IceCube's Development Environment

Simon Patton
LBNL
Overview

• The IceCube Experiment.
• Requirements for our Development Environment.
• Tools selected and created to meet these requirements.
• Walk though of the user’s view of the environment.
The IceCube Experiment

• IceCube is a neutrino telescope.
  • 1 km$^3$ of ice instrumented with Digital Optical Modules, i.e. light detectors.
• 120 Collaborators.
• Base at the South Pole.
  • Only “Winter-over” access for 8 months
  • Satellite access intermittent.
• Software reliability a major asset.
Development Environment Requirements

Major focus on “best practice” approaches.

• Work space management.
  • Support some version of ‘Unified Change Management’ model.

• Java friendly build system.

• Unit testing support.

• Continuous build, test and report system.

• Integrated issue tracking.
Development Environment
Implementation

• No single, affordable, off the shelf, product offered all these features.
• Many open source tools addressed sub-sets of these features.
• IceCube selected those tools which it could use.
• Developed its own tool where no suitable tool was found.
  • Work space Management.
Work Space Management

- bfd (baseline file development) tool
  - Initialized work space with necessary files.
  - Wraps CVS to add policy.
Java friendly build system

• **ant** ([http://ant.apache.org/](http://ant.apache.org/)) is the standard build tool for Java products.

  • Written in Java.
    - Easily extendable.
    - Runs everywhere your Java code can run.

  • Rich collection of preexisting tasks.
    - Skeleton processing.
    - JUnit (see below) integration.
    - Other tools integration, e.g. JDepend. ([http://www.clarkware.com/software/JDepend.html](http://www.clarkware.com/software/JDepend.html))
• Provide users with a standard **build.xml**.

Main targets:

- **clean**: Clean all directories and files built for this project
- **compile**: Compile this project
- **createClass**: Create new .java files for a class and its matching test
- **createInterface**: Create new .java files for an interface and its matching test
- **createPackage**: Create a new package
- **docs**: Create API and test documents for this project
- **javadocs**: Create the Javadocs for this project
- **lib**: Create the library for this project
- **report**: Create the report on the tests run for this project
- **test**: Run the tests for this project
JUnit

• **junit** ([http://junit.org/](http://junit.org/)) is a Java implementation of the XUnit testing pattern.

• **Test skeletons are created by ant.**
  • User just needs to provide test implementations.

• **Text and GUI interfaces for standalone testing.**

• **ant can produce html reports.**
Continuous build system

• A Continuous build system is essential for finding problems early.

• CruiseControl ([http://cruisecontrol.sf.net/](http://cruisecontrol.sf.net/)) provides this for IceCube.
  - Runs standard ant build files.
  - Can be scheduled or watch CVS archive.
  - Web base results ([Click to see example](http://www.mozilla.org/tinderbox.html)).
  - Output can be used by tinderbox ([http://www.mozilla.org/tinderbox.html](http://www.mozilla.org/tinderbox.html)).
Walk through of the user environment

• **Create work space:**

  [patton@glacier patton]$ mkdir work
  [patton@glacier patton]$ cd work
  [patton@glacier work]$ bfd init /home/icecube/tools

  ... <output skipped> ...
  Initialized workspace, associated with the following tools directory.
  /home/icecube/tools

  [patton@glacier work]$ ls -l
  total 20
  -rw-rw-r-- 1 patton patton 291 Mar 22 15:28 build.xml
  drwxrwxrxr-x 4 patton patton 4096 Mar 22 15:28 resources
  -rw-rw-r-- 1 patton patton 3692 Mar 22 15:28 setup.csh
  -rw-rw-r-- 1 patton patton 3669 Mar 22 15:28 setup.sh
  drwxrwxrxr-x 4 patton patton 4096 Mar 22 15:28 tools

  [patton@glacier work]$ source setup.sh
Walk through of the user environment

• **Checkout and build icebucket library:**

  [patton@glacier work]$
  \$ \text{bfd co icebucket}
  
  ... <output skipped> ...

  [patton@glacier icebucket]$
  \$ \text{ant}

  Buildfile: build.xml
  
  ... <output skipped> ...

  BUILD SUCCESSFUL
  Total time: 7 seconds

  [patton@glacier icebucket]$
  \$ \text{cd ..}

  [patton@glacier work]$
  \$ \text{ls -l lib}

  total 8
  -rw-rw-r-- 1 patton patton 2594 Mar 22 15:54 icebucket.jar
  -rw-rw-r-- 1 patton patton 3032 Mar 22 15:54 icebucket-test.jar
Walk through of the user environment

• Create a new project:

```bash
[patton@glacier work]$ bfd co gromit
[patton@glacier work]$ ant -DPACKAGE=icecube.tools.examples \       
          -DPROJECT=gromit createProject

Buildfile: build.xml
  ... <output skipped> ...
BUILD SUCCESSFUL
Total time: 1 seconds
[patton@glacier work]$ ls -l gromit/
total 16
-rw-rw-r--  1 patton   patton        275 Mar 22 16:00 build.xml
-rw-rw-r--  1 patton   patton        292 Mar 22 16:00 project.xml
drwxrwxr-x  3 patton   patton       4096 Mar 22 16:00 resources
drwxrwxr-x  3 patton   patton       4096 Mar 22 16:00 src
[patton@glacier work]$ emacs gromit/project.xml
  ... <add dependency to icebucket>...
[patton@glacier work]$ bfd uadd gromit
```
Walk through of the user environment

• Create a new class:

[patton@glacier work]$ cd gromit/
[patton@glacier gromit]$ ant -DCLASS=Counter2 createClass
[patton@glacier gromit]$ ls -lR src/icecube/tools/examples/src/icecube/
tools/examples/:
  total 20
  -rw-rw-r--  1 patton   patton       4298 Mar 22 14:50 Counter2.java
  -rw-rw-r--  1 patton   patton        571 Mar  4 09:28 package.html
  drwxrwxr-x  3 patton   patton       4096 Mar 22 16:04 test

  src/icecube/tools/examples/test:
  total 16
  -rw-rw-r--  1 patton   patton       5182 Mar 22 14:50 Counter2Test.java
  -rw-rw-r--  1 patton   patton        415 Mar  4 09:26 package.html
Walk through of the user environment

• Write and test the new class.

[patton@glacier examples]$ cd src/icecube/tools/examples/
[patton@glacier gromit]$ emacs test/Counter2Test.java
  ... <write tests> ...
[patton@glacier examples]$ emacs Counter2.java
  ... <implement class>...
[patton@glacier examples]$ cd ../../../..
[patton@glacier gromit]$ ant lib
  ... <output skipped> ...
[patton@glacier gromit]$ cd ..
[patton@glacier gromit]$ java -cp lib/gromit-test.jar \ junit.textui.TestRunner icecube.tools.examples.test.Counter2Test
  ....
Time: 0.143

OK (4 tests)
Walk through of the user environment

• Archiving and delivering a project:

```
[patton@glacier work]$ bfd uadd gromit
... <output skipped> ...
[patton@glacier work]$ bfd ar -m "New example project" gromit
... <output skipped> ...
[patton@glacier work]$ bfd deliver -j gromit
Are you sure you want to deliver version V01-00-00 of "gromit"?
y/n: y
V01-00-00 of "gromit" has been delivered.
```
Walk through of the user environment

• Cleaning up a work space:

[patton@glacier work]$ bfd dispose gromit
No files have been added to, or modified in, "gromit".
There are no unknown files in, "gromit".
Disposed of "gromit".

[patton@glacier work]$ bfd dispose
Are you sure you want to dispose of the entire workspace?
y/n: y
No files have been added to, or modified in, "icebucket".
There are no unknown files in, "icebucket".
Disposed of "icebucket".
Disposed of workspace files...anything left is your own problem.

[patton@glacier work]$ ls -l
total 0
Summary

• IceCube wanted a development environment that encouraged good software process.

• No single, affordable, product provides all of this.

• There are plenty of open source tools that provide parts of this.

• By creating bfd and providing some “standard” files, IceCube has provided an environment that meets its needs.