Version Management with Bazaar
A helping hand for tracking changes in your files

Joakim Lindblad
joakim@cb.uu.se
Centre for Image Analysis
2007-10-29

Outline

What is Bazaar?
• What is Bazaar?
• What is Bazaar good for?
• A small sample session
• Getting started with Bazaar
• More features...
• Conclusions

What is version control?
• http://en.wikipedia.org/wiki/Revision_control

Bazaar is a distributed version control system available under the GPL that reduces barriers to participation in your project.

What is version control good for?
A version control system keeps a history of the changes made to a set of files. It answers questions like:
• What has happened with this project since last week?
• Who changed what and when?
• What other changes did they make at the same time?
And can perform actions like:
• Take me back to how the project looked last Tuesday!
• Merge the changes Lisa did on the text with those that I did last night!
It keeps track of all changes made to the files of a project during its entire lifetime.

Why use Bazaar instead of...?
CVS, Subversion (svn), Visual SourceSafe, ...
Distributed revision control takes a peer-to-peer approach, as opposed to the client-server approach of centralized systems. Rather than a single, central repository on which clients synchronize, each peer's working copy of the codebase is a repository. Synchronization is conducted by exchanging patches (change-sets) from peer to peer.

**Bazaar priorities**

- Good performance
- Safe with your data – No reported data loss bugs in the past two years.
- Friendly – Bazaar "Just Works"
- Free – Bzr is available under the GPL v2 or later.
- Easy to integrate – Python API

Bazaar is designed to support GNU/Linux, UNIX, Windows and the Mac OS.

"If you can run Python 2.4, then you can run Bazaar."

**Comitting your changes**

```
$ bzr commit
```

$EDITOR: enter a log message

Alternatively:

```
$ bzr commit -m "Print Hello world"
```

modified main.c
Committed revision 6.

That is all! ☺

**A sample session**

Arriving to work...

```
$ cd my_project
$ bzr update
Tree is up to date at revision 5.
$ ls -A
.bzr Makefile main.c parser.c
```

hack away at `main.c' . . .

**Ok, there is more to it**

```
$ make
gcc -o glurk main.c parser.c
main.c: In function 'main':
main.c:2: warning: incompatible implicit declaration of built-in function 'printf'
grmbl... What went wrong?
```
What has changed?

Compare current situation with what is committed:

```
$ bzr diff
```

No changes!

Help!

```
$ bzr help diff
```

Usage: bzr diff [FILE...]

aliases: di, dif

Show differences in the working tree or between revisions.

... examples:

bzr diff
Shows the difference in the working tree versus the last commit
bzr diff -r1
Difference between the working tree and revision 1

$ bzr diff -r 5

=== modified file 'main.c'

- main.c 2007-10-28 23:39:20 +0000
+ main.c 2007-10-28 23:44:32 +0000
-1,1 +1,4
-int main() {return 0;}
+int main() {
+ printf("Hello world!");
+ return 0;
+}

Negative numbers count from the latest revision, -1 is the last committed revision.
You can also do things like bzr diff -r date:yesterday

Syntax

```
$ bzr [opts] command [opts] [args]
```

- Commands
  - update - Upgrade branch storage to current format.
  - commit - Commit changes into a new revision.
  - diff - Show differences in the working tree or between revisions.
  - help - how help on a command or other topic.

Take me back!

```
$ bzr uncommit
6 Joakim Lindblad 2007-10-29
Print Hello world
The above revision(s) will be removed.
Are you sure [y/N]? y
```

```
$ bzr status
modified:
main.c
```

Take me back for real!

```
$ bzr revert
M main.c $ bzr status
```

Removes changes made! Warning

```
$ bzr help revert
```

By default, any files that have been manually changed will be backed up first. (Files changed only by merge are not backed up.) Backup files have ".~#" appended to their name, where # is a number.

Puh!

Bazaar is nice to you 😊
How does it work?

As opposed to centralized revision control systems, Bazaar keeps the repository in the working directory, in the directory `.bzr`. We don’t need to know how data is stored in the repository. It keeps all information from the beginning of times! 😊

Where do I find information?

More useful commands

• `bzr help commands` - lists all commands
• `bzr init` - create new project
• `bzr add [file]` - add new file
• `bzr log` - show project log
• `bzr whoami` - show or set user id
• `bzr annotate` - show the origin of each line in a file
• `bzr mv` - move or rename a file
• `bzr ignore file` - ignore file

Solo workflow

1. create project
2. record changes
3. browse history
4. package release

Partner workflow

1. start project
2. `bzr branch`
3. record changes
4. merge changes from peer
5. merge changes
6. merge changes from peer

Working with others

• `branch` - Create a new copy of a branch.
• `merge` - Perform a three-way merge.
• `resolve` - Mark a conflict as resolved.
To intelligently merge changes, the content of the "best common ancestor" is considered.

Given ancestor A and branches B and C:
- $A = B = C$ then line unchanged
- $A = B \neq C$ then take line from C
- $A = C \neq B$ then take line from B
- $A \neq B \neq C$ then conflict!

Some merges can only be completed with the assistance of a human.

The merge command reports these files as having conflicts and creates: a file with embedded markers showing the areas it couldn't resolve three files to assist you:
- foo.THIS
- foo.OTHER
- foo.BASE

After editing to clear up the mess:

```
$ bzr resolve
$ bzr commit -m "Merged to Hello and goodbye"
```

When disconnected from the master branch (e.g. on the road), commits can still be made locally: `bzr commit -local`
Migrating from other tools

I have successfully used Tailor to move from CVS to Bazaar.

http://progetti.arstecnica.it/tailor

Supports: AoX, Bazaar, Bazaar-NG, CVS, Codeville, Darcs, Git, Mercurial, Monotone, Perforce, Subversion and Tla repositories.

More features...

- Plugins — extending Bazaar's abilities
- Editor Integration — emacs, eclipse, ...
- GUIs — bzr-gtk, tortoiseBZR, nautilus-bzr

Bazaar at CBA

Add the following to your environment (~/.login or ~/.bash_profile)

- PYTHONPATH = /usr/local/lib64/python
- PATH = /usr/local/bin
- MANPATH = /usr/local/man

I've written two scripts

- joakim/local/noarch/bin/bzrdirs — print all Bazaar versioned directories
- joakim/local/noarch/bin/bzrall — run command in all Bazaar repositories

Example:

$ bzrall -r update

Conclusions

Working on larger projects, by yourself or with others, version control software can be a very useful tool.

I use Bazaar practically every day, because it helps me to keep order among my files. Maybe it can help you too 😊

http://bazaar-vcs.org/

These slides were made with the aid of wiki2beamer

http://wiki2beamer.sf.net