

What's Next in Version Control for SAS environments?

PhUSE London 13 Oct 2014, David J Garbutt

nnit

01. Abstract

Since 2004 Version Control systems have become recognised as essential for SAS[®] programmers working in medium- to large-scale environments and have been widely implemented as well as being included in environments such as Macumba, EntimICE[®] (R) and SAS[®] Drug Development. But what next? What is new in the computing world and the world of version control? What are the essential drivers of change? Where can, and will, we be in ten years time? This paper presents a view of the future that is nearly here.

The story so far...

First came custom in-house systems like GPS II and Macumba (Weiss) and then commercial systems like SAS[®] Drug Development and entimICE[®] by Entimo. All the systems so far are using the client server model and have a centralised version Control system standing at the ready.



02. Limitations

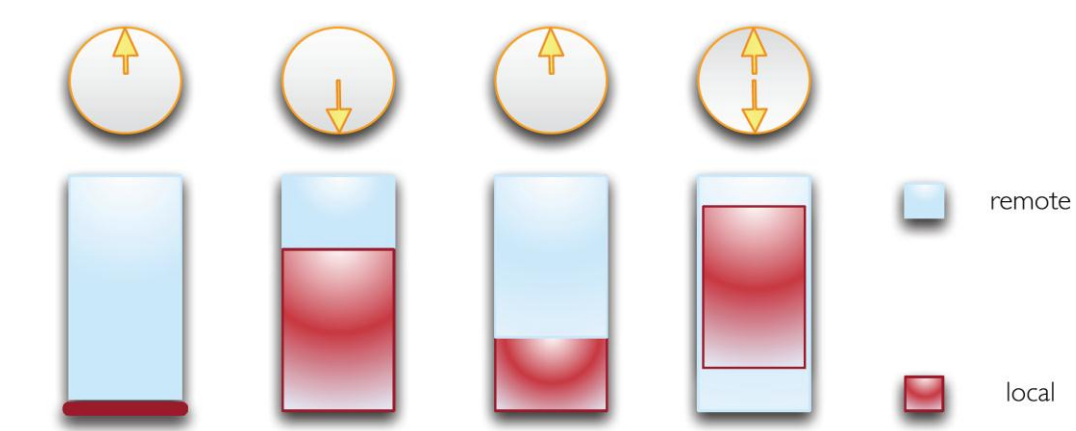
The systems mentioned are successful, how can we talk of limitations?

These systems have structural problems that arise from their client server architecture.

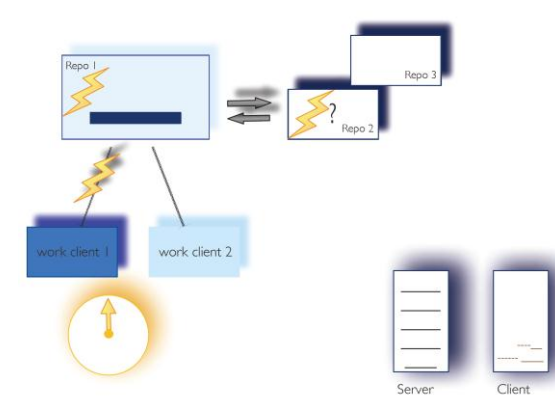
- Centralised systems require big resources and all the costs and testing that goes with it and achieving acceptable response times is a constant effort. Centralised systems are viable for groups of 100 programmers or more.
- Final run durations on large servers are *not* dramatically faster than tricked-out PCs. Consequently a small team can create a final run on their own PC's faster.
- Pharma teams are very globalised and in practice the centralised system do not support this well.
- Documentation is often not in the same system, it is often not machine readable and changes cannot easily be linked to the code or the correct output version.



03. Drivers of change

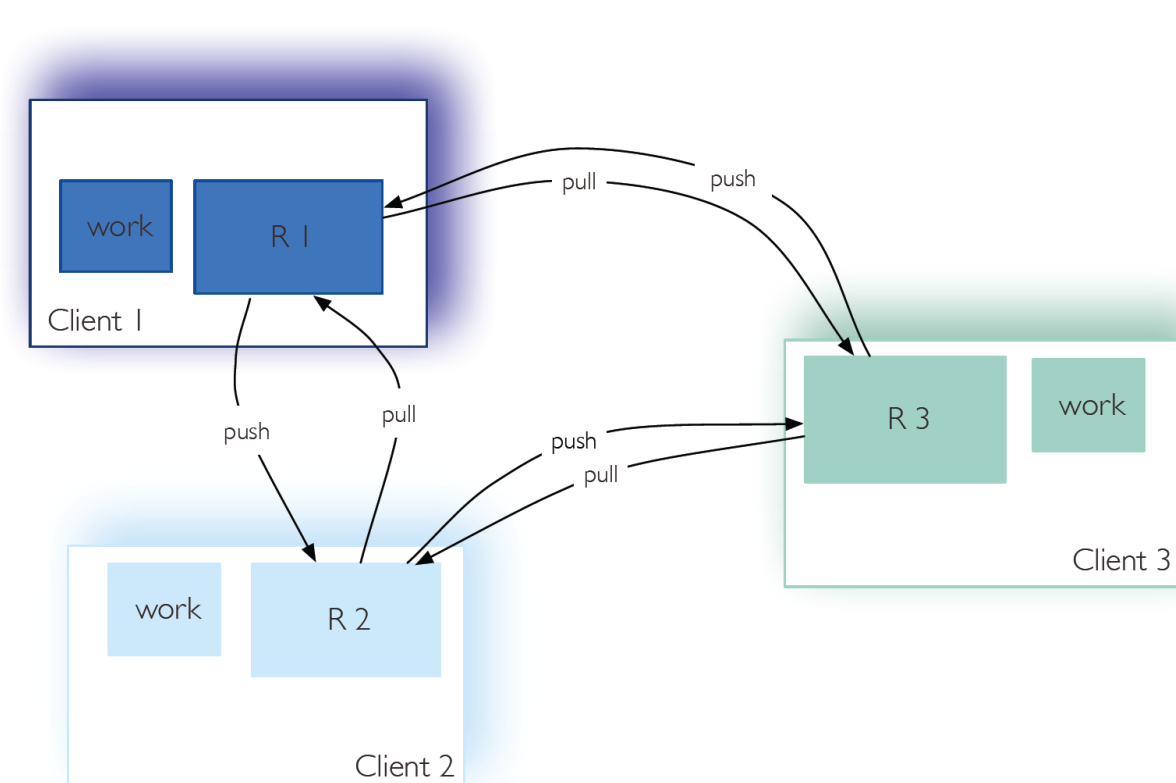


The balance of power between remote and local CPU, and the transfer speed. From the 70's to now.



The client server architecture grew popular in the 90's

04. Distributed VC



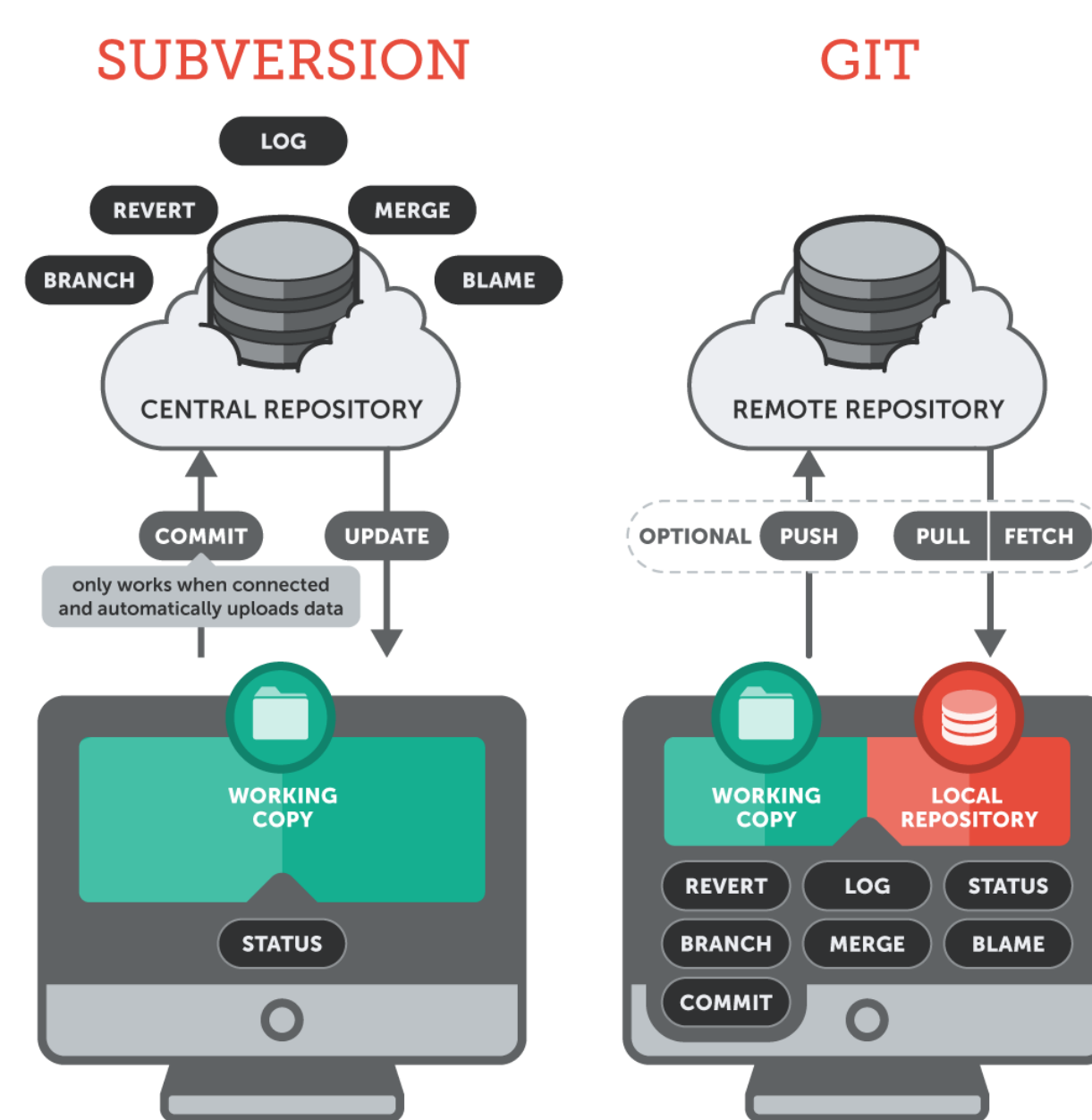
DVC systems were developed in the 2000's. They have no central repository, every machine hosts a full repository copy and most of the commands are executed locally. They also use a model of changes based around the change set rather than single file, (these would be grouped with labels in other systems). Of course one member of the repo can be a central backup server.

Two popular systems are git and Mercurial. Both have active communities and GUI tools and free web-hosting of repositories for easy sharing.



05. SVN & git commands

Where are version control commands executed?



Source: Wikipedia. (details see paper)



06. Integrated DVC

However current systems are not integrated with bug tracking (the best known systems for this are still client server systems (Atlassian Jira and FOGbugz). Integration would also be good with a wiki or other document management.

Systems are being built with all these pieces include Veracity (<http://veracity-scm.com/compare>) which was released in 2011. It also supports project time tracking, and has agile features like burn-down charts and support for sprints.

07. Distributed Databases

The last brick in the wall is the development of NoSQL databases. They are very varied but many have a distributed model as default (examples Redis and Cassandra). These could be used on a single client for metadata and other shared and synchronised information, such as test results. Titles and footnotes could also be shared like this.

08. All-in-One Client

The time could be right to build a single machine client for use as a SAS IDE with version control. All the pieces seem to be no w there.

- A DVCS there will be a shared repository per project member. A server installation will host a copy of every repository, thus keeping a backup.
- A NoSQL auto-replicating DB for metadata and other shared information.

• SAS, JMP and/or R installation .
Perhaps the last piece needed is the auto build system and an automated way to spread final runs across machines.



09 Virtualised

The advantage of the all-in-one client is the build and maintenance simplicity as well as the linear growth of costs without a high start-up price. This would be a breakthrough in cost vs functionality for small – medium companies.

And for larger companies it can be noted that a system in a single machine can be more easily virtualised and therefore scale to hundreds of users.

In principle such a system could be hosted on cloud services such as the NNIT GxP cloud. [NNIT.com services Infrastructure-outsourcing](http://www.nnit.com/services/infrastructure-outsourcing)



10. What is next?

Integrated, distributed systems are becoming the norm and open an opportunity for small and medium sized companies to have a real SAS IDE with Version Control integrated. I hope that *this* is the future for SAS VC systems.

Further Reading:

See the paper version which has more details and more references.
<http://www.lexiansen.com/phuse/2014/pp/pp04.pdf>

www.nnit.com

D Garbutt, NNIT Switzerland
davg@nnit.com
+41 79 774 28 50
Bändliweg 20, 8048 Zürich