版本控制 CWIKI

版本控制（Revision control）是维护工程蓝图的标准作法，能追踪工程蓝图从诞生一直到定案的过程。此外，版本控制也是一种软件工程技巧，借此能在软件开发的过程中，确保由不同人所编辑的同一程式档案都得到同步。

欢迎来到版本控制 CWIKI

在这个页面中，我们将会对软件工程中的版本控制知识使用进行说明，同时我们也会在 OSSEZ 社区中提供相关的技术支持。

透过文档控制（documentation control），能记录任何工程项目内各个模块的改动历程，并为每次改动编上序号。

一种简单的版本控制形式如下：赋给图的初版一个版本等级“A”。当做了第一次改变后，版本等级改为“B”，以此类推。最简单的例子是，最初的版本指定为“1”，当做了改变之后，版本编号增加为“2”，以此类推。

借此，版本控制能提供项目的设计者，将设计回复到之前任一状态的选择权，这种选择权在设计过程进入死胡同时特别重要。

理论上所有的资讯记录都可以加上版本控制，在过去的实务中，除了软件开发的流程，其它的领域中很少有使用较复杂的版本控制技巧与工具（虽然可能为其带来许多好处）。目前已有人开始用版本控制软件来管理CAD电子档案，电路板设计，来补足本来由人手工执行的传统版本控制。

软件版本控制

软件设计师常会利用版本控制来追踪、维护源码、文件以及设定档等等的改动，并且提供控制这些改动控制权的程序。

在最简单的情况下，软件设计师可以自己保留一个程式的许多不同版本，并且为它们做适当的编号。这种简单的方法已被用在很多大型的软件项目中。该方法虽然可行，但不够有效率。除了必须同时维护很多几乎一样的源码备份外；而且极度依赖软件设计师的自我修养与开发纪律，但这却常是导致错误发生的原因。

有时候，一个程式同时存有两个以上的版本也有其必要性，例如：在一个为了部署的版本中程式错误已经被修正、但没有加入新功能；在另一个开发版本则有新的功能正在开发、也有新的错误待解决，这使得同时间需要不同的版本并修改。

此外，为了找出只存在于某一特定版本中（为了修正了某些问题、或新加功能所导致）的程式错误、或找出程式错误出现的版本，软件除错者也必须借由比对不同版本的程式码以找出问题的位置。

软件版本的控制方法

最简单的版本控制就是保留软件不同版本的数份拷贝，并且适当编号。许多大型开发案都是使用这种简单技巧。虽然这种方法能用，但是很没效率。一是因为保存的数份拷贝几乎完全一样，也因为这种方法要高度依靠开发者的自我纪律，而常导致错误。因此，有人开发出了将部份或全部版本控制工作自动化的版本控制系统。

差分编码

大部份的版本控制软件采用差分编码：只保留档案相继版本之间的差异，这个方法可以更有效的储存数个版本的档案。

中央式系统与分散式系统

大部分的软件开发案，会有好几个开发人员同时工作。如果两个人员同时要改变同一个档案，而没有管理存取权限，很可能会覆写彼此的工作。

所以权限管理控制系统会在两种方法中择一解决：采用中央式系统，由中央权威管理存取权限；或是像分散式系统容许多个单位同时进行，包括同时更动同一档案。

传统上版本控制系统都是采用中央式系统：所有版本控制的工作在一个服务器进行，由中央权威管理存取权限“锁上”档案库中的档案，一次只让一个开发者工作。

2000年后，TeamWare、BitKeeper、和GNU开始用分散式系统：开发者直接在各自的本地档案库工作，并容许多个开发者同时更动同一档案，而各个档案库有另一个合并各个改变的功能。这个方式让开发者能不靠网络也能继续工作，也让开发者有充分的版本控制能力，而不需经中央权威许可。分散式系统仍然可以有档案上锁功能。

分散式系统Linux内核的发明人林纳斯·托瓦兹就是分散式版本控制系统的支持者，他开发了目前被开源社群广泛使用的分散式版本控制系统Git。

档案上锁

档案上锁功能能对高难度的合并（例如大幅更改大档案或档案群的许多部份）提供一些保护，但其他开发者仍然可以绕过版本控制系统改变档案（这本身就是很大的问题）。所以档案上锁功能带来的功效与副作用一直饱受争议。

其他功能

有些进步的版本控制工具提供更多功能，例如：

1. 管理谁能改变程式的哪个部位，
2. 提供某一个人控制权来审查哪些改变可以过关；
3. 与开发环境整合。

版本控制的常用术语

基线（Baseline）

基线是软件文档或源码（或其它产出物）的一个稳定版本，它是进一步开发的基础。

仓库（Repository）

存储档案的新版本还有历史资料的地方，通常是在服务器上。有时候也叫Depot（像是在SVK、AccuRev还有Perforce中）

工作版本（Working copy）

从档案库中取出一个本地端（客户端）的复制，针对一个特定的时间或是版本。所有在档案库中的档案更动，都是从一个工作版本中修改而来的，这也是这名称的由来。观念上，这是一个沙盒。

提交（Commit）

将本地端的修改送回档案库。（由版本控制软件处理"跟上次更动相比，哪个档案又被更动"的事）

变更（Change）

对一份文件作的特定更动。

变更记录（Change List）

检出（Check-Out）

从档案库取出档案到本地端（客户端）。

更新（Update）

将档案库的修改送到本地端（与提交相反）

合并（Merge / Integration）

合并各个改变。

版次（Revision）

一个revision或version指的是一系列版本变迁的其中之一。

导入（Import）

导出（Export）

冲突（Conflict）

当两方更动同一份文件会发生冲突。

常用的一些版本控制软件

- BitKeeper
- CVS（Concurrent Versions System）
- Micorosoft Visual SourceSafe
- Perforce
- Rational ClearCase
- RCS（GNU Revision Control System）
- Serena Dimention
- Subversion
- SVK

- Git
- Monotone (软件)
- Bazaar (software)
- Mercurial
- SourceGear Vault

注意：多数不提供中文语言界面包（一部分本身即基于命令行接口），一部分对中文没有有很好的支持，处理中文时有乱码。但一般选用支持UNICODE的软件时，CJK便不成问题。

其中有些版本控制软件已经逐渐淘汰了，例如： CVS。目前使用最多的是 SVN，GIT，Mercurial。

Recent space activity

HoneyMoose

版本控制 CWIKI updated 06/21/2018 view change

Eric Lee

版本控制 CWIKI commented 06/21/2018

YuCheng Hu

提交文件 updated 04/04/2015 view change

新建数据库 updated 04/04/2015 view change

初期设定 updated 04/04/2015 view change

Space contributors

- HoneyMoose (980 days ago)
- YuCheng Hu (2154 days ago)