Coding in a Distributed Team

Testing, Reviewing, Sharing and Merging Code Without Going Crazy

Andrew Bennetts
andrew@puzzling.org

8th December, 2006

Introduction

- Who am I?
- Some common problems

The Trunk

- Tests must always pass on the trunk
- Enforcing that tests pass
- Coping with inter-dependent projects

Mandatory Code Reviews

- Reviews improve code quality
- Reviews spread knowledge
- How to manage reviews

Bazaar: Distributed Version Control

- What is distributed version control?
- Positive Effects
- Why Bazaar?

The End

- Stuff I didn't talk about
- Summary
- Questions?

Who am I?

I work on . . .

- Launchpad — https://launchpad.net/
- Bazaar — http://bazaar-vcs.org/
- Twisted — http://twistedmatrix.com/

CANONICAL

Some common problems

“I need an answer from Steve. . .

Some common problems

“I need an answer from Steve. . . but Steve is in Lithuania.”

Some common problems

“That #?%$ing Australian checked in broken code. . .

Some common problems

“That #?%$ing Australian checked in broken code. . . *again*!”

The Trunk

Tests must always pass on the trunk

Tests must always pass on the trunk.

Tests must always pass on the trunk

Tests must always pass on the trunk.
Tests must always pass on the trunk.

Tests must always pass on the trunk

Tests must always pass on the trunk.
Tests must always pass on the trunk.
Tests must always pass on the trunk.

Tests must always pass on the trunk

. . . you do have an automated test suite, right?

Tests must always pass on the trunk

If tests don’t pass, developers are impeded. They can’t safely commit changes to the trunk. They can’t safely start new work.

Tests must always pass on the trunk

If tests don't pass, developers are impeded. They can't safely commit changes to the trunk. They can't safely start new work.

It's just bad.

CANONICAL

Enforcing that tests pass

Make sure tests pass on the trunk

Enforcing that tests pass

Make sure tests pass on the trunk — automate it.

Enforcing that tests pass with PQM

PQM (Patch Queue Manager) is a tool to enforce that tests pass.

Only PQM can commit to trunk. Developers do not have write access to trunk. PQM is configured to run the test suite before accepting a change. It accepts merge requests (as GPG-signed emails) and checks that they are good before committing them.

Launchpad and Bazaar use PQM.
`http://bazaar-vcs.org/PatchQueueManager`

Enforcing that tests pass with Buildbot

Buildbot is a less intrusive (but less strict) way to enforce passing tests.

Buildbot watches the trunk, and when a commit happens, it kicks off one or more "builds", and reports the results.

<*buildbot*> *Hey! andrew broke the build!*

Twisted uses Buildbot.

`http://buildbot.sourceforge.net/`

Coping with inter-dependent projects

Launchpad depends on many other projects:

- Twisted
- SQLObject
- Zope 3
- and so on. . .

Coping with inter-dependent projects

In many cases, we need development versions, or heavily patched versions.

Coping with inter-dependent projects

In many cases, we need development versions, or heavily patched versions.

Each time we patch or upgrade, there's a risk of breaking Launchpad trunk.

Coping with inter-dependent projects

In many cases, we need development versions, or heavily patched versions.

Each time we patch or upgrade, there's a risk of breaking Launchpad trunk.

Which means tests don't pass, which means PQM rejects *all* our commits. Oops.

Coping with inter-dependent projects

So, we manage these dependencies with PQM too.

For example, if we update our Twisted, PQM runs the Twisted test suite *and* the Launchpad test suite before accepting the commit.

Mandatory Code Reviews

Reviews improve code quality

The usual reasons for code reviews all apply in a distributed team. They can catch:

- bugs
- coding style inconsistencies
- unreadable code
- insufficiently tested code

Reviews spread knowledge

Distributed teams don't:

- gather around watercoolers
- eat lunch together
- overhear conversations in the next cubicule

Reviews spread knowledge

Code reviews help spread knowledge around the team that might not otherwise spread.

- infrastructure that can be reused: “You can use FooHelper here instead of writing that yourself.”
- infrastructure that is needed: “That’s the third time someone has had write that.”

People working on a common project tend to bump into common problems and solutions.

How to manage reviews

All developers should be able read and comment on reviews, whether on a mailing list or in a bug/ticket.

How to manage reviews

All developers should be able read and comment on reviews, whether on a mailing list or in a bug/ticket.

Launchpad uses a wiki page and cron job to pre-generate diffs for reviewers. Robert assigns reviews to reviews daily. Reviews are sent to a mailing list.

How to manage reviews

All developers should be able read and comment on reviews, whether on a mailing list or in a bug/ticket.

Launchpad uses a wiki page and cron job to pre-generate diffs for reviewers. Robert assigns reviews to reviews daily. Reviews are sent to a mailing list.

Bazaar uses its mailing list, and website called "BundleBuggy" helps track the review requests and responses.

How to manage reviews

All developers should be able read and comment on reviews, whether on a mailing list or in a bug/ticket.

Launchpad uses a wiki page and cron job to pre-generate diffs for reviewers. Robert assigns reviews to reviews daily. Reviews are sent to a mailing list.

Bazaar uses its mailing list, and website called “BundleBuggy” helps track the review requests and responses.

Twisted uses its Trac ticket tracker. Every branch has a ticket, tickets with the “review” keyword are automatically in the review queue.

Bazaar: Distributed Version Control

What is distributed version control?

See previous talk!

Branching and merging

Distributed Version Control tools all have excellent branching and merging support.

This is great for distributed teams. Each developer can have a branch for each feature or bug they work on, and can merge back-and-forth with other developers easily.

Feature branches

Good branching and merging support encourages *feature branches*.

Feature branches

Good branching and merging support encourages *feature branches*.

Feature branches have a single, specific purpose: “Add feature X”, or “Fix bug #1234”.

Feature branches

Good branching and merging support encourages *feature branches*.

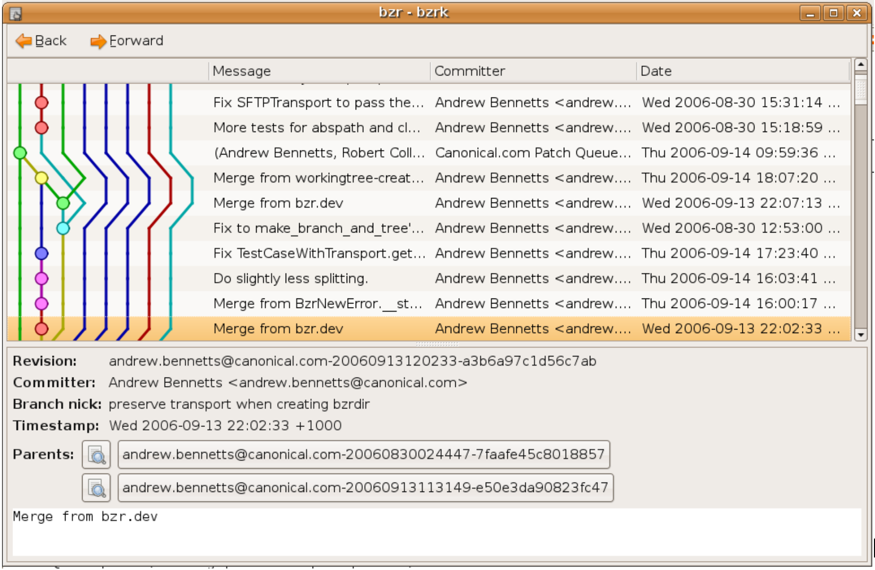
Feature branches have a single, specific purpose: “Add feature X”, or “Fix bug #1234”.

Simpler, single-purpose branches are easier to review.

Smaller commits

Because commits are usually on a branch, rather than on the trunk, you do smaller commits.

This helps tools like “annotate”.



Message	Committer	Date
Fix SFTPTransport to pass the...	Andrew Bennetts <andrew....	Wed 2006-08-30 15:31:14 ...
More tests for abspath and cl...	Andrew Bennetts <andrew....	Wed 2006-08-30 15:18:59 ...
(Andrew Bennetts, Robert Coll...	Canonical.com Patch Queue...	Thu 2006-09-14 09:59:36 ...
Merge from workingtree-creat...	Andrew Bennetts <andrew....	Thu 2006-09-14 18:07:20 ...
Merge from bzr.dev	Andrew Bennetts <andrew....	Wed 2006-09-13 22:07:13 ...
Fix to make_branch_and_tree'...	Andrew Bennetts <andrew....	Wed 2006-08-30 12:53:00 ...
Fix TestCaseWithTransport.get...	Andrew Bennetts <andrew....	Thu 2006-09-14 17:23:40 ...
Do slightly less splitting.	Andrew Bennetts <andrew....	Thu 2006-09-14 16:03:41 ...
Merge from BzrNewError.__st...	Andrew Bennetts <andrew....	Thu 2006-09-14 16:00:17 ...
Merge from bzr.dev	Andrew Bennetts <andrew....	Wed 2006-09-13 22:02:33 ...

Revision: andrew.bennetts@canonical.com-20060913120233-a3b6a97c1d56c7ab
Committer: Andrew Bennetts <andrew.bennetts@canonical.com>
Branch nick: preserve transport when creating bzrdir
Timestamp: Wed 2006-09-13 22:02:33 +1000
Parents: andrew.bennetts@canonical.com-20060830024447-7faafe45c8018857
andrew.bennetts@canonical.com-20060913113149-e50e3da90823fc47

```
Merge from bzr.dev
```

Why Bazaar?

- it's written in Python

CANONICAL

Why Bazaar?

- it's written in Python
- it's easy to use

Why Bazaar?

- it’s written in Python
- it’s easy to use
- the “uncommit” command

Why Bazaar?

- it's written in Python
- it's easy to use
- the "uncommit" command
- the "shelve" command (part of the bzrtools plugin <http://bazaar-vcs.org/BzrTools>)

CANONICAL

Why Bazaar?

- it's written in Python
- it's easy to use
- the "uncommit" command
- the "shelve" command (part of the bzrtools plugin <http://bazaar-vcs.org/BzrTools>)
- PQM already works with it.

The End
(almost)

Stuff I didn't talk about

Stuff I haven't mentioned — but probably should have:

- **VOIP:** talking is better than email and IRC for some things.
- **distributed pair programming:** I hear screen & VOIP work well.

CANONICAL

Summary

- Require — and enforce — that the trunk *always* builds and passes tests.
- Require code reviews to commit to the trunk.
- Use a distributed revision control tool.

Questions?

Questions?