



Subversion: Better Living Through Branches

25th September 2008
Square Pig, London

A WEE DRAM OF
~~SCOTCH ON THE ROCKS~~

What is he on about?

- Subversion is a great version control system
- Most talks cover the basics...
- ...but don't talk about branches in depth
- This will show you how to use branches to
 - make deployment easier
 - make development of multiple releases easier
 - provide conveniences to help organize code

Who is this man?

- VP Engineering, Broadchoice
- Previously:
 - Manager, Adobe Hosted Services
 - Senior Architect, Macromedia
 - Senior Consultant, various web co.
- Architect / Developer:
 - ColdFusion since 2001
 - Java since 1997 (and Groovy since 2008)

Agenda

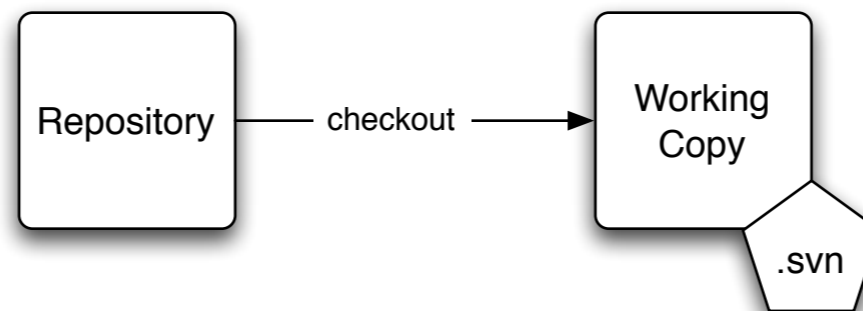
- Subversion - Basic Principles
- Repository Structure
- Branching
 - Deployment
 - Development
 - Convenience
- Live Demo!
- Recap & Resources

Basic Principles (I)

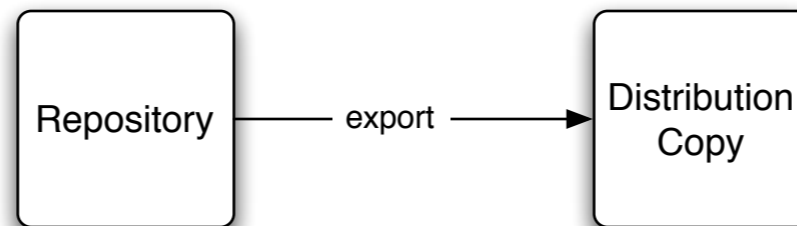
- Subversion provides a remote (or local) backup of your files (repository)
- Stores changes
- Able to reconstruct any prior version
- Able to show who changed what line and when - accountability... or blame :)

Basic Principles (II)

- Checkout - get a working copy from the repository (with .svn folders)

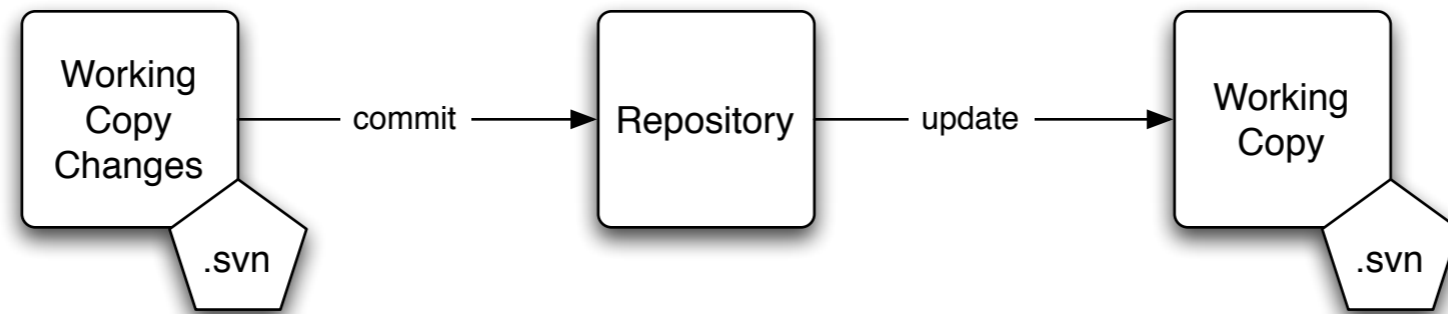


- Export - get a copy for distribution (no .svn)



Basic Principles (III)

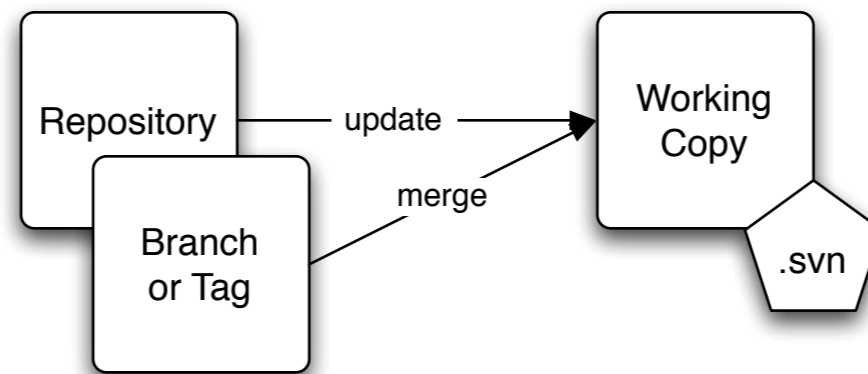
- Update - get latest changes (from repository to working copy)
- Commit - submit local changes back into the repository



- Automatically** merges changes on update

Basic Principles (IV)

- Merge - copy changes to/from branch



- This is the only complicated part of using a version control system!

Alternatives to SVN

- CVS, RCS, SCCS (older)
 - CVS is probably second most popular to SVN
- git (new, distributed system - used by Linux)
- Visual Source Safe (ick!)
- Perforce ("enterprise" strength - \$\$\$)

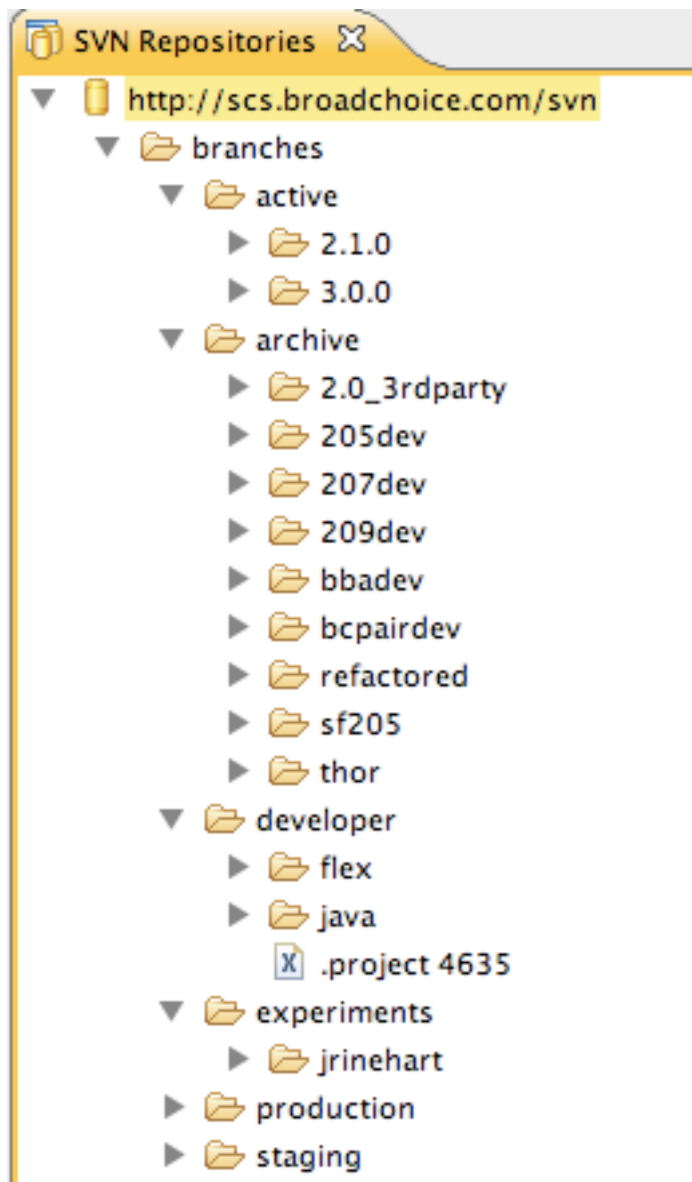
Command Line

- Many GUI tools but the command line is worth knowing, especially since most GUIs cannot handle very large operations...
- `svn {command}`
 - checkout, export, update, merge, commit
- Most can take a version, a URL and a path

Repository Structure

- **trunk/**
 - day-to-day development
- **tags/**
 - any previous version / release so that you can easily reconstruct it
- **branches/**
 - a myriad uses!

What's At Broadchoice?



On the left, branches

Active

On the right, tags

Archive

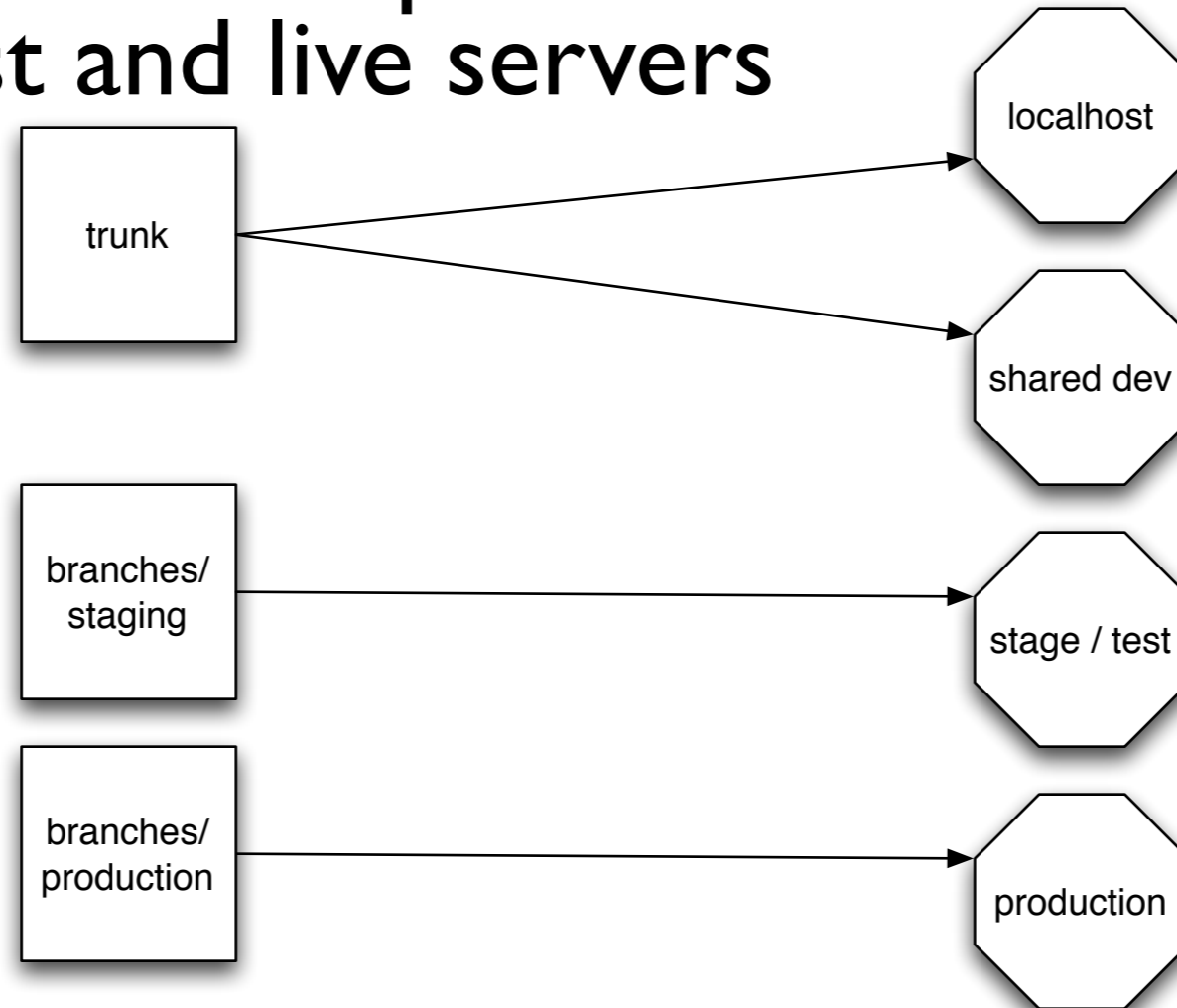
Developer

Experiments

Production & Staging

Branching 4 Deployment

- Trunk is development but create branches for test and live servers

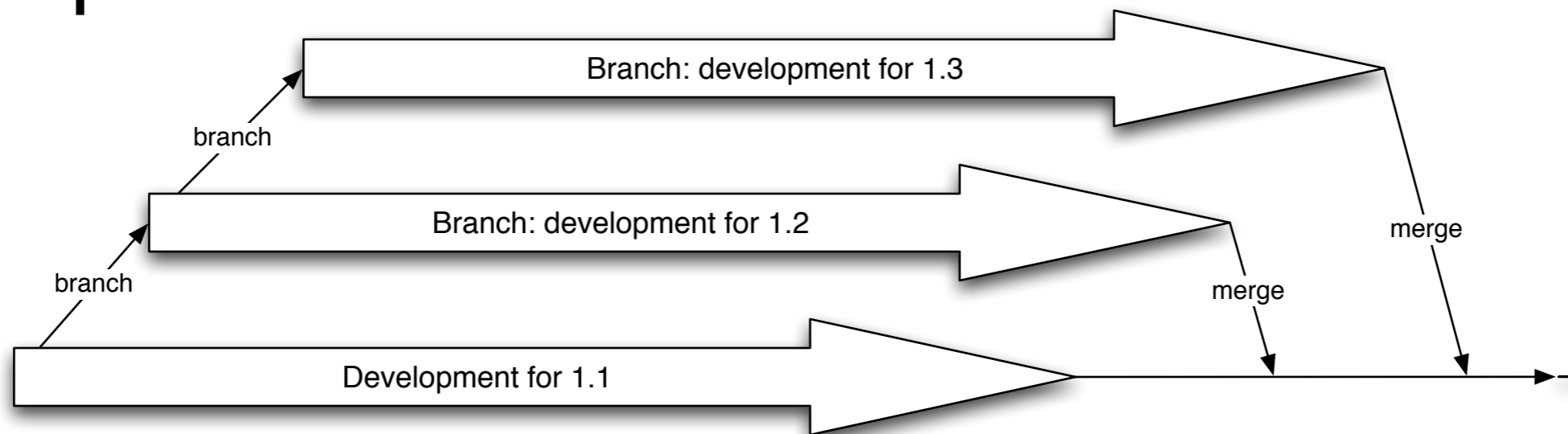


For Deployment

- Create a branch for each tier
- Perform initial checkout to each tier
- Merge from trunk to staging branch for QA
- Merge from staging to production branch
- Deployment becomes just: `svn update`
- **Always create a tag for every release!**

Branching 4 Development

- Trunk is next release but branches are for next+n releases that you work on in parallel



For Development

- Create a branch for a future release when development needs to start before the next release is deployed
- Merge fixes from trunk to branch to keep branch current (as needed)
- Merge branch to trunk once next release is deployed (and retire the branch)
- Can have many parallel branches if needed

Branch Management

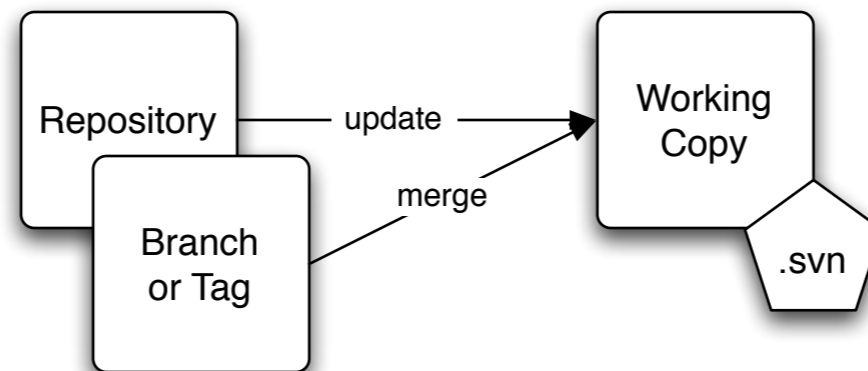
- branches/active/ and branches/archive/
- Multiple branches for a single release
 - To support multiple developers making radical changes to different parts of the system
- Create a branch for a feature that may or may not make a given release
- It can be merged to the appropriate place later

Merging in SVN (I)

- The only (slightly) complicated part of using branches
- You need an up-to-date working copy of the target branch (or trunk)
- You need to know the exact version range for the changes you want to merge
- You merge from the repository to the working copy

Merging in SVN (II)

- `svn merge -r 4589:4613`
[http://repo.com/branches/active/209dev](http://repo.com/branches/active/209dev/path/to/my/copy)
[/path/to/my/copy](#)



Conflicts

- Conflict when same line changed "twice"
- Conflicts can happen when you update
 - Other people's changes collide with yours
- Conflicts are more likely when merging
 - File has been changed on both trunk & branch
- SVN warns of conflicts and keeps "his & hers" copies for you to resolve the conflicts

Branching 4 Convenience

- A branch does not actually have to be based on the trunk (or another branch)
- Branches (and tags) are just a convention for organizing SVN repositories
- You can create a "branch" folder for anything - if it makes your development or deployment processes more convenient
- To some people, this might be **HERESY!**

For Convenience

- We have `branches/developer/` containing third party Flex and Java libraries
 - Shared across projects
 - Only needed on developer laptops
 - Never "deployed" anywhere
- We have a branch for Flex proof of concept work and experimentation
 - Makes it easy to share / collaborate with the team without "polluting" development streams

Live Demo!

- Some simple examples of branching, merging and conflict resolution!

Recap

- Basic version control by itself is very useful, even if you are a solo developer
- The real power of SVN comes from using branches and the ability to merge changes easily across multiple development streams
- SVN can help with deployment too, isolating your QA and production systems, making it easier to know exactly what code is where

Resources

- The "Red Bean Book" from O'Reilly:
 - <http://svnbook.org/> a.k.a.
 - <http://subversion.red-bean.com/>



Q&A

scorfield@broadchoice.com

<http://broadchoice.com/>

sean@corfield.org

<http://corfield.org/>

25th September 2008
Square Pig, London

A WEE DRAM OF
~~SCOTCH ON THE ROCKS~~