Change management with Subversion

Damien Guard (BSc, MBCS)
http://damieng.com
damien@envytech.co.uk

Guernsey Software Developer Forum
http://developers.org.gg

Why manage change?

Why manage change?

- Know and control what goes into a release
- Develop versions in parallel
- Work with any previous code-base
- Apply change where it is needed

Why manage change?

- Function as a team without losing work
- Merge automatically where possible
- Resolve conflicts interactively
- Autonomous off-line or distance working

Why manage change?

- Audit log of what was changed by whom
- Legal due diligence & professional conduct
- Standards & certification compliance
- Individual responsibility for contributions

Why manage change?

- Automated build & deployment
- Continuous integration testing

Why Subversion?

- Open source, cross platform
- Stable, widely used
- Unobtrusive copy-modify-merge model
- Integration via additional tools

Technical benefits

- Transactional commit mechanism
- Metadata support per file & directory
- Off-line support
- Efficient use of network & disk

Terminology

Repository

- Central store located on a server
  e.g. svn://myserver/ https://s.myserver.com/

- Updated through commits

Single project	Multiple project

Commit

- Set of related changes
- Receives unique sequential number *n*
- Revision *n* refers to the repository after commit *n* is applied
- Commit early and often

Trunk

- Primary path used for development
- Active and potentially unstable
- Should still compile ideally
  ...and pass automated tests

Branch

- Copy of trunk, a branch or tag for isolation
- When preparing for a release
  ...and trunk needs to carry on
- When change is too disruptive
  ...and will hinder other developers
- Merge changes back to trunk later

Tag

- Copy of trunk or a branch for information
- Each tag represents a specific release
- Easier than remembering revision & path
- No further commits
  ...so control /tags/ with security

Working copy

- Partial or full checkout of the repository
- Local copy for a single developer
- Native file formats - edit as usual
- Subversion client data in .svn directories

Server options

- Hosted open - *Google Code, SourceForge*
- Hosted private - *CVSDude $7-$30/month*
- Svnserve dedicated - *Simple set-up*
- Apache mod - *HTTP, SSL, SSPI*

Clients

- Command-line, *cross platform & scriptable*
- TortoiseSVN *for Windows Explorer*
- AnkhSVN *for Visual Studio*
- Subclipse *for Eclipse*

Getting started

- Setup repository folders
- Checkout
- Import files
- Commit

Typical process

- Make changes
- Update working copy
- Resolve any conflicts
- Test
- Commit

Other operations

- Revert
- Patches
- Branch & switch
- Blame
- Lock & unlock

Additional tools

- Trac
  ticket & milestone tracking, wiki
- CruiseControl
  continuous integration
- ViewVC
  web interface

Conclusion

- Subversion can work for you
- Simple & inexpensive to get started
- Address release, audit and collaboration headaches

More information

- Official web site
  http://subversion.tigris.org
- Version Control with Subversion
  http://svnbook.red-bean.com
- Bruce Perens: Subversion Version Control
  http://phptr.com/perens/

Questions?

Developer buy-in

- Commitment
  Developers must be on-board

- Unobtrusive
  Stay out of the way until needed

- Enabling
  Make bold changes & remove dead code

Disadvantages

- Apache installation process
- Large file handling via Apache > 15MB
- Management of security
- Tracking of merge

Alternatives

Product	Points	Per user
Perforce	✔ large file support ✔ merge & branch tracking ✘ IDE integration	$800
ClearCase	✔ comprehensive	$1,770
Team Services	✔ Visual Studio integration ✔ bug-tracking & testing ✘ Interoperability	$3,300
SourceSafe	✘ Unreliable ✘ Obtrusive lock/unlock cycle	$0