Cinderella 'TAP
The lazy evaluation sisters of TAP::Parser

Steffen Schwigon, AMD | August 04, 2009
YAPC::EU 2009
Cinderella 'TAP
The lazy evaluation sisters of TAP::Parser

Steffen Schwigon, AMD | Aug 04, 2009
YAPC::EU 2009
Agenda

- Context
  - Test automation
  - Participation model
  - Producing and consuming
- Problem
  - The no-problem
  - The Query Gap
- Solution
  - SYNOPSIS: Query Interface
  - Query language
  - Modules
  - Glue
Context
Our Mission

- AMD
- Operating System Research Center (OSRC)
- QA team → Testing
- OS x Virtualization x AMD hardware x OSRC patches
Target audience

- “Classical” end users
  - Web user-interface

- Developers
  - Command line advocates
  - But too busy to fiddle with complicated toolchain
Test Infrastructure - Key ideas

- KVM/Xen Test Automation
  → http://xrl.us/xentestautomation
  German Perl Workshop 2009

- TAP (“Test Anything Protocol”)
- Trivial reporting
- Reports archive and query framework

- “non-aristocratic” participation model
Non-aristocratic?

- Make the toolchain trivially accessible
  - Shell script level,
  - “no XML”,
  - netcat

- Non-interactive protocols (fire & forget reporting)
- Easy interactive protocols

- Scalable complexity – start easy, then escalate
  - Test protocol
  - Query language
TAP example

1..3

# Artemis-Suite-Name: oprofile
# Artemis-Suite-Version: 2.013

ok 1 - Looks like oprofile kernel

ok 2 - other stuff

not ok 3 - last line # TODO just specced

---

message: Failed test 'last line' at t/ltp.t line 317.

data:
  got: 'foo'
  expected: 'bar'

...
Easy with shell

```bash
#!/bin/sh
echo "1..2"
echo "# Artemis-Suite-Name: oprofile"
echo "# Artemis-Machine-Name: " `hostname`
if uname | grep -vq oprofile ; then echo -n "not " ; fi
echo "ok - Looks like oprofile kernel"
echo "ok - other stuff"
```

Same with C, Python, Perl, ...
- with or without toolchain
Report interface (1)

- ./test_script.sh | netcat bancroft 7357
- Produce TAP
- Just drop into port
- “fire & forget”
Report interface (2)

- Hide internal complexity
  - TAP::Parser
  - TAP::Formatter::HTML
  - TAP::DOM
  - Meta information
  - Sections
  - Aggregated results

- How to trivially access results?
Problem
First the “no-problem”

- WebApp for “end users”
  - Catalyst
  - DBIx::Class (::Schema::Versioned)
  - TAP::* (Parser, Formatter::HTML)
Web Application for “end users”
Web Application for “end users”

<table>
<thead>
<tr>
<th>ID</th>
<th>Date</th>
<th>Suite</th>
<th>Success</th>
<th>Result</th>
<th>Grouped by</th>
</tr>
</thead>
<tbody>
<tr>
<td>11116</td>
<td>2009-07-16</td>
<td>DailyReport</td>
<td>PASS</td>
<td>PASS</td>
<td></td>
</tr>
<tr>
<td>11117</td>
<td>2009-07-16</td>
<td>KERNEL</td>
<td>PASS</td>
<td>PASS</td>
<td></td>
</tr>
<tr>
<td>11118</td>
<td>2009-07-16</td>
<td>KERNEL</td>
<td>PASS</td>
<td>PASS</td>
<td></td>
</tr>
<tr>
<td>11119</td>
<td>2009-07-16</td>
<td>KERNEL</td>
<td>PASS</td>
<td>PASS</td>
<td></td>
</tr>
<tr>
<td>11120</td>
<td>2009-07-16</td>
<td>KERNEL</td>
<td>PASS</td>
<td>PASS</td>
<td></td>
</tr>
<tr>
<td>11121</td>
<td>2009-07-16</td>
<td>KERNEL</td>
<td>PASS</td>
<td>PASS</td>
<td></td>
</tr>
<tr>
<td>11122</td>
<td>2009-07-16</td>
<td>KERNEL</td>
<td>PASS</td>
<td>PASS</td>
<td></td>
</tr>
<tr>
<td>11123</td>
<td>2009-07-16</td>
<td>KERNEL</td>
<td>PASS</td>
<td>PASS</td>
<td></td>
</tr>
<tr>
<td>11124</td>
<td>2009-07-16</td>
<td>KERNEL</td>
<td>PASS</td>
<td>PASS</td>
<td></td>
</tr>
<tr>
<td>11125</td>
<td>2009-07-16</td>
<td>KERNEL</td>
<td>PASS</td>
<td>PASS</td>
<td></td>
</tr>
<tr>
<td>11126</td>
<td>2009-07-16</td>
<td>KERNEL</td>
<td>PASS</td>
<td>PASS</td>
<td></td>
</tr>
<tr>
<td>11127</td>
<td>2009-07-16</td>
<td>KERNEL</td>
<td>PASS</td>
<td>PASS</td>
<td></td>
</tr>
<tr>
<td>11128</td>
<td>2009-07-16</td>
<td>KERNEL</td>
<td>PASS</td>
<td>PASS</td>
<td></td>
</tr>
</tbody>
</table>

16 Cinderella ‘TAP: The lazy evaluation sisters of TAP::Parser | Aug 4, 2009
Web Application for “end users”
Web Application for “end users”

<table>
<thead>
<tr>
<th>ID</th>
<th>date</th>
<th>suite</th>
<th>machine</th>
<th>success</th>
<th>ratio</th>
<th>grouped by</th>
</tr>
</thead>
<tbody>
<tr>
<td>3687</td>
<td>2009-02-07</td>
<td>Artemis-CTCS</td>
<td>caesium</td>
<td>FAIL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3618</td>
<td>2009-02-07</td>
<td>Artemis-CTCS</td>
<td>rutherford</td>
<td>FAIL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3556</td>
<td>2009-01-27</td>
<td>Artemis-CTCS</td>
<td>caesium.com.amd.com</td>
<td>FAIL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3519</td>
<td>2009-01-27</td>
<td>Artemis-CTCS</td>
<td>caesium.com.amd.com</td>
<td>FAIL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3512</td>
<td>2009-01-27</td>
<td>Artemis-CTCS</td>
<td>artemis.com.amd.com</td>
<td>FAIL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3509</td>
<td>2009-01-27</td>
<td>Artemis-CTCS</td>
<td>armium.com.amd.com</td>
<td>FAIL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3584</td>
<td>2009-01-27</td>
<td>Artemis-CTCS</td>
<td>david</td>
<td>FAIL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3502</td>
<td>2009-01-27</td>
<td>Artemis-CTCS</td>
<td>alanium.com.amd.com</td>
<td>FAIL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3486</td>
<td>2009-01-27</td>
<td>Artemis-CTCS</td>
<td>utium.com.amd.com</td>
<td>FAIL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3486</td>
<td>2009-01-26</td>
<td>Artemis-CTCS</td>
<td>vanium.com.amd.com</td>
<td>FAIL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3486</td>
<td>2009-01-26</td>
<td>Artemis-CTCS</td>
<td>filamentum.com.amd.com</td>
<td>FAIL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3476</td>
<td>2009-01-26</td>
<td>Artemis-CTCS</td>
<td>vanadium.com.amd.com</td>
<td>FAIL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3476</td>
<td>2009-01-26</td>
<td>Artemis-CTCS</td>
<td>vanadium.com.amd.com</td>
<td>FAIL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3476</td>
<td>2009-01-26</td>
<td>Artemis-CTCS</td>
<td>filamentum.com.amd.com</td>
<td>FAIL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3466</td>
<td>2009-01-26</td>
<td>Artemis-CTCS</td>
<td>dubium.com.amd.com</td>
<td>FAIL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3466</td>
<td>2009-01-26</td>
<td>Artemis-CTCS</td>
<td>neptunium.com.amd.com</td>
<td>FAIL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3466</td>
<td>2009-01-26</td>
<td>Artemis-CTCS</td>
<td>plutonium.com.amd.com</td>
<td>FAIL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3466</td>
<td>2009-01-26</td>
<td>Artemis-CTCS</td>
<td>proteonium.com.amd.com</td>
<td>FAIL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3466</td>
<td>2009-01-26</td>
<td>Artemis-CTCS</td>
<td>plutonium.com.amd.com</td>
<td>FAIL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3466</td>
<td>2009-01-26</td>
<td>Artemis-CTCS</td>
<td>beryllium.com.amd.com</td>
<td>FAIL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3466</td>
<td>2009-01-26</td>
<td>Artemis-CTCS</td>
<td>beryllium.com.amd.com</td>
<td>FAIL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3466</td>
<td>2009-01-26</td>
<td>Artemis-CTCS</td>
<td>neptunium.com.amd.com</td>
<td>FAIL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3466</td>
<td>2009-01-26</td>
<td>Artemis-CTCS</td>
<td>neptunium.com.amd.com</td>
<td>FAIL</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Web Application for “end users”
Web Application for “end users”
Web Application for “end users”

KernBench 0.020016

Context

Test results

<table>
<thead>
<tr>
<th>Test file</th>
<th>Test results</th>
</tr>
</thead>
<tbody>
<tr>
<td>/atennis</td>
<td>100.0%</td>
</tr>
<tr>
<td>/meta-information</td>
<td>100.0%</td>
</tr>
<tr>
<td>/clocksource</td>
<td>100.0%</td>
</tr>
<tr>
<td>/cmd</td>
<td>100.0%</td>
</tr>
<tr>
<td>/kernel</td>
<td>100.0%</td>
</tr>
<tr>
<td>/kernbench</td>
<td>100.0%</td>
</tr>
<tr>
<td>/kernbench-test</td>
<td>100.0%</td>
</tr>
<tr>
<td>/kernbench-utime</td>
<td>100.0%</td>
</tr>
<tr>
<td>/kernbench-utime</td>
<td>100.0%</td>
</tr>
<tr>
<td>/kernbench-utime</td>
<td>100.0%</td>
</tr>
<tr>
<td>/stats</td>
<td>100.0%</td>
</tr>
<tr>
<td>/proc/interrupts</td>
<td>100.0%</td>
</tr>
<tr>
<td>/sys</td>
<td>100.0%</td>
</tr>
<tr>
<td>/var/log_messages</td>
<td>100.0%</td>
</tr>
</tbody>
</table>

15 tests; 29 ok, 0 failed, 0 skipped, 0 parse errors
exit status: 0, wall time: 0
elapsed time: 0 wallclock secs ( 0.14 user 0.04 sys  0.00 cwbk 0.04 sys  0.04 cwbk 0.26 CPU)

raw TAP report

Reports of same group (Testrun)

<table>
<thead>
<tr>
<th>ID</th>
<th>date</th>
<th>suite</th>
<th>machine</th>
<th>success</th>
<th>notes</th>
<th>grouped by</th>
</tr>
</thead>
<tbody>
<tr>
<td>10722</td>
<td>2009-07-28</td>
<td>Topics-Xen</td>
<td>lebted</td>
<td>PASS</td>
<td></td>
<td>lebted</td>
</tr>
<tr>
<td>10721</td>
<td>2009-07-28</td>
<td>LTP</td>
<td>lebted-amd1-arch-amd.com</td>
<td>FAIL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10720</td>
<td>2009-07-28</td>
<td>LTP</td>
<td>lebted-amd2-arch-amd.com</td>
<td>FAIL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10719</td>
<td>2009-07-28</td>
<td>TAP</td>
<td>lebted-kernels</td>
<td>FAIL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10718</td>
<td>2009-07-28</td>
<td>Host-Overview</td>
<td>lebted</td>
<td>PASS</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Web Application for “end users”
Web Application for “end users”
Web Application for “end users”
Web Application for “end users”

25 Cinderella 'TAP: The lazy evaluation sisters of TAP::Parser | Aug 4, 2009
Web Application for “end users”
The Query Gap

- Scriptable querying
- The same ease as reporting
- Again: shell level, netcat
The Query Gap (2)

- Use Cases
  - generally access our own reports
  - track single test over time
  - track benchmark results (YAML in TAP)
  - custom visualize the data

- Challenges
  - test suites change over time → fuzzy find
  - hide the toolchain
Solution
Query interface (1)

- Provide template mechanism
- With embedded query language “DPath”
- Dialog-oriented protocol
  - send with `netcat`
  - HERE-doc style
  - receive answer
Query interface (2) - Synopsis

- Command:

  $ cat report.mas | netcat bancroft 7358 > result.txt

- Template:

  ```
  #! mason <<EOF
  Planned oprofile tests:

  % foreach $plan (reportdata ' { suite_name => "oprofile" } :: //tap/tests_planned') {
      <%- $plan %> 
  % }

  EOF
  ```

- Result

  Planned oprofile tests:
  
  3
  4
  17
#! mason_debug=1 <<EOTEMPLATE

TITLE = "success ratio: CTCS"
set title TITLE offset char 0, char -1
set style data linespoints
set term png size 1200, 800
set output "CTCS_ratio.png"
set yrange [80:110]
plot '-' using 0:2 with linespoints lt 3 lw 1 title "ratio"

% my @time = reportdata '{ suite_name => "CTCS" } :: /report/created_at_ymd';
% my @ratio = reportdata '{ suite_name => "CTCS" } :: //success_ratio';
% foreach my $i (0..@ratio) {
  <% $time[$i] %> <% $ratio[$i] %>
% }

EOTEMPLATE
Query interface (4) - Synopsis

$ cat CTCS_ratio.gnuplot | netcat bancroft 7358 | gnuplot

(generated CTCS_ratio.png)
How does it work

- Modules
  - TAP::DOM - TAP as data structure
  - Data::DPath - XPath like language

  - MyApp::DOM - Project add-ons (report meta)
  - MyApp::DPath - Project add-ons (db layer)
Anatomy of a MyApp::DPath

{ suite_name => "CTCS" } :: //tests_planned[value > 10]/../summary/passed

- Virtual DOM, 2 orthogonal concepts

- **database axis**: provide but hide relational access
  - DBI::Class
  - SQL::Abstract
  - project-specific add-ons
  - the “history of reports”

- **report axis**: inside single reports data structure
  - TAP::DOM
  - Data::DPath
  - “at one point in history”
Data::DPath

- XPath like language
- Optimized for Perl
  - data structures
  - filter sub language
  - more “Why not XPath?” in Data::DPath docs

→ [http://xrl.us/dpathvxpath](http://xrl.us/dpathvxpath)
use Data::DPath 'dpath';

my $data = { AAA => { BBB => { CCC => [ qw/ XXX YYY ZZZ / ] } },
             RRR => { CCC => [ qw/ RR1 RR2 RR3 / ] } },
             DDD => { EEE => [ qw/ uuu vvv www / ] } }

$resultlist = $data -- dpath '/AAA/*/CCC';

→ [ [ 'XXX', 'YYY', 'ZZZ'],
     [ 'RR1', 'RR2', 'RR3' ] ]
use Data::DPath 'dpath';

my $data = {
    AAA => {
        BBB => {
            CCC => [ qw/ XXX YYY ZZZ / ],
            RRR => {
                CCC => [ qw/ RR1 RR2 RR3 / ],
                DDD => {
                    EEE => [ qw/ uuu vvv www / ]
                }
            }
        }
    }
};

$resultlist = $data -- dpath '/AAA/*/CCC';

    → [ [ 'XXX', 'YYY', 'ZZZ' ],
         [ 'RR1', 'RR2', 'RR3' ] ]
Data::DPath - Synopsis (3)

$data -- dpath '/AAA/*/CCC'
$data -- dpath '/AAA/BBB/CCC/../../../'
$data -- dpath '//AAA/
$data -- dpath '//AAA/*'
$data -- dpath '//AAA//SOMEWHERE//BELOW'
$data -- dpath '/EE/E/CCC'
$data -- dpath '/AAA/BBB/CCC/*[1]'
$data -- dpath '//AAA/BBB/*[key eq "CCC"]'
$data -- dpath '//CCC/*[value eq "RR2"]'
Data::DPath - Synopsis (3)

```perl
$data -- dpath '/AAA/*/CCC'
$data -- dpath '/AAA/BBB/CCC/..../..
$data -- dpath '//AAA'
$data -- dpath '//AAA/*'
$data -- dpath '//AAA//SOMEBEWHERE//BELOW'
$data -- dpath '//EE/E'/CCC'
$data -- dpath '/AAA/BBB/CCC/*[1]'
$data -- dpath '//AAA/BBB/*[key eq "CCC"]'
$data -- dpath '//CCC/*[value eq "RR2"]'
```

- What is our $data?
TAP::DOM

- TAP as data structure
- TAP → TAP::Parser → TAP::DOM → DOM
use TAP::DOM;

# same options as TAP::Parser
my $tapdata = new TAP::DOM ( tap => $tap );
print Dumper( $tapdata );
bless ( {
    'tests_planned' => 6,
    'tests_run' => 8,
    # [...]
    'summary' => {
        'status' => 'FAIL',
        'total' => 8,
        'passed' => 6,
        'failed' => 2,
        'skipped' => 1,
        'todo' => 4,
        'todo_passed' => 2,
        # [...]
    },
    'lines' => [ {
        'number' => '1',
        'is_ok' => 1,
        'description' => '- use Data::DPath;',
        '_children' => [ {
            'is_yaml' => 1,
            'data' => [ {
                'name' => 'Hash one',
                'value' => '1'
            }, {
                'name' => 'Hash two',
                'value' => '2'
            } ]
        } ]
    }, 'TAP::DOM'
}
Complex TAP::DOM is easy with Data::DPath

- Look at complete DOM once, use dpath '//'
- Find interesting keys
- Use dpath '//key':

```perl
dpath '//'summary/passed'
dpath '//'description//foo'
```
Daemon + Template + DPath

- One function for everything, hide iterating $data 
  $\rightarrow$ reportdata($\text{path}$)

- Prelude in template engine

- Daemon answers after HERE-documents

```shell
#!/mason
<<EOTEMPLATE
Planned oprofile tests:
% foreach $plan (reportdata '{ search } :: //path') {
  <% $plan %>%
}%
EOTEMPLATE

$ cat report.mas | netcat bancroft 7358 > result.txt
```
New-school dependencies

```perl
use MooseX::Declare;
use 5.010;

class Data::DPath is dirty {
    clean;
    method match (Any $data, Str $path) {
        given ($step->kind) {
            when ('ANYWHERE') {
                # ...
            }
        }
    }
}
```
Challenges

- Data::DPath → speed!
- TAP::DOM → size!
Cache!

- 3 levels
  - TAP::DOM
    - in DB
  - Data::DPath queries
    - path + report_id
    - Cache::FileCache
  - MyApp::DPath queries
    - path + count of matching reports
    - Cache::FileCache
Superlarge TAP::DOMs

- Single monster TAPs still in RAM
  - 70MB TAP → xGB TAP::DOM
  - how to compress data structure?
  - open issue
Project spin-offs

- App::DPath
- Emacs tap-mode
App::DPath

- App::DPath
  - cmdline tool \texttt{dpath} around Data::DPath
  - input filters: YAML, JSON, Data::Dumper, INI, TAP
  - output filters: YAML, JSON, Data::Dumper
  - useful for developing/debugging DPaths
App::DPath - Example

- Find passing TODO tests

perl foo.t | dpath -i tap '://has_todo[value==1]/../is_actual_ok[value==1]/..'
Emacs tap-mode

---

TAP version 13
1.6
ok 1 - use Data::DPath;
   ...
   - name: 'Hash one'
     value: 1
   - name: 'Hash two'
     value: 2
   ...
ok 2 - KEYS + PARENT
ok 3 - quoted KEY contain g slash
pragma +strict
not ok 4 # TODO spec only
  # Failed (TODO) test at t/data_dpath.t line 144.
  # Structures begin differing at:
  # $got->[0] = Does not exist
  # $expected->[0] = ARRAY(0x8e4c238)
ok 5 - ANYWHERE + KEY + FILTER int # TODO spec only
ok 6 # skip rethink twice
not ok 7 # TODO spec only
  # Failed (TODO) test at t/data_dpath.t line 356.
  # Structures begin differing at:
  # $got->[0] = Does not exist
  # $expected->[0] = 'interesting value'
ok 8 - FILTER eval regex # TODO too dirty, first cleanup _filter_eval

---

53 Cinderella 'TAP: The lazy evaluation sisters of TAP::Parser | Aug 4, 2009
Summary (1)

- Test automation & participation infrastructure

- Test scripts emitting TAP
  - echo '1..2'
  - echo 'ok'
  - echo 'not ok'

- Trivially report
  - $ ./testscript | netcat
Summary (2)

- Review results via WebApp

- Query interface
  - Data::DPath
  - TAP::DOM
  - Templates

```
% my @ratio = reportdata '{ search } :: //dpath';
% foreach my $i (0..@ratio) {
    <% $time[$i] %> <% $ratio[$i] %>
% }

$ cat template | netcat > result
```
Trademark Attribution

AMD, the AMD Arrow logo and combinations thereof are trademarks of Advanced Micro Devices, Inc. in the United States and/or other jurisdictions. Other names used in this presentation are for identification purposes only and may be trademarks of their respective owners.

©2009 Advanced Micro Devices, Inc. All rights reserved.