Requirements for a Central Repository for Document and Library Dependencies Management

# The needs

* A given C++ project requires external libraries in order to be built
  + External/OpenSource library
    - Public Web Site
  + External, proprietary library (binary)
    - Delivered by an external company. No public URL available.
    - Need to be stored in a secure location (access credentials)
  + External library
    - We may want to patch the sources.
    - Stored either in tarball + patch, either in a VCS
  + Internal libraries/project
    - We may want the latest « stable » version
    - We checkout directly from a given branch/tag in the VCS

The problem

* Each project writes its own set of script in order to retrieve the dependencies and build them
* Each projet use different directory schemes, compiler, build system, off/in source pattern
* Each platform may have different configuration
  + Linux Mount point (/mnt/…) are irrelevant under windows
  + Samba are not treated the same way under windows ([\\Server\Share\file.txt](file:///\\Server\Share\file.txt)) and linux (/???/???/Share/file.txt)
* Some dependencies are even stored extracted, exposing symbolic links (only accessible under NFS)
* When the administrator wants to reorganize the files, it cannot do anything without breaking old builds
* you have to mount this share under /mnt/youdonthavechoice or P:\  and configure each machine so!
* Generally, expecting each machine to have the same mount points, the same drive mapping is a nightmare. Murphy says there is too many room for error, so error WILL happens.

Path to solution : URL is the Key

* URL means Universal Resource Location
* Accessible with the same « identifier » under Windows, Linux, Mac, …
  + <http://machine/path/to/library> is valid whatever the system/environment/build system/local configuration,…
* No configuration / administration / machine synchronization mechanism needed on developer machines.
* A simple web server is required… but this is not enough

Proposed Solution: A Central repository with the following Specifications

* Simple access : [http://centralrepo.evs.tv](http://centralrepo.evs.tv/)
  + Choose the server name so it will never change.
  + Or if it has to change, the cost for upgrading all tools is negligible
* Suited for storing binary
  + Tarball / zip / installation script
  + But also any king of document
* Search System (in names, but also texts and pdf)
* Versionning system for items(keep the last n revision or a given document)
* Perpetual URL for any document (see next section)
* Hierarchical organization (we can reorganize documents/items without breaking URL)
* External access point
  + A single URL
* Multiple Edit interface
  + WebDAV
  + Web Site

Why Perpetual URL ?

* Document changes, projects are reorganized, so the location of a given document/tarball WILL change one day
  + For instance : /projects/myprojects/dependencies/tool.zip may be reorganized in
  + /departements/projects/2012/myprojects/dependencies/tool.zip
  + So URL should be persistant.
* How?
  + Use a unique ID for URL:
    - <http://centralrepo.evs.tv/go/ID=47877425747874?filename=tool.zip>
  + This URL is unique and perpetual, no matter where the document is logically reorganized afterward
  + This allow to wget it everytime we need it AND to reorganize it.
  + Just store this URL in your script as it is
* Example:
* MyProject needs Boost (OpenSource, publically downloadable on the net) and Driver SDK (provided by an external company, source or binary).
  + Boost is 50 Mb, so downloading it is not transparent. Imaging a 1 Go dependency
* Before :
  + Under Linux, I grab Boost from /mnt/DATA1/dependencies/boost.tar.gz
  + Under Windows, I download it directly from the net since this location is not accessible via Windows Share
  + For the SDK, I write a README telling the next developper how to install it
  + And Now
    - Each project is responsible for writing a single script (bash, python, whatever…) telling to download these libraries from the following URL:
      * http://centralrepo.evs.tv/go/ID=47877425747874?filename=boost.tar.gz
      * http://centralrepo.evs.tv/go/ID=87855587721122?filename=driversdk.zip
    - And that’s all!
    - Works straight under Windows
    - Works without special mount point under Linux
    - Even work under MacOS !
    - The admin / project manager can even change the logical organization of the libraries without breaking any build

Available Solutions

* [OpenText](http://www.opentext.fr/4/global.htm) ?

Own solution

* Web Server with different user roles:
  + anonymous (public browsing),
  + private (login required, document update)
  + administrator mode
* Frameworks for Web Developments
  + Django
  + Wt / Wt::DBO
* Database
  + SQLite
  + Postgre
  + …
* Dummy access plugin for major build system:
  + CMake
  + Python
* Advanced features
  + Remote repository caching (like Nexus)
  + WebDAV access to drop/update files directly from Windows Explorer