Beyond Unit Testing

Steve Loughran
Julio Guijarro
HP Laboratories, Bristol, UK

steve.loughran at hpl.hp.com
julio.guijarro at hpl.hp.com

ApacheCon Europe 06

About Us

Julio Guijarro

Research scientist at HP Laboratories on
Grid-Scale Deployment

Leads the SmartFrog open source effort

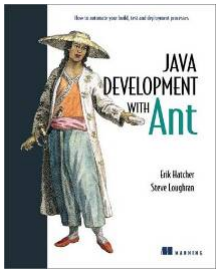
Steve Loughran

Research scientist at HP Laboratories on
Grid-Scale Deployment

Apache Ant & Axis committer

Co-author of
Java Development with Ant

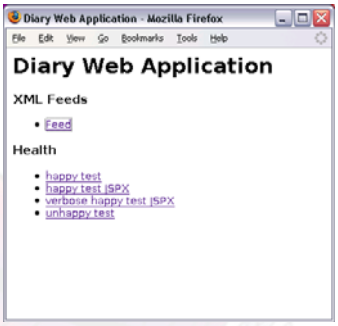
Behind schedule on the 2nd edition



ApacheCon Europe 06

two different distributed systems

CERN Large Hadron Collider



Multi-tier webapp

How do you show it works?

- Europe's high-end server farms
- Years of simulations
- Nobel Prize winners, Computer Scientists and physics PhD students

- An old laptop nobody wants
- Any spare time before you ship
- You

Classic unit tests

- Run in a test harness
- Don’t stress the system
- Don’t run on real servers
- Don't run with real data

A modest proposal

Write less Unit Tests!

Apply Formal Methods!

- Integrating *Formal Methods* with XP development.
- How to use *axiomatic theorem proofs* to verify correctness in a large-scale distributed system.
- How Milner's *π-calculus* is the underpinnings for the BPEL workflow language.
- *Continuations* vs. *bisimilar state machines* -which is better for correctness proofs?
- How relaxing your *concurrency constraints* results in higher throughput.

Or:
System Testing

System Tests

- Deploy the app
- Add a real dataset
- Use the app server
- Remotely test from other sites/hosts
- Test in the client
- Are big, complex and distributed

How to test big systems

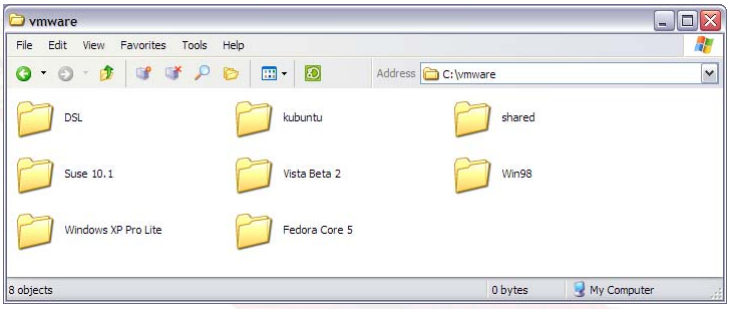
- Simulate the production system.
- Automate deployment
- Write functional tests
- Remote test from clients

Embrace Virtualization

- VMWare player free; workstation for $£€
- Create VM images that resemble production configurations.
- Deploy and test into virtual machines
- Host continuous integration server in VMs
- Simulate complex/broken networks

Download VMware Player

…and become a cluster admin



- PXE System Installers: linuxcoe.sf.net
- Auto-rollback images during test *and* production
- Isolate insecure platforms on virtual network

Simulate the production system
Automate deployment
Write functional tests
Remote test from clients

Automate app deployment

- RPM/APT/.msi packages pushed out to hosts
- *SmartFrog*: http://smartfrog.org/
- *Cargo*: http://cargo.codehaus.org
- Shell Scripts
- Ant build files using scp, ssh

Database setup

- Data setup is too time consuming to do every test
- Use the same DB that production will have.
- Automated set up of the database
- keep this DB snapshot and revert to it after a run. (or the entire virtual machine image)

```
<mysql-admin>
  CREATE DATABASE diary;
  GRANT ALL PRIVILEGES ON diary.*
    TO 'diary'@'localhost';
  SET PASSWORD FOR 'diary'@'localhost' =
    PASSWORD('${mysql.diary.pass}');
</mysql-admin>
```

Simulate the production system
Automate deployment
Write functional tests
Remote test from clients

What to test?

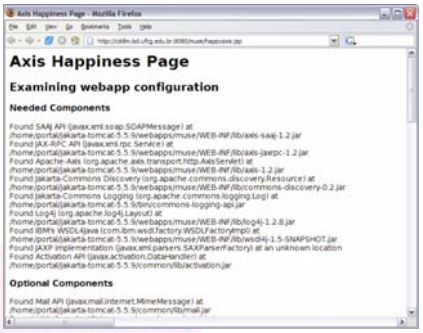
- system health tests
- In-container unit tests
- Remote web service/HTML tests
- In-browser GUI testing
- Load tests
- Network failure simulations

...

ApacheCon Europe 06

Health Test: “happy pages”

```
<%@ taglib uri="/WEB-INF/diary.tld"
prefix="h" %>
<body>
<ha:happy
 classMustExist="org.jdom.JDOMException"
 errorText="JDom missing"/>
We are happy
</body>
</html>
```



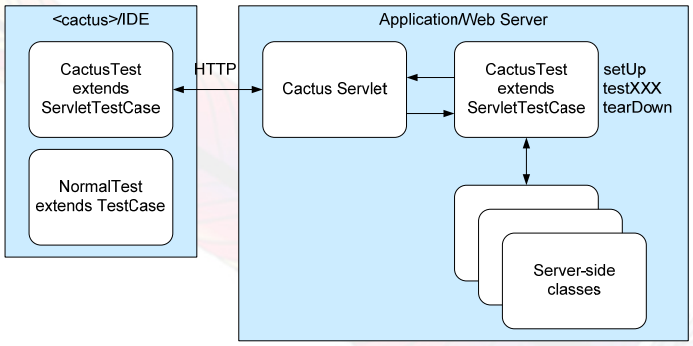
Delegate to machines:

```
<waitfor maxwait="30" maxwaitunit="second"
    timeoutproperty="unhappy">
  <http url="http://server/happyaxis.jsp"/>
</waitfor>
<fail if="unhappy"/>
```


ApacheCon Europe 06

Test in-container with cactus

Simulate the production system
Automate deployment
Write functional tests
Remote test from clients



Cactus Test Case

Simulate the production system
Automate deployment
Write functional tests
Remote test from clients

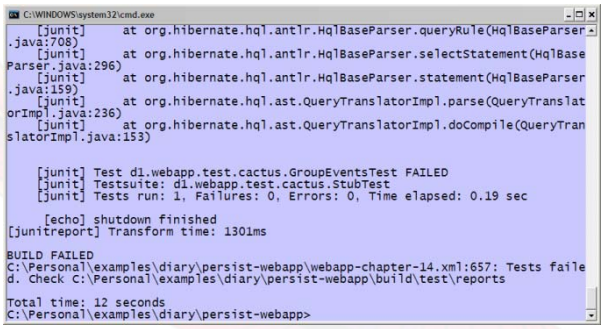
```
public class CactusPersistTest extends ServletTestCase {
    private static int counter = 0;
    private SessionFactory factory;

    public void testPersist() throws Exception {
        Event event = createTestEvent();
        Session session = factory.openSession();
        try {
            session.persist(event);
        } finally {
            session.close();
        }
        assertEventIsInDB(event);
    }
}
```

Simulate the production system
Automate deployment
Write functional tests
Remote test from clients

<cactus> task *choreographs*

```
<cactus:cactus warfile="${cactus.war}"
  errorProperty="cactus.failed" failureProperty="cactus.failed">
  <containerset>
    <generic name="server" port="8080">
      <startup>
        <copy file="${cactus.war}" tofile="${cactus.destfile}"
          overwrite="true"/>
      </startup>
      <shutdown>
        <delete file="${cactus.destfile}"/>
      </shutdown>
    </generic>
  </containerset>
  <classpath><path refid="test.classpath"/></classpath>
  <formatter type="xml"/>
  <batchtest todir="${test.data.dir}">
    <fileset dir="test" includes="**/*Test.java">
  </batchtest>
</cactus:cactus>
```


Cactus Demo



- Needs classpath right for client and server
- cactus servlet is possible security risk

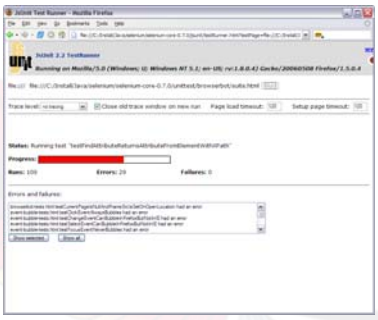
Simulate the production system
Automate deployment
Write functional tests
Remote test from clients

GUI testing hurts

- Static HTML is the easiest (HttpUnit)
- Swing, DHTML, SWT, Flash hard.
- Most people stop at the “model”
- Whoever does a new GUI -fix this!

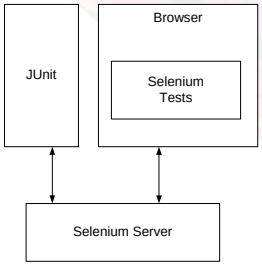
jsUnit is JUnit for JavaScript

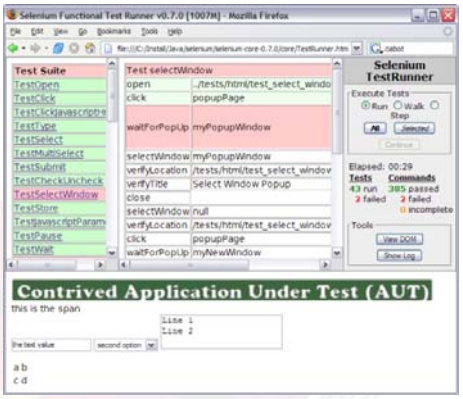
```
function test3() {
  var buffer = top.testManager.documentLoader.buffer();
  var emps = buffer.document.getElementsByTagName('employee');
  assert('expected 5 employees, not ' + emps.length,
    emps.length == 5);
  var empid = emps[0].getElementsByTagName('employeeId');
  assert('employeeId[0] was  '
    + empid[0].firstChild.data,
    empid[0].firstChild.data == 'EMP0001');
}
```



Selenium: tests in a table

```
<tr>
  <td>verifyTitle</td>
  <td>Click Page Target</td>
  <td> </td>
</tr>
```





http://www.openqa.org/selenium/

ApacheCon Europe 06

WS Interop Testing

Simulate the production system
Automate deployment
Write functional tests
Remote test from clients

- Use the real client API/classes
- Pass down URLs via system properties

```
protected String getOption(String property,
      boolean required) {
  String option = System.getProperty(property);
  if (required && option== null) {
    fail("No property " + property);
  }
  return option;
}
```

- Test different endpoints in parallel processes
- Include timeouts; proxy support
- Log for blame assignment

ex: http://deployapi.iseran.com:8080/logs/

ApacheCon Europe 06

Simulate the production system
Automate deployment
Write functional tests
Remote test from clients

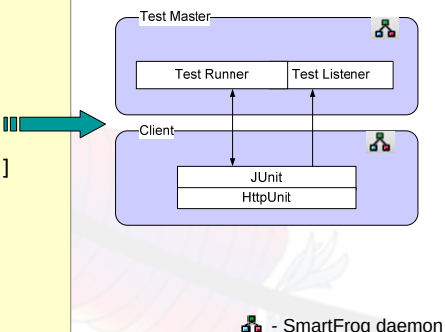
Distributed Testing

- Allocate & configure test systems
- Deploy application across nodes
- Deploy tests on other nodes
- Collect and correlate results
- Try to understand what went wrong

SmartFrog

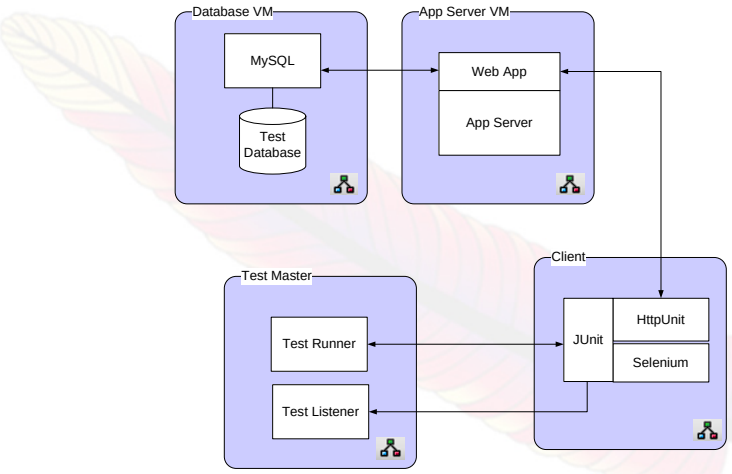
A framework for describing, deploying and managing distributed service components.

```
HttpUnitTests extends JUnitTestSuite {
 package "d1.webapp.test";
    name "HttpUnitTests";
  server.url TBD;
  sfProcessHost "client";
  properties [
    ["server.url",server.url],
    ["cactus.contextURL",server.url]
  ];
  classes [
    "EventFeedTest",
    "HappyTagTest",
    "IndexTest"
    ];
}
```



http://smartfrog.org/

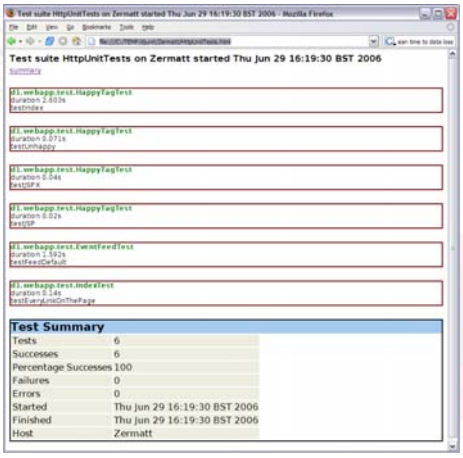
Distributed Deployment of App & JUnit



- SmartFrog daemon

ApacheCon Europe 06

XHTML output of test results

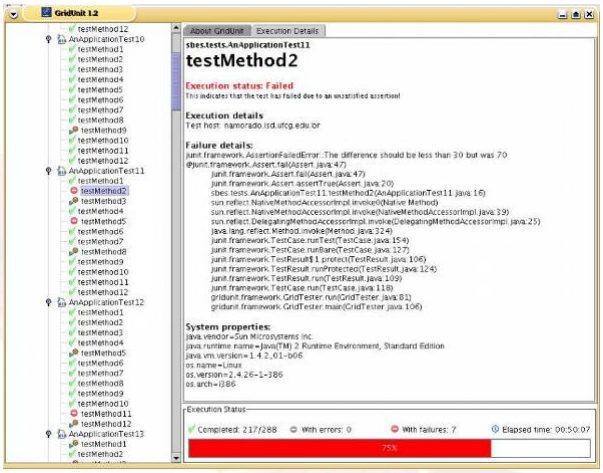


+ ~live output
+ log capture
- no x-system summary
- no merging of logs from different systems
- no notification

ApacheCon Europe 06

Future GUI? GridUnit

- Swing GUI for testing on OurGrid
- Unit test across many different machines
- But not (yet) distributed applications
- Aggregate view of results
- “partial” success
- Common JUnit wire format



http://gridunit.sourceforge.net/

ApacheCon Europe 06

Call to Action

- Focus on system tests
- Embrace Virtualization: VMWare, Xen
- Use Cactus for in-container testing
- Use Selenium/jsUnit for browser tests
- Join us in distributed system testing

ApacheCon Europe 06

Junit4?

- Java5 only
- Extension tools not there yet
- Integration with Ant, Maven coming along.
- Ant 1.7 <junit> will work with junit4.jar
- JUnit team plan their own task (Ant team are working with them)