederlands](https://git-scm.com/book/nl), |
| [Русский](https://git-scm.com/book/ru), |
| [Slovenščina](https://git-scm.com/book/sl), |
| [Tagalog](https://git-scm.com/book/tl), |
| [Українська](https://git-scm.com/book/uk) |
| [简体中文](https://git-scm.com/book/zh), |

Partial translations available in

|  |
| --- |
| [Čeština](https://git-scm.com/book/cs), |
| [Македонски](https://git-scm.com/book/mk), |
| [Polski](https://git-scm.com/book/pl), |
| [Српски](https://git-scm.com/book/sr), |
| [Ўзбекча](https://git-scm.com/book/uz), |
| [繁體中文](https://git-scm.com/book/zh-tw), |

Translations started for

|  |
| --- |
| [Беларуская](https://git-scm.com/book/be), |
| [Deutsch](https://git-scm.com/book/de), |
| [فارسی](https://git-scm.com/book/fa), |
| [Indonesian](https://git-scm.com/book/id), |
| [Italiano](https://git-scm.com/book/it), |
| [Bahasa Melayu](https://git-scm.com/book/ms), |
| [Português (Brasil)](https://git-scm.com/book/pt-br), |
| [Português (Portugal)](https://git-scm.com/book/pt-pt), |
| [Türkçe](https://git-scm.com/book/tr). |

The source of this book is [hosted on GitHub.](https://github.com/progit/progit2)  
Patches, suggestions and comments are welcome.

[**Chapters ▾**](https://git-scm.com/book/fr/v2/D%C3%A9marrage-rapide-%C3%80-propos-de-la-gestion-de-version)2nd Edition

1.1 Démarrage rapide - À propos de la gestion de version

Ce chapitre traite du démarrage rapide avec Git. Nous commencerons par expliquer les bases de la gestion de version, puis nous parlerons de l’installation de Git sur votre système et finalement du paramétrage pour commencer à l’utiliser. À la fin de ce chapitre vous devriez en savoir assez pour comprendre pourquoi on parle beaucoup de Git, pourquoi vous devriez l’utiliser et vous devriez en avoir une installation prête à l’emploi.

**À propos de la gestion de version**

Qu’est-ce que la gestion de version et pourquoi devriez-vous vous en soucier ? Un gestionnaire de version est un système qui enregistre l’évolution d’un fichier ou d’un ensemble de fichiers au cours du temps de manière à ce qu’on puisse rappeler une version antérieure d’un fichier à tout moment. Dans les exemples de ce livre, nous utiliserons des fichiers sources de logiciel comme fichiers sous gestion de version, bien qu’en réalité on puisse l’utiliser avec pratiquement tous les types de fichiers d’un ordinateur.

Si vous êtes un dessinateur ou un développeur web, et que vous voulez conserver toutes les versions d’une image ou d’une mise en page (ce que vous souhaiteriez assurément), un système de gestion de version (VCS en anglais pour *Version Control System*) est un outil qu’il est très sage d’utiliser. Il vous permet de ramener un fichier à un état précédent, de ramener le projet complet à un état précédent, de visualiser les changements au cours du temps, de voir qui a modifié quelque chose qui pourrait causer un problème, qui a introduit un problème et quand, et plus encore. Utiliser un VCS signifie aussi généralement que si vous vous trompez ou que vous perdez des fichiers, vous pouvez facilement revenir à un état stable. De plus, vous obtenez tous ces avantages avec peu de travail additionnel.

**Les systèmes de gestion de version locaux**

La méthode courante pour la gestion de version est généralement de recopier les fichiers dans un autre répertoire (peut-être avec un nom incluant la date dans le meilleur des cas). Cette méthode est la plus courante parce que c’est la plus simple, mais c’est aussi la moins fiable. Il est facile d’oublier le répertoire dans lequel vous êtes et d’écrire accidentellement dans le mauvais fichier ou d’écraser des fichiers que vous vouliez conserver.

Pour traiter ce problème, les programmeurs ont développé il y a longtemps des VCS locaux qui utilisaient une base de données simple pour conserver les modifications d’un fichier.



Figure 1. Gestion de version locale.

Un des systèmes les plus populaires était RCS, qui est encore distribué avec de nombreux systèmes d’exploitation aujourd’hui. Même le système d’exploitation populaire Mac OS X inclut le programme rcslorsqu’on installe les outils de développement logiciel. Cet outil fonctionne en conservant des ensembles de patchs (c’est-à-dire la différence entre les fichiers) d’une version à l’autre dans un format spécial sur disque ; il peut alors restituer l’état de n’importe quel fichier à n’importe quel instant en ajoutant toutes les différences.

**Les systèmes de gestion de version centralisés**

Le problème majeur que les gens rencontrent est qu’ils ont besoin de collaborer avec des développeurs sur d’autres ordinateurs. Pour traiter ce problème, les systèmes de gestion de version centralisés (CVCS en anglais pour *Centralized Version Control Systems*) furent développés. Ces systèmes tels que CVS, Subversion, et Perforce, mettent en place un serveur central qui contient tous les fichiers sous gestion de version, et des clients qui peuvent extraire les fichiers de ce dépôt central. Pendant de nombreuses années, cela a été le standard pour la gestion de version.



Figure 2. Gestion de version centralisée.

Ce schéma offre de nombreux avantages par rapport à la gestion de version locale. Par exemple, chacun sait jusqu’à un certain point ce que tous les autres sont en train de faire sur le projet. Les administrateurs ont un contrôle fin des permissions et il est beaucoup plus facile d’administrer un CVCS que de gérer des bases de données locales.

Cependant ce système a aussi de nombreux défauts. Le plus visible est le point unique de panne que le serveur centralisé représente. Si ce serveur est en panne pendant une heure, alors durant cette heure, aucun client ne peut collaborer ou enregistrer les modifications issues de son travail. Si le disque dur du serveur central se corrompt, et s’il n’y a pas eu de sauvegarde, vous perdez absolument tout de l’historique d’un projet en dehors des sauvegardes locales que les gens auraient pu réaliser sur leurs machines locales. Les systèmes de gestion de version locaux souffrent du même problème — dès qu’on a tout l’historique d’un projet sauvegardé à un endroit unique, on prend le risque de tout perdre.

**Les systèmes de gestion de version distribués**

C’est à ce moment que les systèmes de gestion de version distribués entrent en jeu (DVCS en anglais pour *Distributed Version Control Systems*). Dans un DVCS (tel que Git, Mercurial, Bazaar ou Darcs), les clients n’extraient plus seulement la dernière version d’un fichier, mais ils dupliquent complètement le dépôt. Ainsi, si le serveur disparaît et si les systèmes collaboraient via ce serveur, n’importe quel dépôt d’un des clients peut être copié sur le serveur pour le restaurer. Chaque extraction devient une sauvegarde complète de toutes les données.



Figure 3. Gestion de version distribuée.

De plus, un grand nombre de ces systèmes gère particulièrement bien le fait d’avoir plusieurs dépôts avec lesquels travailler, vous permettant de collaborer avec différents groupes de personnes de manières différentes simultanément dans le même projet. Cela permet la mise en place de différentes chaînes de traitement qui ne sont pas réalisables avec les systèmes centralisés, tels que les modèles hiérarchiques.

[prev](https://git-scm.com/book) | [next](https://git-scm.com/book/fr/v2/D%C3%A9marrage-rapide-Une-rapide-histoire-de-Git)

[About this site](https://git-scm.com/site)  
Patches, suggestions, and comments are welcome.

Git is a member of [Software Freedom Conservancy](https://git-scm.com/sfc)