MPDL Servicearchitektur für FoDa /
Evaluation of Solutions for Big scientific working platform

MPDL INTERN

MAX PLANCK
digital library

Evaluated solutions

Filesharing

- ownCloud.org
- academictorrents.com

Distributed Version Control Systems (DVCS)

- git plain
- git-annex
- git-lfs

Git management solutions

- GitHub
- GitLab

GitSwarm and Helix Perforce (Commercial DVCS)

Evaluated solutions

Dataset for testing

- 6156 jpeg files (faces images)
- Collection supplemented data: TOC (cvs, xml), License (pdf)
- File sizes: from 14Kb to 72Mb
- Total size: 3,5Gb

Test PCs

- Server
  - Ubuntu server 14.04 @ lenovo T430 laptop, 4CPUs@2900 MHz 16GB RAM, Disk 500 GB
- Workstations
  - Ubuntu desktop 14.04 @ lenovo T430 laptop, 4CPUs@2600 MHz 8GB RAM, Disk 500 GB
  - Windows 7 SP1 Prof @ lenovo T430 laptop, 4CPUs@2600 MHz 8GB RAM, Disk 500 GB

Filesharing: ownCloud.org

MPDL DRG server: http://10.20.5.7:81/owncloud

- PROs
  - Interface is simple and intuitive for use, does what it should do (syncing of files)
  - Desktop clients for all platforms
  - Performance (d/u): 6m15s/1h20m!!!
  - Easy to install and manage (server and clients)
  - app Store
  - CE and EE Servers
  - Mature SW (version 8.2)
  - Mobile clients (non-free for CE!)

- CONs
  - Not suitable for BLOB sharing (due to http limits)
  - Version control is rudimentary and not sustainable: old versions are removed if size quota for user is exceeded, i.e. not for LTA conform
  - No DVCS features, not suitable for development of distributed SW Projects
  - slow by upload of many files

Filesharing: academictorrents.com

- PROs
  - Bittorrent technology is perfect for sharing and distribution of BLOBs
  - Clients for all platforms
  - Good sharing performance by many seeders (much better as by ownCloud)
  - Easy commitment of torrent for shared asset to academictorrents.com

- CONs
  - The distribution of an asset via BT is only possibly with an open incoming port on seeder PC. The incoming port opening is not allowed in NAT-networks (like in MPDL) due to security vulnerability, thus seeder PC should be specially managed: moved to DMZ, dedicated FW settings, etc. (excessive management efforts)
  - Good sharing performance is achieved if many seeders/peers are online
  - BT is exclusively focused on file sharing, no other features
  - Modern hardware FWs are filtering BT protocol packets
  - Direct P2P sharing is not implemented for all BT clients, an external tracker is needed for sharing.

DVCS: git plain

- PROs
  - Very popular and powerful system for distributed SW project development, state of art of open source SW development
  - Can be used in very heterogeneous development environment: many supported platforms, 4 transport protocols, can be flexible configured in any infrastructure landscape
  - No centralized repo is needed
  - Advanced branching, powerful merging mechanism, full control over revisions, etc.: lot of powerful features and concepts
  - Good documentation, huge and comprehensive community
  - Lot of git-based tools and management services (GitHub, GitLab, Atlassian Stash, GitSwarm)

- CONs
  - Suitably for small and medium-size projects, repo cannot be bigger as 2 Gb, otherwise performance problems
  - many git concepts are complex to understand and to use, steep learning curve
  - Not suitable for BLOBs handling, inborn disability (git duplicates BLOBs in working directory and object DB, excessive disk volume usage)
  - No narrow cloning
  - Authentication over authorization
  - No security restriction concept

DVCS: git-annex, https://git-annex.branchable.com/

- PROs
  - Allows to work with files in git repositories without checking into it (solution for BLOBs)
  - Feature full:
    - Syncing for annexed files: clear concept and syntax, no central bare repo is needed, separate MD and content syncing
    - whereis command indicates the file copies distribution in all repos
    - controlled backups
    - file integrity features: fsck, unused
  - Annex assistant: synchronization tool for OSX, Linux, Android
  - Supported by GitLab EE since v7.8

- CONs
  - Special workflow for BLOBs: commands and concepts, which are not git conform
  - Not really working under Windows (fallback in direct mode, symlinks is not not supported, needs x32 version of git, bad support of symlinks)
  - Annex assistant is wired and buggy (OpenSSH issue), i.e. only command line!
  - Developed and supported by one person in Haskell (>700 Mb deps). Not really mature solution
  - Future is not clear: GiLab CE and EE supports git-lfs since last release

DVCS: git-lfs, https://git-lfs.github.com/

- PROs
  - Allows to work with text pointers to files keeping the files on remote servers
  - Clear concept, reduced set of commands, same git workflow
  - Text pointers are good supported by plain git, much better as symlinks in git-annex, i.e. can be used in windows environment
  - Easy to install
  - Implemented in go (no deps, potentially good performance)
  - Implementations for storage backends like Amazon S3 available
  - Supported by GitHub and GitLab in both CE and EE versions, i.e. now becoming standard
  - Good development community, many contributors, lot of commits

- CONs
  - Lack of documentation: no clear how to use the feature in production
  - pure command line feature
  - Not mature: v1.1.0, no usage experience, supported by GitHub and GitLab only in latest releases
  - GitHub: after pull/checkout/commit of LFS repository the pointers are overridden with real files, it's not clear how to generate new pointers for v2 (no workflow explained)
  - Git-lfs-test reference server implementation is not really fully functional (e.g. no pull/clone)

Git management solutions: GitLab http://10.20.5.7/

- PROs
  - Powerful git management system
  - Project management features like user and groups management, private repositories, issue tracking, Wiki, Graphs, Milestons, etc. Highly adjustable
  - Many features already in CE, which can be locally installed
  - Git-LFS support in CE edition from GitLab 8.2+
  - Easy to install, good bundled for Linux platform
  - Good performance for remote repos (same as by plane git)
  - Supports git-lfs and git-annex (only in EE version)
  - Mature solution (current version 8.2) Intensively developed, broadly used

- CONs
  - Many features for groups support only in EE: LDAP, Kerberos, Sharing in groups, etc.
  - Git-LFS is not jet stable, not clear workflows for LFS usage
  - Complex installation procedure for Git-LFS (only https)

Git management solutions: GitHub https://github.com/

- PROs
  - Powerful git management system
  - Project management features like user and groups management, private repositories, issue tracking, Wiki, Graphs, Milestons, etc.
  - Status of arts for open source SW development management systems: the biggest platform in the world
  - Supports git-lfs

- CONs
  - No private repos
  - CE version is only online
  - Support of groups and organizations is only in EE (can be installed locally)

GitSwarm 2015.2 and Helix Perforce (P4) http://10.20.5.8

- PROs
  - Commerical reference DVCS, mature and stable, used by many world leading enterprises
  - For large enterprises with 1000's users and TBs repositotories, highly scalable
  - Native support for BLOBs, no limits on number and size of files
  - Support of groups, highly granulated permission engine on any access level
  - Narrow cloning and monorepo support
  - Detailed documentation, good tech support
  - Natively supported on all platforms as well for server as clients, GUI and CLI
  - GitSwarm is based on GitLab CE, no learning efforts by migration from GitLab to GitSwarm
  - GitSwarm and Git Fusion makes P4 git agnostic, i.e. teams can work with repo via git and p4 clients simultaneously as preferred by team
  - Workflow Mainline, Streams concept
- CONs
  - Expensive
  - Helix Perforce is completely new SW ecosystem, which should be learned by users/admins in order to use all advantages of it

Problems

- Bittorent is doubtful due to security policy in the organizations like MPG
- ownCloud is not sustainable and performant
- Git has inborn disability for BLOBs handling, which can be partly covered with git module solutions like git-lfs and git-annex.
- Git cannot be used in context of big big enterprises with many users and many repositories

Possible solution

- Use GitSwarm and Helix Perforce