

Simple Version Control of SAS Programs and SAS Data Sets

Magnus Mengelbier, Limelogic Ltd, United Kingdom

ABSTRACT

SAS data sets and programs that reside on the local network are most often stored using a simple file system with no capability of version control, audit trail of changes and all the benefits. We consider the possibility to capitalise on the capabilities of Subversion and other simple straightforward conventions to provide version control and an audit trail for SAS data sets, standard macro libraries and programs without changing the SAS environment.

INTRODUCTION

Most organisations will use the benefits of a local network drive, a mounted share or a dedicated SAS server file system to store and archive study data in multiple formats, analytical programs and their respective logs, outputs and deliverables.

A manual process is most often implemented to retain versions and snapshots of data, programs and deliverables with varying degrees of success most often. Although not perfect, the process is sufficient to a degree.

Organisations may invest in comprehensive enterprise environments such as SAS Drug Development and Oracle Life Science Data Hub in order to implement stricter controls and compliance.

- Standards
- Versioning
- Audit trail
- Electronic signatures

The step from a local file system to those enterprise environments can be a fair investment and a high degree of change management if you already have an analytics environment.

Off-the-shelf software, both Open Source and commercial, exist that provide simple source code control with versioning, audit trail and other features such as electronic signatures that can complement or even be combined with the current file system storage with little or no change to the current IT infrastructure.

Subversion, as one and this example, is one of the popular Open Source version control systems that would allow version control and audit trail to easily be implemented. Additional features such as electronic signatures and business controls can also be added, dependent on requirements.

SUBVERSION

Apache Subversion is a version and revision control system designed to replace systems based on the popular CVS and is widely used in both Open Source projects, communities and in commercial applications.

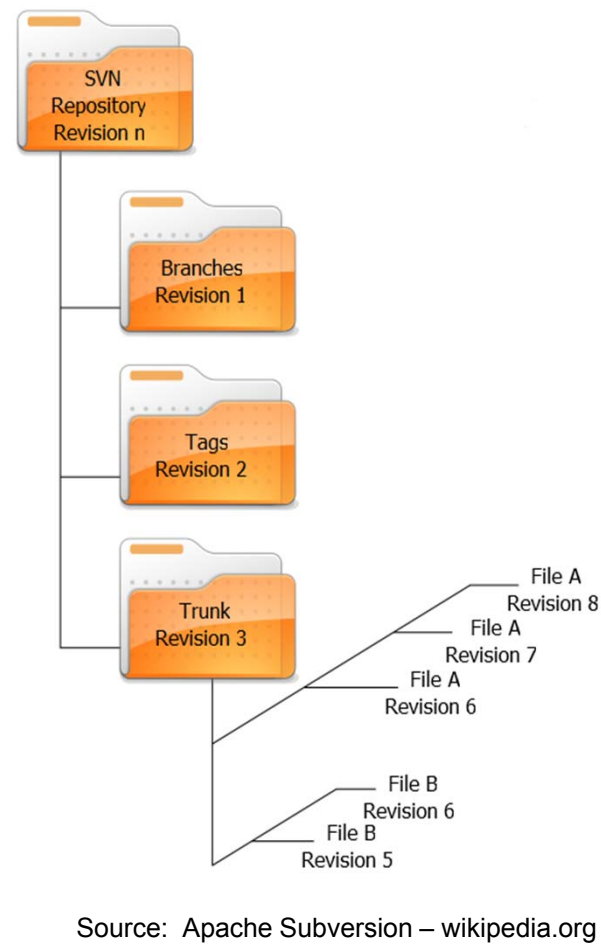
Subversion manages files and folders, and keeps track of any changes over time. Subversion is extremely simple and general system to manage any collection of files. It does not include features, such as natively understand programming languages, common in larger Software Configuration Management (SCM) systems.

The basic nature and simple features makes Subversion a very simple repository for SAS data sets, programs, logs and outputs for both small office to larger global teams across multiple sites and regions.

The Subversion “file system” is essentially two-dimensional.

- 1st dimension : The *path*, just like you would expect on a Unix, Linux or Windows local or network share.
- 2nd dimension : The *revision*. A revision is not on a single file, but the entire repository and is a very simple way to refer to versions of all files in the repository at a any point in time.

Subversion is also extremely efficient at storing multiple versions of the same file as it only saves the differences and not the entire file.



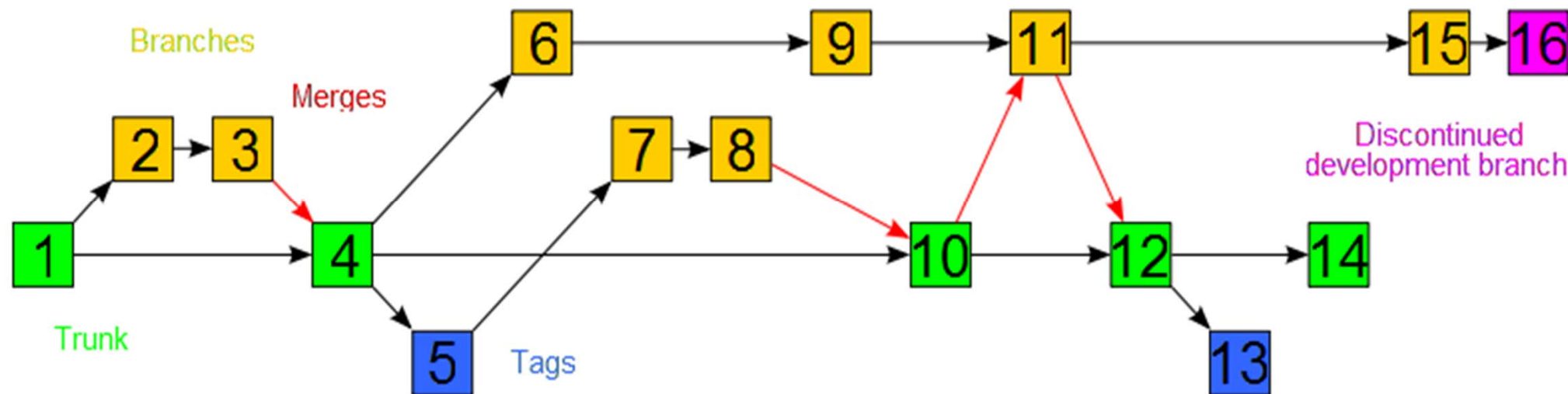
TRUNK – BRANCHES – TAGS

A Subversion repository – the location of all Subversion is stored within a *repository* – is empty by default. The repository does not require any specific directory or folder structure, and certainly not a directory or folder structure convention.

Revision 1 – the first change – of a repository is most often the empty default directory structure as this would be the first item(s) to create. Most documentation refers to three root folders in a Subversion repository; the trunk, branch and tag.

- trunk The main line of development
- branch Development lines for multiple versions of the same product
- tag Mark or highlight notable revisions in the history of the repository, such as “version 1.0”

With a Life Science perspective, the basic principle of the trunk, branches and tags is to strive to track, coordinate and *merge* all the updates to the Statistical Analysis Plan and output Shells with the actual programming and changes to deliverables. Revisions (numbers within the squares below) in Subversion performs this ballet very well.



SUBVERSION AND LIFE SCIENCES

Subversion can fit very well within Life Sciences and with a tweak here and there, the version and revision control can be a foundation for a standard and compliant analytics environment and process.

TRUNK – BRANCHES – TAGS OR DEV – QC - PROD

The approach with trunk, branches and tags can also be used within reporting clinical trials, if outputs are standardized for a specific study and used in multiple reporting events.

- trunk Pre-lock data and programs for reporting purposes
- branch Deliverables for a specific reporting event such as Investigator Brochure (IB), Investigational New Drug (IND), Clinical Study Reports (CSRs), etc
- tag Dry run, Database Lock, Draft Outputs, Final Outputs

Since the top level directories and folders of a repository are treated just like any folder and file, the common folder structure and workflow for Dev – QC – Prod are also extremely easy to implement.

ONE OR MANY REPOSITORIES

Subversion can manage a single very large repository or many smaller repositories effectively. There are benefits to both, but a convention of one repository per Study Protocol has clear benefits.

- ✓ Simplified access control
- ✓ Less revisions to track
- ✓ Revision is specific to effort on a protocol, e.g. lets use the table from revision 1026
- ✓ Greater control over process compliance
- ✓ Easy to migrate to a new process standard

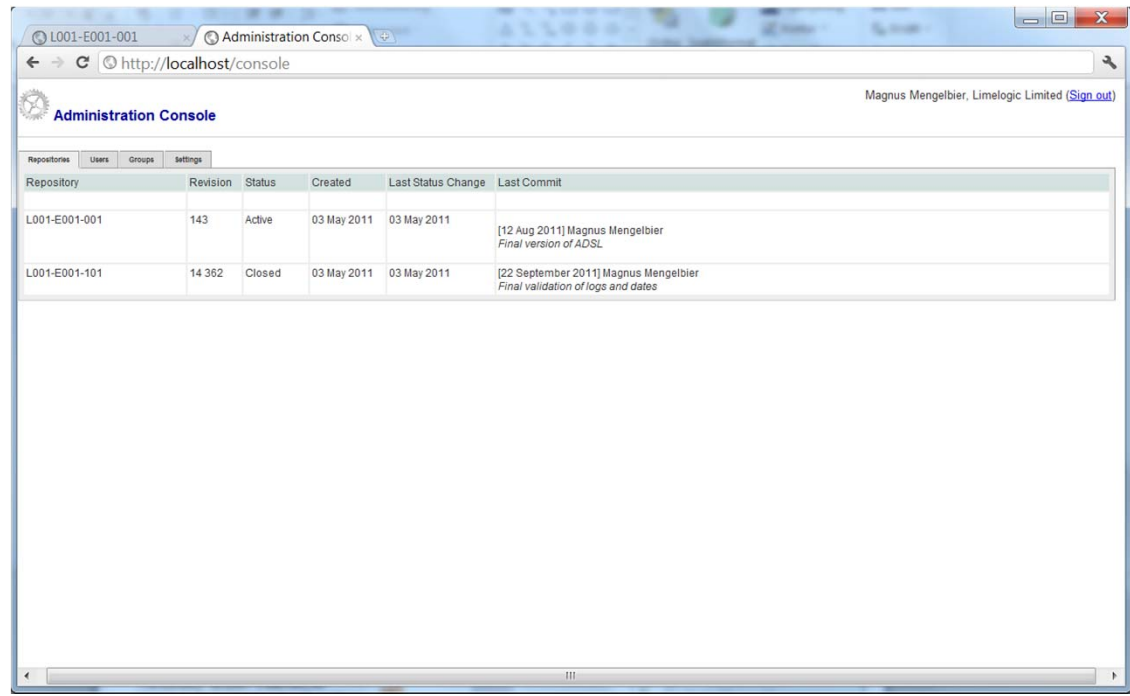


Figure 1. Simple administration console

A simple Administration Console (Figure 1) created using the Subversion programming libraries (APIs) makes creating and managing multiple smaller repositories including access control and other repository tasks a simple activity.

INTEGRATING SUBVERSION WITH STANDARD TOOLS

The programming APIs also provide a simple method to obtain and display information about programs and outputs stored in the repository. A good example would be to display dates and revision information for a SAS program in a Status and Tracking tool (Figure 2).

The Status and Tracking tool can also be extended to perform actions on the repository as well. In Figure 3, the Status and Tracking tool has been extended with the capability to lock a program file for editing, e.g. the lock beside the revision information, by a specific repository user.

Subversion lacks the traditional check-out / check-in functionality and implements a similar function with the ability to lock a file.

It is fairly easy to implement additional features and business process controls in Subversion itself using *hooks*. A hook is a small script that executes during an action on or event in the repository, which can be a general feature or specific to your business process.

A business process compliance rule can easily be added to Subversion via the *commit hook* to check if a QC program is being added to or updated in the repository by the same user that created the primary program and then take the appropriate action, such as refuse the update.

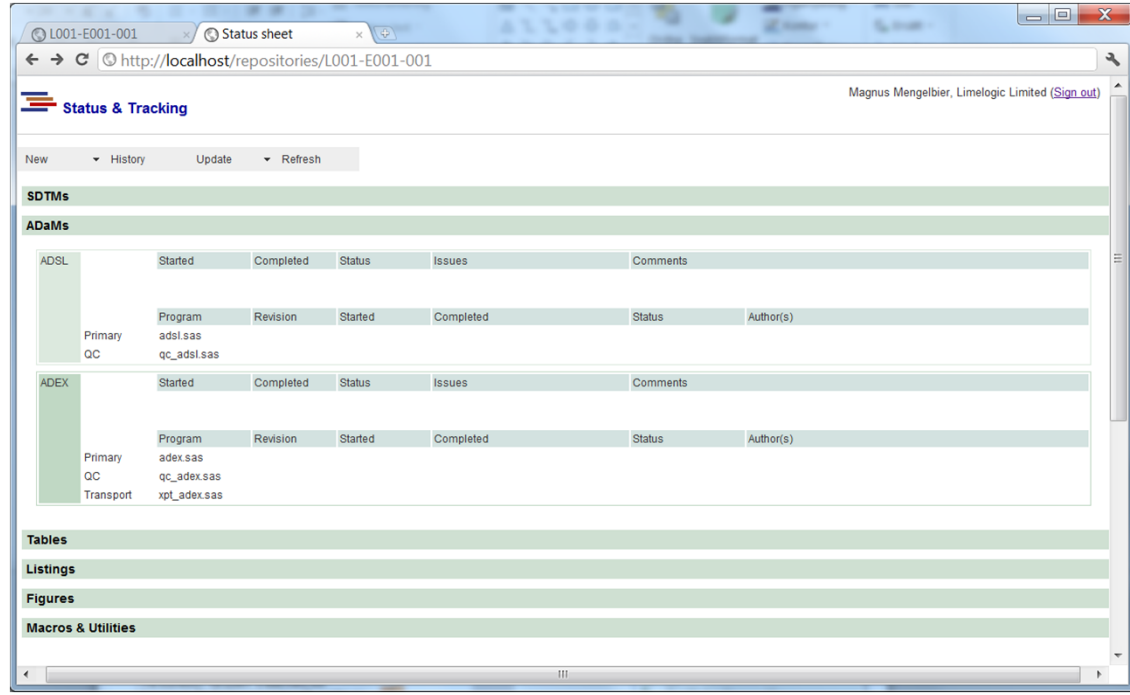


Figure 2. Status and Tracking

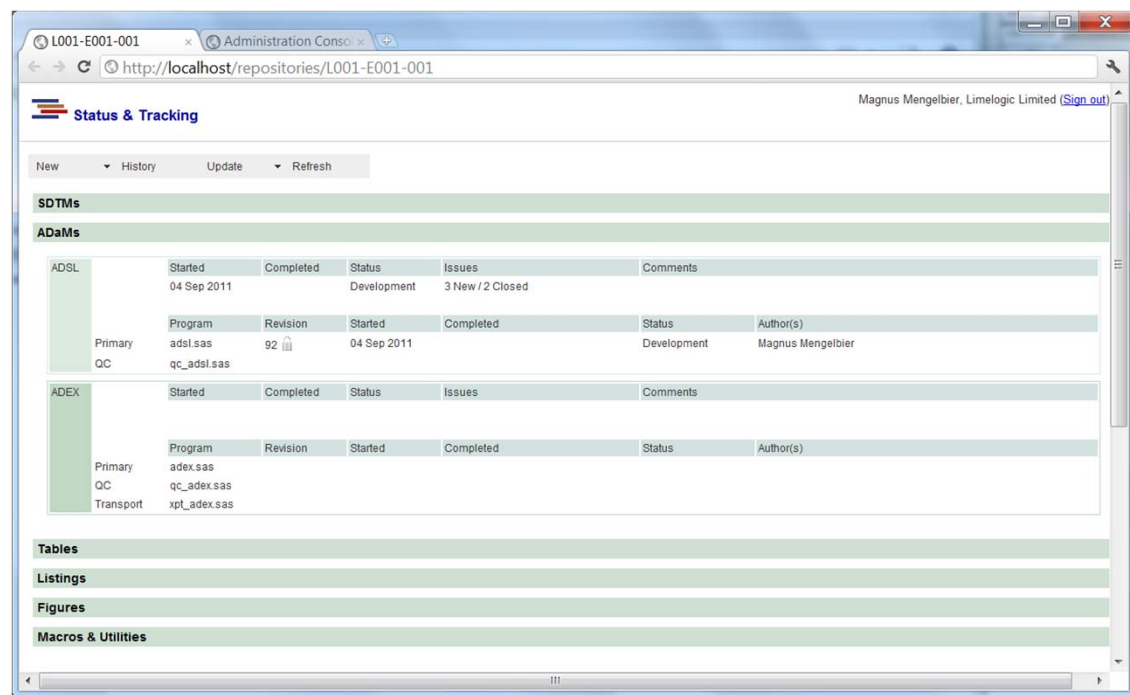


Figure 3. Lock a file for editing

CONCLUSION

Subversion is a good fit for the Life Sciences industry, simply due to its basic function and the simplicity to set up and manage one or multiple repositories. Add the possibility to adapt and extend Subversion features as well as integrate with standard process tools, and Subversion has become a very good candidate to provide version control in a Life Science analytics environment.

REFERENCES

- [1] Apache Subversion (http://en.wikipedia.org/wiki/Apache_Subversion)
- [2] Version Control with Subversion (<http://svnbook.red-bean.com/>)