



版本控制系統 (Version Control System)

Part 1 簡介

蘇維宗(Wei-Tsung Su)

[suwt\(at\)niu.edu.tw](mailto:suwt(at)niu.edu.tw)

Department of Computer Science and Information Engineering
National Ilan University





歷史版本

版本	說明	日期	負責人
v1.0	中文化初版	2019/02/26	蘇維宗
v1.1	修正圖片錯誤	2023/11/20	蘇維宗
v1.2	修正文字內容、新增 GitLab 誤刪事件新聞	2024/02/01	蘇維宗
v1.3	將revision翻譯從"版本"改為"修訂版本"	2025/04/08	蘇維宗



參考文件

1. https://en.wikipedia.org/wiki/Version_control
2. <http://homes.cs.washington.edu/~mernst/advice/version-control.html>
3. <https://git-scm.com/book/en/v2>
4. <https://nulab.com/zh-tw/learn/software-development/git-tutorial/>



內容

- 版本控制
- 版本控制系統(Version Control System, VCS)
 - VCS 的類型
 - VCS 面臨的挑戰
- VCS 服務
 - Git 雲端服務
 - VCS 工具



版本控制

- 在軟體工程中，**版本/修訂版本控制 (version/revision control)**是一種負責**管理**文件、電腦程式、大型網站、或其它控制資訊**變更**的方法並透過資訊系統來實踐。(以下統一以**版本控制**進行說明)
- 在版本控制中，通常每次變更都會以一個"**版本號碼 (version number)**"來加以辨識(例如, 0.1.3 版)。

資料來源: [Wikipedia](#)

為什麼要使用版本控制？

有時候，你可能手殘砍掉檔案(整個專案)...

趨勢

工程師「意外」輸入「rm -rf」，讓他的公司瞬間歸零無法救回！

2016/04/15 • Lisa • 資訊、工程師、工作技巧

「rm -rf」，這是一段基礎卻可以幫你消除一切的程式碼。「rm」就是叫電腦進行消除的動作；「r」表示消除指定目標中的所有的一切；「f」代表「強制 (force)」，意思就是告訴電腦執行檔案刪除時，可以忽略平常的提醒和警示。

資料來源

: <https://www.inside.com.tw/article/6147-man-accidentally-deletes-his-entire-company-with-one-line-of-bad-code>



即使是專業工程師也會發生 ...



GitLab.com is currently offline due to issues with our production database.

We're working hard to resolve the problem quickly. Follow [GitLabStatus](#) for the latest updates.

[about.gitlab.com](#)

2017.02.02 | 創業

GitLab工程師熬夜工作誤刪300GB資料，5種備份機制全失效、只剩「透明度」值得嘉許

開源程式碼管理工具GitLab發生誤刪正式資料庫事件，且5種備份機制都失效。所幸資料庫已回復完成，但仍有6小時左右的資料遺失，包含議題、合併請求等資料；儲存庫和維基等主要資料則未受影響。

#開發者 #工程師文化



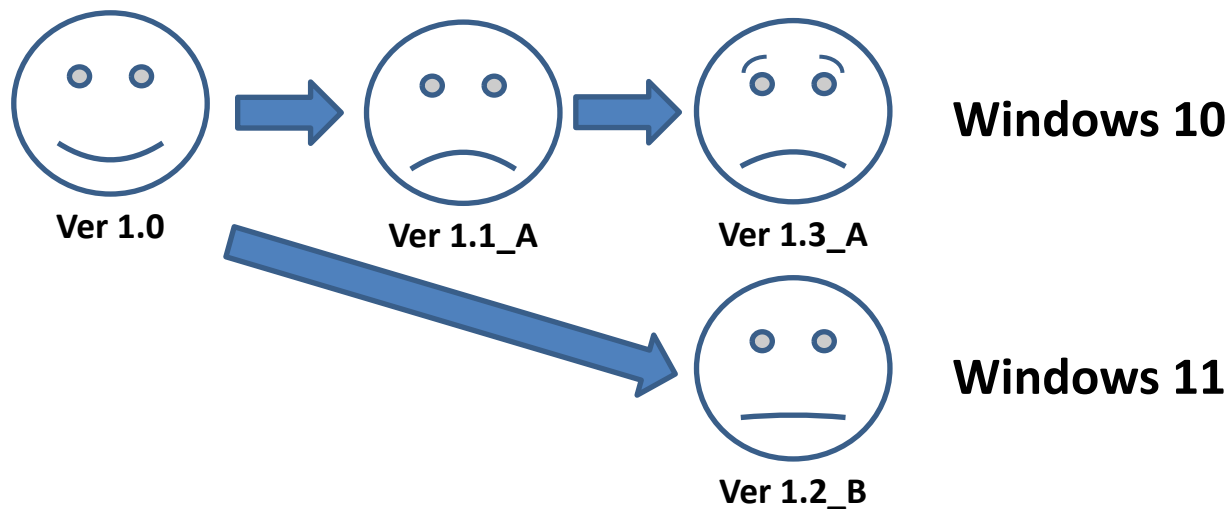
張庭瑜

資料來源：<https://www.bnext.com.tw/article/42985/gitlab-suffers-major-backup-failure-after-data-deletion-incident>



為什麼要使用版本控制？(續)

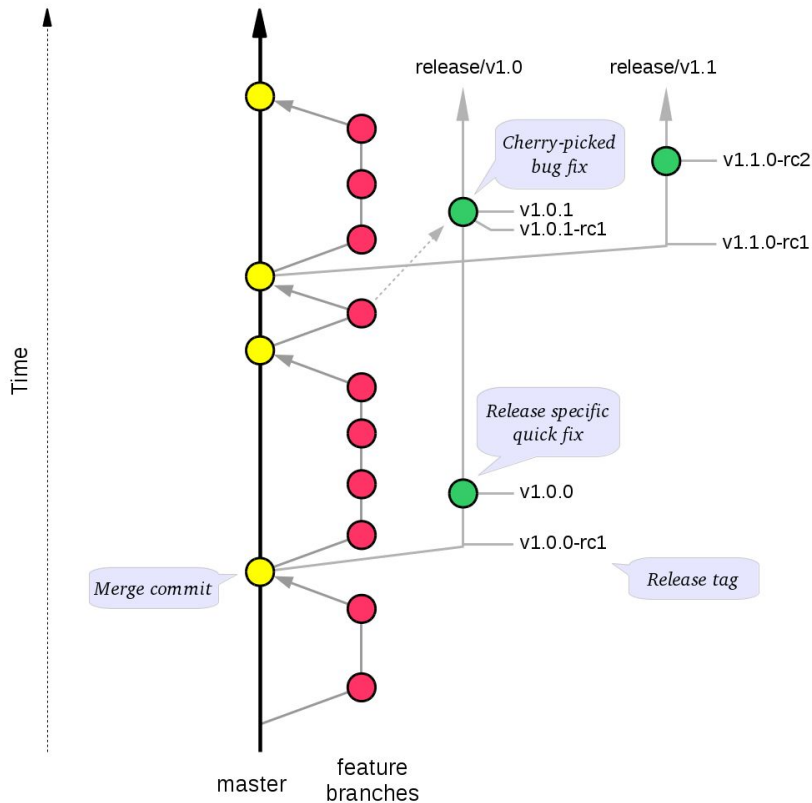
需要管理客製化功能(針對不同作業系統、不同客戶、不同產品等) ...



版本控制的特性

現代**版本控制**有以下幾個特性

- 可以記錄所有變更的歷史版本
- 可以支援多人協同開發的環境
- 可以支援專案分支
 - 不同平台、客戶或產品等
- ...



資料來源

: <https://www.bitsnbites.eu/a-stable-mainline-branching-model-for-git/>



版本控制系統 (Version Control System)

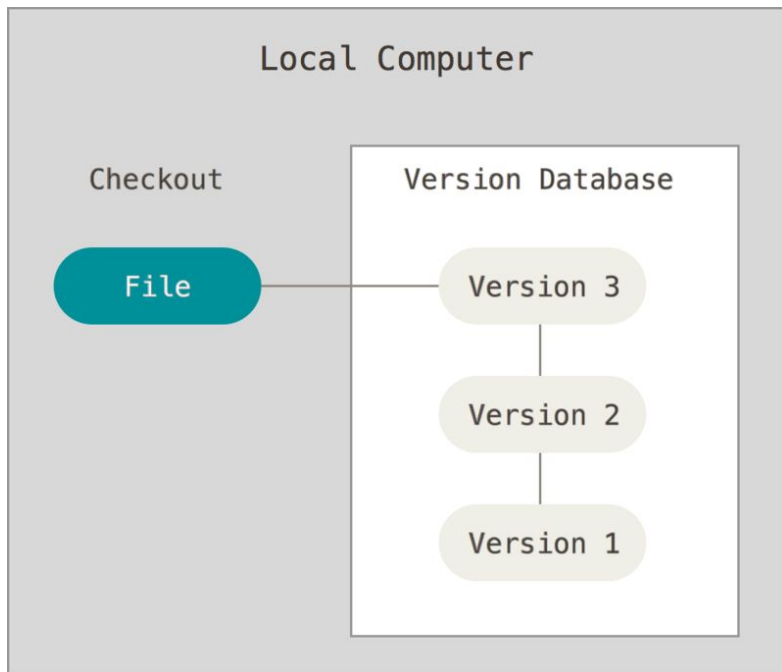
- **版本控制系統 (VCS)** 是一種資訊系統可以用來記錄檔案(或一組檔案)隨著時間改變的各個版本以利後續可以取得特定的版本。

資料來源: *Pro Git*

- VCS 的種類
 - 本地式 VCS (例如, [RCS](#))
 - 集中式 VCS (例如, [CVS](#) 與 [SVN](#))
 - 分散式 VCS (例如, [Git](#))

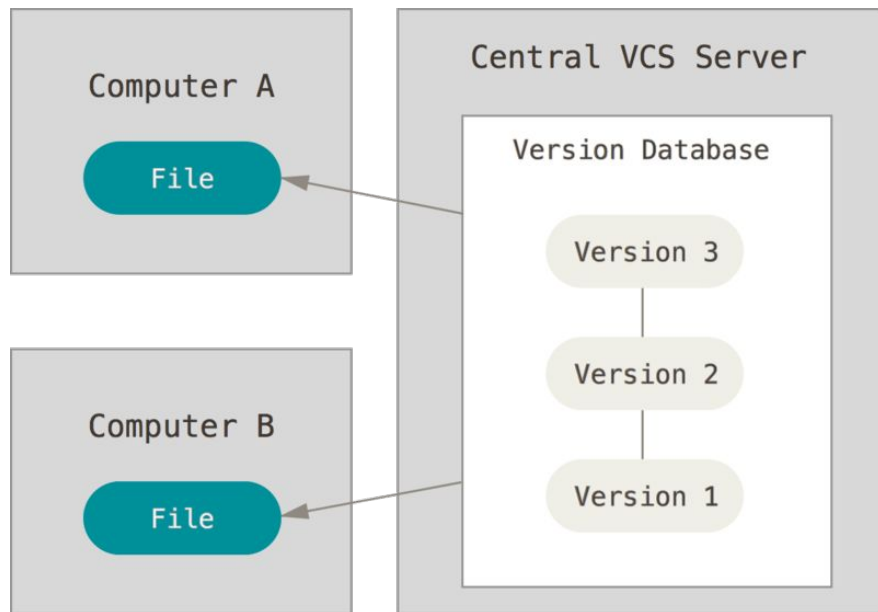
本地式 VCS

- 檔案的所有版本只會被儲存在本地端。
- 因此，本地式 VCS 適合單人開發的專案。
- RCS 是一種本地式 VCS 的版本控制系統。



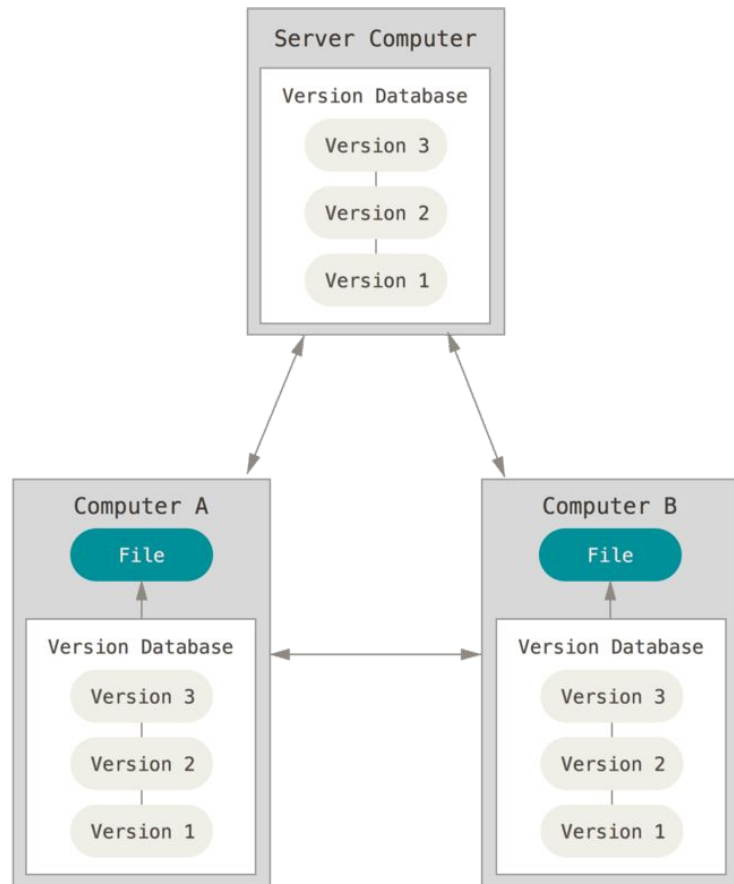
集中式 VCS

- 檔案的所有版本只會被儲存在遠端伺服器上。
- 集中式 VCS 適合多人開發的專案，但缺點是需要透過網路連上遠端伺服器才能夠變更(更新)版本。
- [SVN](#) 是一種集中式 VCS 的版本控制系統。



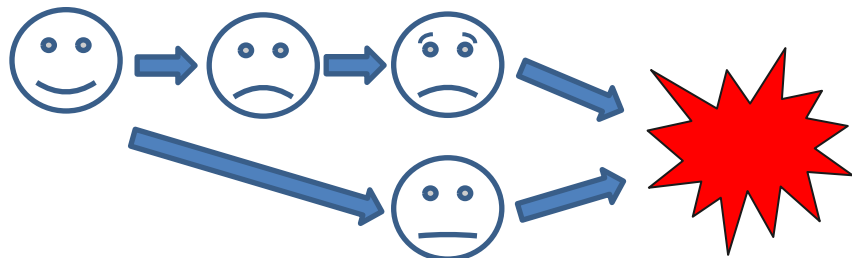
分散式 VCS

- 檔案的所有版本會同時儲存在本地端與遠端伺服器中。
- 分散式 VCS 適合多人開發的專案，變更(或更新)版本時不需要連上網路但會占用較多的本機端儲存空間。
- Git 是一種分散式 VCS 的版本控制系統。



使用 VCS 面臨的挑戰

- 使用 VCS 面臨的挑戰之一就是管理**衝突(conflict)**。
- 當有多位開發者對同一個檔案進行修改時就可能會導致衝突而讓 VCS 無法合併檔案來產生新的版本。
- 發生衝突時，必須透過手動解決。





如何避免衝突？

幾個避免衝突的建議

- 模組化專案, 並讓不同開發者負責不同的專案模組。
- 不要將專案自動產生的檔案上傳到遠端伺服器
 - 在 Git 中可以利用 `.gitignore` 檔案來達到此目的
- 先將遠端伺服器的最新版本更新到本地端合併後再上傳
- 在本地端測試後再上傳
- 經常進行整合(拖久了可能會發生更多衝突而難以合併)
- ...

知名的 Git 雲端服務

- GitHub
 - <https://github.com/>
- GitLab
 - <https://gitlab.com>
- Bitbucket
 - <https://bitbucket.org>
- SourceForge
 - <http://sourceforge.net/>





VCS 相關工具

- Git
 - git-scm: <https://www.git-scm.com>
 - Sourcetree: <https://www.sourcetreeapp.com/>
 - Gitkraken: <https://www.gitkraken.com/>
 - TortoiseGit Client: <https://tortoisegit.org>
- Subversion (SVN)
 - Apache Subversion Server: <https://subversion.apache.org>
 - VisualSVN: <https://www.visualsvn.com>
 - TortoiseSVN Client: <https://tortoisesvn.net>



Q & A



Computer History Museum, Mt. View, CA