# Working in the Cloud: Web-based Version Control System for Task-oriented Group and Individual Projects

# Literature review

This section was about the literatures related to this project – a web-based version control system.

## Version Control

Version control, usually be called as revision control or source control, is a method of managing files related to development of projects through their whole life cycle [1]. Most version control software includes CVS and SVN run at client-server model, so they support more than one users working with their deployment at same time [2] [3]. The general features provided by version control include store each commit/edition/version of files or directories, allow rollback, show changing history and assist merge/integration [1]. Based on the features, many kinds or parts of project can use version control to be effective managed, such as software development and document writing. Each commit/change can be marked with properties of which author did this commit and what time it committed. The commits also comes with a unique version code to identify the times of changes of it. Users can add some comments of the version when they commit it, so it is very clear to identify what the users changed in there committed version.

## Existing Version Control System

This project was focuses the development of an easy to use version control system for use on web. At least, the system should allow users to working in a group with great and efficient coordination as well as manage their private work. As the coordination and work management were important features of the system, some design can be learn from operating mechanism of existing version control system. In the following, the popular software-based client-server version control system Concurrent Versions System (CVS) [2] and Subversion (SVN) [3] will be analysed for their mechanism.

### Unique version number

Every change of directory and file will generate a new version to the directory and file itself. Copying, renaming and deleting will be considered as a change. Each historical version will be keep for in-case use [3].

### Atomic commits

For coordination considering, every submitting of change set will generate a version of change history, even only a part of the set has been changed [3]. As this project was task-based, so it could be different from the SVN versioning mechanism, which is every files will update to a latest same version code after an every submitting command. Task-based feature limited area of change set to tasks, therefore, the versioning mechanism could be designed as every time of file and directory change will increase the version number of file and directory individually, and the task version will be increased when a submit operation has been executed, no matter how many changes of file and directory have been made, even only renamed a file.

### Locking

To avoid unsynchronised editing, lock-work-submit-unlock is a very good feature provided by SVN for clash free editing [3]. In this project, the task as a minimum assignable unit, could be locked when a user start doing the task. When a lock has been set to a task, other user except the user who sets the lock cannot request to do the task, even he/she was already been assigned to the task.

The project was designed as task-based, also helps members in a group doing their common project under schedule. In contrast, none of existing version control system provides scheduling features, including the most popular software-based version control system – Subversion. To achieve the schedulable feature, the tasks could be set up with relationship with other tasks. In 1910, Henry Gantt first published the concept of predecessor in his famous Gantt chart [4]. It resolved the organisation problem of group collaboration in scheduling very well. This project uses the concept from Gantt chart to scheduling tasks. In the system, a task could have a predecessor/father task, which means a task can be start only when its predecessor task already finished.

Interaction Design

Principles

Design – Redesign

Prototype

Evaluation

Web

# Bibliography

|  |  |
| --- | --- |
| [1] | B. Collins-Sussman, F. W. Brian and C. M. Pilato, Version Control with Subversion, O'Reilly, 2004. |
| [2] | D. Price, “CVS v1.11.23 Manual,” Ximbiot LLC, 8 5 2008. [Online]. Available: http://ximbiot.com/cvs/manual/cvs-1.11.23/cvs.html. [Accessed 1 7 2011]. |
| [3] | The Apache Software Foundation, “Apache Subversion Features,” [Online]. Available: http://subversion.apache.org/features.html. [Accessed 1 7 2011]. |
| [4] | H. Gantt, Work, Wages and Profit, New York: The Engineering Magazine, 1910. |