Migration of the Gaudi and LHCb Software Repositories from CVS to Subversion

M. Clemencic, H. Degaudenzi

CERN - LHCb

CHEP 2010 - Taipei Taiwan
Outline

Introduction
  Why migrate?

Preparing the Migration
  The Structure

The Migration
Outline

Introduction
  Why migrate?

Preparing the Migration
  The Structure

The Migration
Why migrate to Subversion?

What’s wrong with CVS?

• Subversion was developed to replace CVS overcoming its limitations (see Subversion’s History)
• stable and mature product (started in 2000)
• CERN IT started a Subversion service
  • planned stop of the CVS service
Why migrate to Subversion?

What’s wrong with CVS?

• Subversion was developed to replace CVS overcoming its limitations (see Subversion’s History)
• stable and mature product (started in 2000)
• CERN IT started a Subversion service
  • planned stop of the CVS service
Subversion Philosophy
From the Subversion F.A.Q.

Interface to the repository

[...] Subversion’s repository interface [...] is a "versioned filesystem" in the sense that it stores a directory tree whose state is remembered from revision to revision. [...] this particular filesystem doesn’t lose data when written to; old tree states can be retrieved as easily the most recent state.*

Tags and branches

[...] branches and tags are conventions built on top of copies, instead of being basic concepts built into Subversion itself [...]*/
Subversion vs. CVS

CVS
- history of single files
- directories in the repository to emulate structure
- logical names (tags) to identify special revisions

Subversion
- history of the repository as a whole
- efficient storage of copies
- no built-in concept of tags
- atomic commits
- meta-data (properties)
Outline

Introduction

Why migrate?

Preparing the Migration

The Structure

The Migration
Importance of the Structure

Subversion doesn’t have built-in tags and branches

• *emulated* with copies
• conventional names (not enforced)
  • trunk: main development
  • tags: copies meant to be stable
  • branches: copies for parallel work
• conventional structure fits simple projects
  • tags can be created only for directories
    `svn cp svn://myrep/trunk svn://myrep/tags/v1r0`
• large/complex projects need several level of tags
  • projects, modules, submodules...
LHCb Needs

LHCb Software Development Scheme:

- Packages
  - smallest tagged units
- Projects
  - flexible collection of packages
- Repository
  - One repository to host several projects

Project tags are used to refer to version of the packages

- custom tool to check out packages and projects

Special use-case: check out projects with plain cvs/svn

- need for project tags encompassing the packages
LHCb Needs

LHCb Software Development Scheme:

- Packages
  - smallest tagged units
- Projects
  - flexible collection of packages
- Repository
  - One repository to host several projects

Project tags are used to refer to version of the packages
- custom tool to check out packages and projects

Special use-case: check out projects with plain cvs/svn
- need for project tags encompassing the packages
LHCb Needs

LHCb Software Development Scheme:

- Packages
  - smallest tagged units
- Projects
  - flexible collection of packages
- Repository
  - One repository to host several projects

Project tags are used to refer to version of the packages
- custom tool to check out packages and projects

Special use-case: check out projects with plain cvs/svn
- need for project tags encompassing the packages
LHCb Needs

LHCb Software Development Scheme:

- Packages
  - smallest tagged units
- Projects
  - flexible collection of packages
- Repository
  - One repository to host several projects

Project tags are used to refer to version of the packages

- custom tool to check out packages and projects

Special use-case: check out projects with plain cvs/svn

- need for project tags encompassing the packages
LHCb Needs

LHCb Software Development Scheme:

- Packages
  - smallest tagged units
- Projects
  - flexible collection of packages
- Repository
  - One repository to host several projects

Project tags are used to refer to version of the packages

- custom tool to check out packages and projects

Special use-case: check out projects with plain cvs/svn

- need for project tags encompassing the packages
The Structure

- Projects
- Main development line
  - packages
- Tags
  - package tags
  - project tags
- Branches
- Meta-data for efficient look-up
The Structure

- Projects
  - Main development line
    - packages
  - Tags
    - package tags
    - project tags
  - Branches
  - Meta-data for efficient look-up
The Structure

- Projects
- Main development line
  - packages
- Tags
  - package tags
  - project tags
- Branches
- Meta-data for efficient look-up
The Structure

- Projects
- Main development line
  - packages
- Tags
  - package tags
  - project tags
- Branches
- Meta-data for efficient look-up
The Structure

- Projects
- Main development line
  - packages
- Tags
  - package tags
  - project tags
- Branches
- Meta-data for efficient look-up
The Structure

- Projects
- Main development line
  - packages
- Tags
  - package tags
  - project tags
- Branches
- Meta-data for efficient look-up
The Structure

- Projects
- Main development line  
  - packages
- Tags  
  - package tags
  - project tags
- Branches
- Meta-data for efficient look-up
The Structure

- Projects
- Main development line
  - packages
- Tags
  - package tags
  - project tags
- Branches
  - Meta-data for efficient look-up
The Structure

- Projects
- Main development line
  - packages
- Tags
  - package tags
  - project tags
- Branches
- Meta-data for efficient look-up
Helping the Users

The most affected by a migration are the users

- Most common use cases
  - check out packages and projects
    - already existing tool for check out
  - commit
  - tag
- For a smooth migration
  - update the check-out tool to work with CVS and Subversion
  - prepare new tools to flatten out the differences
- User support
  - Tutorials
  - Up-to-date instructions
Helping the Users

The most affected by a migration are the users

- **Most common use cases**
  - check out packages and projects
    - already existing tool for check out
  - commit
  - tag

- **For a smooth migration**
  - update the check-out tool to work with CVS and Subversion
  - prepare new tools to flatten out the differences

- **User support**
  - Tutorials
  - Up-to-date instructions
Helping the Users

The most affected by a migration are the users

- **Most common use cases**
  - check out packages and projects
    - already existing tool for check out
  - commit
  - tag

- **For a smooth migration**
  - update the check-out tool to work with CVS and Subversion
  - prepare new tools to flatten out the differences

- **User support**
  - Tutorials
  - Up-to-date instructions
Helping the Users

The most affected by a migration are the users

• **Most common use cases**
  • check out packages and projects
    • already existing tool for check out
  • commit
  • tag

• **For a smooth migration**
  • update the check-out tool to work with CVS and Subversion
  • prepare new tools to flatten out the differences

• **User support**
  • Tutorials
  • Up-to-date instructions
Planning

• *Incremental* migration
  • coexistence of CVS and Subversion
• First packages/projects on a volunteer basis
• Dead-line for the complete migration
  • flexible, migration agreed with project managers
Procedure

The librarian takes care of migrating the packages/projects.

- Template configuration for `cvs2svn` (Python)
  - insert the packages and projects to migrate
  - the actual configuration computed automatically
- Well defined steps
  - announce migration
  - lock write access to CVS for the packages
  - generate dump with `cvs2svn`
  - filter the dump to remove unneeded parts
  - prepare the structure in svn repository
  - load the dump into svn repository
  - disable check-out from CVS
  - enable check-out from svn (repository properties)
  - announce the completed migration
Procedure

The librarian takes care of migrating the packages/projects.

- Template configuration for *cvs2svn* (Python)
  - insert the packages and projects to migrate
  - the actual configuration computed automatically

- Well defined steps
  - announce migration
  - lock write access to CVS for the packages
  - generate dump with *cvs2svn*
  - filter the dump to remove unneeded parts
  - prepare the structure in svn repository
  - load the dump into svn repository
  - disable check-out from CVS
  - enable check-out from svn (repository properties)
  - announce the completed migration
Procedure

The librarian takes care of migrating the packages/projects.

- Template configuration for `cvs2svn` (Python)
  - insert the packages and projects to migrate
  - the actual configuration computed automatically
- Well defined steps
  - announce migration
  - lock write access to CVS for the packages
  - generate dump with `cvs2svn`
  - filter the dump to remove unneeded parts
  - prepare the structure in svn repository
  - load the dump into svn repository
  - disable check-out from CVS
  - enable check-out from svn (repository properties)
  - announce the completed migration
Summary

• Subversion is very flexible, but...
  • ...the migration requires careful definition of the structure.
  • It’s useful to wrap the conventions in custom tools
    • mandatory for incremental migration.
• The migration of the LHCb software almost completed
  • 27 projects out of 28 migrated (645 packages)
  • completion scheduled for December
Summary

• Subversion is very flexible, but...
• ...the migration requires careful definition of the structure.
• It’s useful to wrap the conventions in custom tools
  • mandatory for incremental migration.
• The migration of the LHCb software almost completed
  • 27 projects out of 28 migrated (645 packages)
  • completion scheduled for December
Summary

• Subversion is very flexible, but . . .
• . . . the migration requires careful definition of the structure.
• It’s useful to wrap the conventions in custom tools
  • mandatory for incremental migration.
• The migration of the LHCb software almost completed
  • 27 projects out of 28 migrated (645 packages)
  • completion scheduled for December
Summary

- Subversion is very flexible, but...
- ...the migration requires careful definition of the structure.
- It’s useful to wrap the conventions in custom tools
  - mandatory for incremental migration.
- The migration of the LHCb software almost completed
  - 27 projects out of 28 migrated (645 packages)
  - completion scheduled for December