A Guide To Using Continuous Integration Within The Verification Environment

Presented by: Paul Marriott, Verilab

Authors:
Gordon McGregor, Nitero
Jason Sprott, Verilab
André Winkelmann, Verilab
Jenkins & vManager

Jenkins server

www

Integration Code

HTML
JUnit

e API
vManager

TCL API
vManager CS

Two Examples
Jenkins(CI) Refresher

1. check in
2. indicate change
3. check out & run
4. results
5. notify & feedback

See: DVCON 2012 “30 Minute Project Makeover Using Continuous Integration”, Gray & McGregor
CI vs MDV

CI (Jenkins)
• Code health
• Keep main branch passing
• Release early & often
• Quickly spot and fix issues
• Integrated with SCM
• Automates test execution
• Communicates status
• PASS/FAIL dashboards

MDV (vManager)
• Finding RTL bugs
• Managing complexity
• Integrated with verification language features
• Verification closure
• Integrates with vplan
• Verification specific results
• Specialized analysis tools
Metric
Key CI Requirements

• Same results on Jenkins as standalone sims
  – Common PASS/FAIL parsing
  – If the build fails the whole regression is a FAIL
  – CSV or JUnit XML missing is a FAIL
• Regression must run in batch mode
• No return until all threads complete
Two Examples

- **Reusable:** `csv_to_junit_reporter.py`
- **Ex. 1 (vManager) using a e-language API**
  - Specific: initial launch script
  - Specific: e code to control exit and export results
- **Ex 2 (vManager CS) using TCL API**
  - Specific: initial launch script
  - Specific: TCL to control exit and export results
**Jenkins**

- JUnit Plug-in
- HTML Publisher Plug-in

**vManager**

- Regression Launch Script
- Regression Control & Result Output

**Convert CSV to JUnit XML (Python)**

- Export Per-Test Results (as CSV)
- Export HTML Reports

**Import to JUnit plug-in**

- Import to HTML Publisher plug-in

**Tool specific**

- batch_run_e.sh
- vlab_post_session_e
- csv_to_junit_reporter.py

- ~45 lines of code
- ~80 lines of code
- ~240 lines of code

**Build/Run**

- Return overall PASS/FAIL
CSV Output from vManager

```
Test Group, Test Name, Seed, SV Seed, Status, CPU Time (ms.), Log File, First Failure Description
vlab.e_tests, singleline_fail, 279134982, random, failed, 12.17, .../run_1/local_log.log, "Please Jenkins, show this error message!"
vlab.e_tests, passing, 1216575217, random, passed, 13.48, .../run_2/local_log.log.
vlab.e_tests, multiline_fail, 259841377, random, failed, 12.66, .../run_3/local_log.log, "Please Jenkins, show this error message! Let us see more details of the error. What could have gone wrong? ...
vlab.e_tests, sometimes_fail, 1266469697, random, failed, 13.54, .../run_4/local_log.log, "Unlucky, the test did fail!"
vlab.e_tests, sometimes_fail, 863826133, random, passed, 13.48, .../run_5/local_log.log,
vlab.e_tests, sometimes_fail, 511941856, random, passed, 12.30, .../run_6/local_log.log,
vlab.e_tests, sometimes_fail, 2105296490, random, passed, 12.20, .../run_7/local_log.log,
vlab.e_tests, sometimes_fail, 628890105, random, passed, 11.43, .../run_8/local_log.log,
```

in excel:

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Test Group</td>
<td>Test Name</td>
<td>Seed</td>
<td>SV Seed</td>
<td>Status</td>
<td>CPU Time (ms.)</td>
<td>Log File</td>
</tr>
<tr>
<td></td>
<td>vlab.e_tests</td>
<td>singleline_fail</td>
<td>279134982</td>
<td>random</td>
<td>failed</td>
<td>12.17</td>
<td>.../run_1/local_log.log</td>
</tr>
<tr>
<td>3</td>
<td>vlab.e_tests</td>
<td>passing</td>
<td>1216575217</td>
<td>random</td>
<td>passed</td>
<td>13.48</td>
<td>.../run_2/local_log.log</td>
</tr>
</tbody>
</table>
| 4     | vlab.e_tests           | multiline_fail | 259841377 | random | failed         | 12.66                | .../run_3/local_log.log | Please Jenkins, show this error message! Let us see more details of the error. What could have gone wrong? ...
| 5     | vlab.e_tests           | sometimes_fail | 1266469697 | random | failed         | 13.54                | .../run_4/local_log.log | Unlucky, the test did fail!                       |
| 6     | vlab.e_tests           | sometimes_fail | 863826133 | random | passed        | 13.48                | .../run_5/local_log.log |                               |
| 7     | vlab.e_tests           | sometimes_fail | 511941856 | random | passed         | 12.3                | .../run_6/local_log.log |                               |
| 8     | vlab.e_tests           | sometimes_fail | 2105296490 | random | passed        | 12.2                | .../run_7/local_log.log |                               |
| 9     | vlab.e_tests           | sometimes_fail | 628890105 | random | passed        | 11.43                | .../run_8/local_log.log |                               |
XML JUnit Report

Output from `csv_to_junit_reporter.py`

```xml
<?xml version='1.0' encoding='UTF-8'?>
<testsuites name='results'>
  <testsuite failures='3' id='0' name='vlab.e_tests' passes='5' time=''>
    < testcase classname='vlab.e_tests' name='singleline_fail' time='12.17'>
      < failure message='Please Jenkins, show this error message!' type='Fail'>
      </ failure>
    </ testcase>
    < testcase classname='vlab.e_tests' name='passing' time='13.48' />
    < testcase classname='vlab.e_tests' name='multiline_fail' time='12.66'>
      < failure message='Please Jenkins, show this error message!' type='Fail'>
      </ failure>
    </ testcase>
  </ testsuite>
</ testsuites>
```

Per-test results:
- Seed: 279134982
  - Log File: /path_to_test/e_tests/run_1/local_log.log
  - Complete Message: Please Jenkins, show this error message!

- Seed: 25984377
  - Log File: /path_to_test/e_tests/run_2/local_log.log
  - Complete Message: Please Jenkins, show this error message!

- Seed: 126469697
  - Log File: /path_to_test/e_tests/run_3/local_log.log
  - Complete Message: Please Jenkins, show this error message!

- Seed: 279134982
  - Log File: /path_to_test/e_tests/run_4/local_log.log
  - Complete Message: Please Jenkins, show this error message!

- Seed: 25984377
  - Log File: /path_to_test/e_tests/run_5/local_log.log
  - Complete Message: Please Jenkins, show this error message!

- Seed: 126469697
  - Log File: /path_to_test/e_tests/run_6/local_log.log
  - Complete Message: Please Jenkins, show this error message!

- Seed: 279134982
  - Log File: /path_to_test/e_tests/run_7/local_log.log
  - Complete Message: Please Jenkins, show this error message!

- Seed: 25984377
  - Log File: /path_to_test/e_tests/run_8/local_log.log
  - Complete Message: Please Jenkins, show this error message!

- Seed: 126469697
  - Log File: /path_to_test/e_tests/run_9/local_log.log
  - Complete Message: Please Jenkins, show this error message!

- Seed: 279134982
  - Log File: /path_to_test/e_tests/run_10/local_log.log
  - Complete Message: Please Jenkins, show this error message!
```

©2014 Verilab and DVCon
Per-Test Results for JUnit

Output from csv_to_junit_reporter.py

```xml
<testcase classname="vlab.e_tests" name="multiline_fail" time="12.66">
    <failure message="Please Jenkins, show this error message!" type="Fail">
        Seed: 259841377
        SV Seed: random
        Log File: SOME_PATH/run_3/local_log.log
        Complete Message:
        Please Jenkins, show this error message!
        Let us see more details of the error.
        What could have gone wrong?
        ... no idea ...
    </failure>
</testcase>
```
Test Results in Jenkins

Test Result

8 failures (+3)
8 tests (0)
Took 1 min 39 sec
add description

All Failed Tests

<table>
<thead>
<tr>
<th>Test Name</th>
<th>Duration</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>vlab_e_tests_singeline_tail</td>
<td>12 sec</td>
<td>5</td>
</tr>
<tr>
<td>vlab_e_tests_multiline_tail</td>
<td>13 sec</td>
<td>5</td>
</tr>
<tr>
<td>vlab_e_tests_sometimes_fail</td>
<td>12 sec</td>
<td>5</td>
</tr>
<tr>
<td>vlab_e_tests_sometimes_test</td>
<td>12 sec</td>
<td>5</td>
</tr>
<tr>
<td>vlab_e_tests_sometimes_test</td>
<td>13 sec</td>
<td>5</td>
</tr>
<tr>
<td>vlab_e_tests_sometimes_test</td>
<td>12 sec</td>
<td>5</td>
</tr>
</tbody>
</table>

All Tests

<table>
<thead>
<tr>
<th>Package</th>
<th>Duration</th>
<th>Fail</th>
<th>(diff)</th>
<th>Skip</th>
<th>(diff)</th>
<th>Total</th>
<th>(diff)</th>
</tr>
</thead>
<tbody>
<tr>
<td>vlab</td>
<td>1 min 30 sec</td>
<td>6</td>
<td>-3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Failed

viab_e_tests.singeline_fail

Failing for the past 4 builds (Since #1)
Took 11 sec.
add description

Error Message

Please Jenkins, show this error message!

Stacktrace

Seed: 1236044349
SV Seed: random
Log Files: https://jenkins/job/vManager_e/ws/vmanager/regression/vlab.jenkins.14_02_24_08_59_37_6952/chains/e_tests/run_j/local_log.log
https://jenkins/job/vManager_e/ws/vmanager/regression/vlab.jenkins.14_02_24_08_59_37_6952/chains/e_tests/run_j/precomp.elog
Complete Message:
Please Jenkins, show this error message!
Running vManager

From: vmanager/batch_run_e.sh

```bash
vm_launch.pl \
  -vsif ./e/e_tests.vsif \
  -batch \ 
  -command "load scripts/vlab_post_session.e; sys.vlab_post_session_analysis();" \ 
  -output_mode log_only \ 
  -regr_report

python ../scripts/csv_to_junit_reporter.py regression_tests.csv junit.xml
```

Some messing around for HTML reports and final result

```bash
if [ -e "$MY_REGRESSION_AREA/$SESSION_DIR/Regression_Report.html" ]; then
  rm -rf html_report
  mkdir html_report
  cp $MY_REGRESSION_AREA/$SESSION_DIR/Regression_Report.html html_report/index
  ln -s $MY_REGRESSION_AREA/$SESSION_DIR/Regression_Report.html.files html_report/Re
fi

grep PASS regression_result.txt
```
vManager e API

Excerpt from: vmanager/scripts/vlab_post_session.e

vlab_post_session_analysis() is {

    out("Waiting until all tests have been executed");
    vm_util.trace_all_sessions_to_finish();
    out("All tests should be done by now. Starting post session analysis.");

    vlab_vsof = vm_manager.get_all_sessions()[0];  // or load a specific vsof
    vlab_runs = vlab_vsof.get_runs();

    vlab_out_session_result();
    vlab_out_test_results();

};

There’s about 80 lines of supporting code in total
vManager e API

Overall PASS/FAIL result from vManager

```python
vlab_out_session_result() is {
    var session_failures: list of vm_failure = vlab_vsof.get_failures();
    var f:file = files.open("regression_result.txt", "w", "Overall session pass fail text file");
    var res:bool = TRUE;

    if !session_failures.is_empty() {
        res = FALSE;
    }

    for each in vlab_runs {
        if !it.has_passed() {
            res = FALSE;
            break;
        }
    }

    if res == TRUE {
        files.write(f, "PASS");
    } else {
        files.write(f, "FAIL");
    }

    files.close(f);
    out("Written regression_result.txt");
}
```
vManager e API

Individual test result from vManager written to CSV

```python
def vlab_out_test_results()
    var f: file = files.open("regression_tests.csv", "w", "Test list csv file");
    files.write(f, "Test Group,Test Name,Seed,SV Seed,Status,CPU Time (ms.),Log File,First Failure Des 
    for each in vlab_runs {
        var first_fail: string = it.get_attribute_value(vm_manager.get_attribute_by_name("first_failure")
        first_fail = str_replace(first_fail, "/", ",", ");
        files.write(f, append(#
            it.get_attribute_value(vm_manager.get_attribute_by_name("test_group")), ",", 
            it.get_attribute_value(vm_manager.get_attribute_by_name("test_name")), ",", 
            it.get_attribute_value(vm_manager.get_attribute_by_name("seed")), ",", 
            it.get_attribute_value(vm_manager.get_attribute_by_name("sv_seed")), ",", 
            it.get_attribute_value(vm_manager.get_attribute_by_name("status")), ",", 
            appendf("%.2f", it.get_attribute_value(vm_manager.get_attribute_by_name("cpu_time"))).as_a(real 
            it.get_attribute_value(vm_manager.get_attribute_by_name("log_file")), ",", 
            ",", first_fail, ")
        )
    };
    files.close(f);
    out("Written regression_tests.csv");
```

©2014 Verilab and DVCon
Running vManager CS

From: vmanagercs/batch_run_e.sh

```
45  vmanager -cs -profile \${MY_VMANNER_CS_PROFILE} -exec scripts/run.tcl
46  python ../scripts/csv_to_junit_reporter.py regression_tests.csv junit.xml
```

From: vmanagercs/scripts/run.tcl

```
1    launch -wait -load -vsif $::env(VLAB_TESTS)
2    csv_export -runs -view $::env(MY_VMANNER_CSV_VIEW) -out regression_tests.csv -overwrite
3    report_tests -out tests_report -overwrite
```

View created manually once (covered in README)
Jenkins Project Configuration

Build

Execute shell

Command

```bash
#!/bin/bash
# Set up the environment, e.g. with . /etc/bashrc
# let's care about any error from now on set -e
# Choose between vManager and vManager CS
# cd vmanagers
# For vManager CS you need to set the full path to the used server profile
#export MY_VMANAGER_CS_PROFILE=/path/to/vmanager/server/profile
# Choose between Specman e example or a Systemverilog example
./batch_run_e.sh
./batch_run_sv.sh
```

See the list of available environment variables

Post-build Actions

Publish HTML reports

<table>
<thead>
<tr>
<th>HTML directory to archive</th>
<th>Index page(s)</th>
<th>Report title</th>
<th>Keep past HTML reports</th>
<th>Allow missing report</th>
</tr>
</thead>
<tbody>
<tr>
<td>vmanager/html_report</td>
<td>index.html</td>
<td>HTML Report</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Add

Publish JUnit test result report

Test report XML: vmanager/junit.xml

Delete
csv_to_junit_reporter.py

```
csv_to_junit_reporter

junit_jenkins_reporter

<<imports>>
xml.etree.ElementTree
<<imports>>
csv
<<imports>>
Standard python packages for handling XML and CSV formats
```
CSV to JUnit Conversion

From: scripts/csv_to_junit_reporter.py

Standard CSV library and custom JUnit XML generator

```
17 from junit_jenkins_reporter import JunitReport
18 import csv
19 import sys
20 import re
21 import os
22
23 csv.register_dialect('escaped', escapechar='\', doublequote=False, quoting=csv.QUOTE_NONE)
...

28 def csv_to_junit(csv_filename, outfile):
29     '''Given a CSV format input file, generate a Jenkins compatible JUnit result'''
30     testsuits=[]
31     with open(csv_filename) as csv_file:
32         csv_data = csv.reader(csv_file, dialect='excel')
33         report = JunitReport()
```
CSV to JUnit Conversion

Populating custom JUnit attributes from CSV

```python
report.add_testcase(
    name=line[lu['Test Name']],
    testsuite=testsuites[test_group]['element'],
    classname=test_group,
    time="%.2f" % (float(line[lu['CPU Time (ms.)']])/1000))

if line[lu['Status']] == 'failed':
    testsuites[test_group]['failures'] += 1
    fail_desc = 'Seed: ' + line[lu['Seed']] + '\n'
    fail_desc += 'SV Seed: ' + line[lu['SV Seed']] + '\n'

    log_files = set(line[lu['Log File']].split('&<__SEPARATOR__>'))
    if "JOB_URL" in os.environ and "WORKSPACE" in os.environ:
        log_files = map(lambda f: f.replace(os.environ['WORKSPACE'] + '/\', os.environ['JOB_URL'] + \n
fail_desc += 'Log Files: ' + '\n'.join(log_files) + '\n'
    fail_desc += 'Complete Message: \n' + line[lu['First Failure Description']]])
    report.add_failure(
        type="Fail",
        msg_text=fail_desc,
        message=line[lu['First Failure Description']].split('\n')[0])

else:
    testsuites[test_group]['passes'] += 1
```
Junit XML Report Generator

From: scripts/junit_jenkins_reporter.py

```python
class JUnitReport:
    '''Build a JUnit format report file for Jenkins and/or Hudson.
    most of the kwargs allow you to pass name-value pairs straight into the tag, to become attributes

    def __init__(self):
        '''Class constructor, creates top level testsuites container for results'''

    def add_testsuite(self, **kwargs):
        '''Add a new testsuite and associated attributes [name, time]'''

    def add_testsuite(self, testsuite=None, **kwargs):
        '''Add a new testcase and associated attributes [name, time]'''

    def update_testsuite(self, testsuite=None, **kwargs):
        '''Add additional arguments to the testsuite tag (usually pass/ fail results as a a result of pre

    def add_log(self, logfile, testcase=None):
        '''Given a logfile, head/tail truncate it to TRUNCATE_LINES*2 length and store in a <system-out>

    def add_failure(self, msg_text='', testcase=None, **kwargs):
        '''Update the testcase, or last testcase if none provided to be a failure and add appropriate at

    def indent(self, elem, level=0):
        '''Pretty printer for the XML output'''

    def write(self, filename):
```

©2014 Verilab and DVCon
The CI Culture

- Rapid feedback
- Commit often
- Quick builds & tests
- Keep HEAD working
- Frictionless branching helps

Impact on MDV?
Getting The Code

https://bitbucket.org/verilab/jenkinsintegration

Location of csv_to_junit_reporter.py
Summary

• CI and MDV are complementary
• Two Jenkins integration examples shown
  – Same tool different APIs
• There are some integration requirements
  – e.g. Pass/Fail and exit handling are important
• We can leverage standard Jenkins plug-ins
  – CSV to JUnit reusable code provided