

Prototype Development Environment and Analysis Framework

Simon Patton
(LBNL)

Overview

- **Reprise of the what, and why, of a Development Environment.**
- **Walk through of current prototype D.E. and framework.**
- **Review of features to be added next.**

What is a Development Environment?

- **A set of tools that help create software.**
- **A set of procedures that provide overall organization to a software project.**
- **Can include one or all of the following:**
Editors, compilers, build system, source code archiving, configuration management, test system, release system, tracking systems.

Why We Should Have a Development Environment

- **Large (more than one person) projects need to have efficient communications.**
- **If all developers speak “the same language” communications are easier, faster and less prone to errors.**
- **Only need to invest the development time once.**

Each developer does not have to create one for themselves.

Current State of Development Environment

- Working on a prototype Development Environment in which to develop an Analysis Framework.

WARNING: All work presented here is still in the prototype stage!

- **Prototype implies:**
 - Limited functionality;
 - Interface and Implementation are, by no means, final;
 - Very “clunky” in places.

Review of the Development Process

- **User creates a “Work Space” in which to work on a specific change.**
- **Creates tests to verify modifications result in required change.**
- **Implements code so tests run successfully.**
- **Commits changes (tests and code) to software repository.**
- **“Delivers” changes to be integrated with the rest of the system.**

Where to Begin

- **Generate Work Space by making a directory and then uncompressing proto-x.y.z.tar.gz**
 - `gunzip -cf <file> | tar -x`
 - **Version numbering XX-YY-ZZ:**
 - XX: Defines the major features of the package/distribution;
 - YY: Minor changes to the public interface;
 - ZZ: Changes to the non-public interfaces and implementation.

What is in a Basic Work Space (1/2)

- **README.txt** - description of how to use the distribution.
- **setup.sh, setup.csh** scripts to set up session environmental variables.
- **build.xml** - Ant file to manage the Work-Space.
- **tools** directory - contains tools used by Development Environment.
- **template** directory - file templates.

What is in a Basic Work Space (2/2)

- **lib directory** - destination for .jar files.
- **docs directory** - destination for generated documentation files.
- **icecube, faye, amanda, etc. directories** - projects
 - Projects contain files used to generate .jar files, resources and documentation.

Current Set of Tools

- **Ant: a dependency and build system.**
- **JUnit: a unit testing framework for Java.**
- **Xalan: an XSLT processor for transforming XML documents.**
- **Jython: an interactive scripting tool for Java.**
- **Log4j: a completely configuration logging system for Java.**

Starting a New Project

- Start by creating Work Space.
- Set up local links to tools.
- Set up local environmental variables.
- Build base projects.
- Start a new project using **ant**.
 - `amanda.analysis.sjp` used as the default package in this example.
- Edit `package.html` of the default package.

Setting up a Work Space

```
[patton@acme patton]$ mkdir ws1
[patton@acme patton]$ gunzip -cf proto-0.1.0.tar.gz | tar -C ws1 -x
[patton@acme patton]$ ls -l ws1
total 52
drwxrwxr-x    4 patton  patton    4096 Jun  7 11:27 amanda
drwxrwxr-x    2 patton  patton    4096 Jun  7 11:28 docs
drwxrwxr-x    4 patton  patton    4096 Jun  7 11:27 faye
drwxrwxr-x    4 patton  patton    4096 Jun  7 11:27 icecube
drwxrwxr-x    2 patton  patton    4096 Jun  7 11:28 lib
-rw-rw-r--    1 patton  patton    8763 Jun  7 11:26 README.txt
[patton@acme ws1]$ cd ws1
[patton@acme ws1]$ ./icecube/resources/scripts/mklnks /usr/java/tools
[patton@acme ws1]$ ls -F tools
ant@  bin/  jython@  lib/
[patton@acme ws1]$ . ./setup.sh
[patton@acme ws1]$ ant -DPROJECT=icecube
[patton@acme ws1]$ ant -DPROJECT=faye
[patton@acme ws1]$ ant -DPROJECT=amanda
```

Building a Project with ant

```
[patton@acme ws1]$ ant -DPROJECT=sjp -DPACKAGE=amanda.analysis.sjp \  
-Dpackage_dir=amanda/analysis/sjp createProject  
Buildfile: build.xml  
createDefaultPackage:  
  
setProperty:  
  
createPackage:  
  [mkdir] Created dir: /home/patton/ws1/sjp/src/amanda/analysis/sjp/test  
  [copy] Copying 1 file to /home/patton/ws1/sjp/src/amanda/analysis/sjp  
  [copy] Copying 1 file to /home/patton/ws1/sjp/src/amanda/analysis/sjp/  
test  
  
setProperty:  
  
createProject:  
  [mkdir] Created dir: /home/patton/ws1/sjp/resources  
  [copy] Copying 2 files to /home/patton/ws1/sjp  
  
BUILD SUCCESSFUL  
Total time: 5 seconds  
[patton@acme ws1] emacs sjp/src/amanda/analysis/sjp/package.html &
```

Sidebar: The Frame, Stream, Stop Model (1/2)

- Readout of detector equivalent to electronic “movie” of the detector.
- Analysis based on instance (small slice) in time - a *Frame* in the movie.
- Frame is composed of many features, many of which have different “lifetimes”.
- Conceptually related items with the same “lifetime” grouped into a *Record*.

Sidebar: The Frame, Stream, Stop Model (2/2)

- **Frame is composed of a set of Records**
 - e.g. Detector, Calibrations, Event.
- **A set of the same Records with different *SyncValues* (time) make up a *Stream*.**
- **Analysis executes whenever a new *Stop* occurs.**
 - Stop is a Stream and SyncValue.
 - Occurs whenever a selected Stream has a new Record.

Example Analysis

- **Most Amanda analyses will subclass `AmandaPlugInAdapter`.**
 - (Adapter is empty implementation of an interface, in this case `AmandaPlugIn`.)
- **Then override methods representing Streams of interest.**
 - e.g. `event()`, `detector()`.
- **Example will print Azimuth of a Fit.**

Creating a new Class

- **Use `ant` to create class and test files.**

(The modifications needed to run the first steps are supplied as 'patch' file listing. These can either be applied by editing or by running patch.)

- **Set up tests for the new class.**

- **Subclass `AmandaPlugInAdapterTest` as class is subclass off matching class.**

- **Otherwise subclass `TestCase`.**

- **Check subclassing works.**

- **Add tests for responsibilities of class.**

Patch file for project.properties

```
--- sjp/project.properties
+++ sjp/project.properties
@@ -2,7 +2,8 @@
  DEFAULT_PACKAGE = amanda.analysis.sjp
  default_package_dir = amanda/analysis/sjp
- all.dependencies = log4j.jar,junit.jar
- project.dependencies = ../tools/lib/log4j.jar
- test.dependencies = ../tools/lib/junit.jar
+ all.dependencies = faye.jar faye-test.jar amanda.jar,amanda-test.jar,\
+ log4j.jar,junit.jar
+ project.dependencies = amanda.jar ../tools/lib/log4j.jar
+ test.dependencies = amanda-test.jar ../tools/lib/junit.jar
  project.packages = amanda.analysis.sjp
  test.packages = amanda.analysis.sjp.test
```

Patch file for test (1/2)

```
--- sjp/src/amanda/analysis/sjp/test/AzimuthMonitorTest.java
+++ sjp/src/amanda/analysis/sjp/test/AzimuthMonitorTest.java
002
@@ -12,4 +12,6 @@

import junit.framework.*;
+import amanda.analysis.frame.test.AmandaPlugInAdapterTest;
+import amanda.analysis.sjp.AzimuthMonitor;

/**
@@ -19,5 +21,5 @@
    * @author patton
    */
-public class AzimuthMonitorTest extends TestCase {
+public class AzimuthMonitorTest extends AmandaPlugInAdapterTest {

    // public static final member data
@@ -59,8 +61,14 @@
    // protected instance member function (ctor first then alphabetic)

-//    /**
-//    * Sets up the fixture, for example, open a network
connection.
```

Patch file for test (2/2)

```
-//      * This method is called before a test is executed.
-//      */
+      /**
+      * Sets up the fixture, for example, open a network connection.
+      * This method is called before a test is executed.
+      */
+      protected void setUp() {
+          _monitor = new AzimuthMonitor();
+          setAdapter(_monitor);
+          super.setUp();
+      }
+
+      //      protected void setUp() {
+      //      }
@@ -94,4 +102,4 @@
+      // private instance member data
+      /** The object being tested. */
-      private AzimuthMonitor _object;
+      private AzimuthMonitor _monitor;
+      }
```

Patch file for AzimuthMonitor.java

```
--- sjp/src/amanda/analysis/sjp/AzimuthMonitor.java
+++ sjp/src/amanda/analysis/sjp/AzimuthMonitor.java
@@ -11,4 +11,6 @@
    package amanda.analysis.sjp;

+import amanda.analysis.frame.AmandaPlugInAdapter;
+
+/**
+ * This class ...does what?
@@ -17,5 +19,5 @@
+ * @author patton
+ */
- public class AzimuthMonitor {
+ public class AzimuthMonitor extends AmandaPlugInAdapter {

        // public static final member data
@@ -42,5 +44,5 @@
        * creation of an instance of the class.
        */
-    private AzimuthMonitor() {
+    public AzimuthMonitor() {
    }
```

Running Initial Test for the New Class

```
[patton@acme ws1]$ ant -DPROJECT=sjp -DCLASS=AzimuthMonitor createClass
[patton@acme ws1]$ patch -p0 < properties.diff
[patton@acme ws1]$ patch -p0 < test.diff
[patton@acme ws1]$ patch -p0 < monitor.diff
[patton@acme ws1]$ ant -DPROJECT=sjp
Buildfile: build.xml

    < CUT >

multitest:
    [junit] Running amanda.analysis.sjp.test.AzimuthMonitorTest
    [junit] Tests run: 11, Failures: 0, Errors: 0, Time elapsed: 0.84 sec

singletest:

test:

report:
[junitreport] Using Xalan version: Xalan Java 2.2.D11
[junitreport] Transform time: 11918ms

BUILD SUCCESSFUL

Total time: 33 seconds
```

Adding the “real” Test for the Class

```
public void testOutput() {
    testConfigureFromReady();
    prepareFrameSupplied();
    Fit fit = new Fit(FIT_NAME,
                    "",
                    new Position(0F,0F, 0F),
                    0F,
                    TEST_AZIMUTH,
                    0F,
                    0F,
                    0F);
    getAmandaFrame().put(fit,
                        FIT_NAME);
    // Set up VerifyAppender to test the monitors output
    String[] messages = new String[] {"Azimuth of Fit is " + TEST_AZIMUTH};
    addVerifyAppender(messages);
    // Supply the Frame
    _monitor.frameSupplied(getFrameToken(),
                        EVENT_STOP);
    // Remove the VerifyAppender which confirms the output.
    removeVerifyAppender();
}
```

Implementing Enough to Pass the Test

```
public static final String FIT_NAME_KEY =
    "amanda.analysis.example.AzimuthMonitor.fitname";

public void configure(Configuration configuration) {
    _fitName = configuration.getProperty(FIT_NAME_KEY);
}

public void event(AmandaFrame frame,
    Stop stop) {

    // Get the Fit from the Frame if it exists and log the azimuth.
    Fit fit = (Fit) frame.get(Fit.class,
        _fitName);

    if (null != fit) {
        getLogger().info("Azimuth of Fit is " + fit.getAzimuth());
    }
}

private String _fitName;
```

Run real test:

```
[patton@acme ws1]$ ant -DPROJECT=sjp
```

Run the new Class in a Job Using Java

• Create main to run the Job

```
public static void main(String[] args) {
    AmandaFaye faye = new AmandaFaye();
    faye.setProperty(AzimuthMonitor.FIT_NAME_KEY,
                    "1");
    String resource = "test/fiveReal.f2k";
    URL url = faye.getClass().getClassLoader().getResource(resource);
    faye.setProperty(F2kSource.FILE_NAME_KEY,
                    url.toString());
    F2kSource source = new F2kSource();
    faye.addAmandaSource(source);
    AzimuthMonitor plugin = new AzimuthMonitor();
    faye.addAmandaPlugIn(plugin);
    faye.supplyFrames(10);
    faye.dispose();
}
```

• Run Java from the Prompt.

```
[patton@acme ws1]$ java -cp lib/amanda.jar:lib/amanda.jar \
amanda.example.sjp.RunAzimuthMonitor
```

Run the new Class in a Job Using Jython

- **Create Jython script (or type in as `stdin`)**

```
import java
import amanda
faye = amanda.analysis.frame.AmandaFaye()
faye.setProperty(amanda.analysis.example.AzimuthMonitor.FIT_NAME_KEY, "1")
res = java.lang.String("test/fiveReal.f2k")
url = faye.getClass().getClassLoader().getResource(res)
faye.setProperty(amanda.format.f2k.F2kSource.FILE_NAME_KEY, url.toString())
source = amanda.format.f2k.F2kSource()
faye.addSource(source)
plugin = amanda.analysis.example.AzimuthMonitor()
faye.addPlugIn(plugin)
faye.supplyFrames(-1)
faye.dispose()
```

- **Run Jython.**

```
[patton@acme ws1]$ jython RunAzimuthMonitor.py
```

Caveat: Need complete CLASSPATH

Features to be Added

- **Continuous build system.**
- **Installable distributions.**
- **Baseline File Development system.**
- **Integration testing.**
- **Faye integration to JAS.**
- **“Properties” driven version of Faye.**
- **Clean up much of the clunkiness.**
- **Fill out implementations.**

Summary

- **Review of the what a Development Environment means.**
- **Walked through a prototype D.E. and analysis framework.**
- **Reviewed what features are missing and need to be added.**