cadence

ams Sensing is life.

Processing deliberate verification errors during regression

Alastair Lefley, ams AG, Kemble, UK (Alastair.Lefley@ams.com)
Roger Witlox, ams AG, Eindhoven, NL (Roger.Witlox@ams.com)
Clemens Süßmuth, ams AG, Premstätten, AT (Clemens.Suessmuth@ams.com)
Thomas Ziller, Cadence, München, DE (ThomasZ@cadence.com)
Kawe Fotouhi, Cadence, München, DE (Fotouhi@cadence.com)

accellera SYSTEMS INITIATIVE

2019 DESIGN AND VERIFICATION DVCON CONFERENCE AND EXHIBITION EUROPE

Quis custodiet ipsos custodes / Who will check the checkers?

- VIPs have teams of bodyguards – but do they remain loyal?
- DV teams write many checkers – but do they remain functional?

The drive for automated regression

- As design complexity increases, so too do DV needs increase;
- The DV team will create a suite of testbenches and testcases.
  - Regression testing returns to, and reruns, the suite of testbenches and testcases;
  - Regression involves automatically producing coverage and pass/fail statistics;
  - We return in order to re-verify, as there will have been changes;
- While regressing, are all of our checks still functional, or are they compromised by the various changes?

While regressing, are all checks still functional, or are they compromised by the various changes?

- Why might a previous check cease to become active?
  - A qualifying condition is no longer met;
  - It gets disabled;
  - The sample clock is changed: deactivated, or masking duty-cycle;
  - ‘define macros get redefined;
- Important that the DV regression proves all checks are still active;

Methodology to check checkers during regression

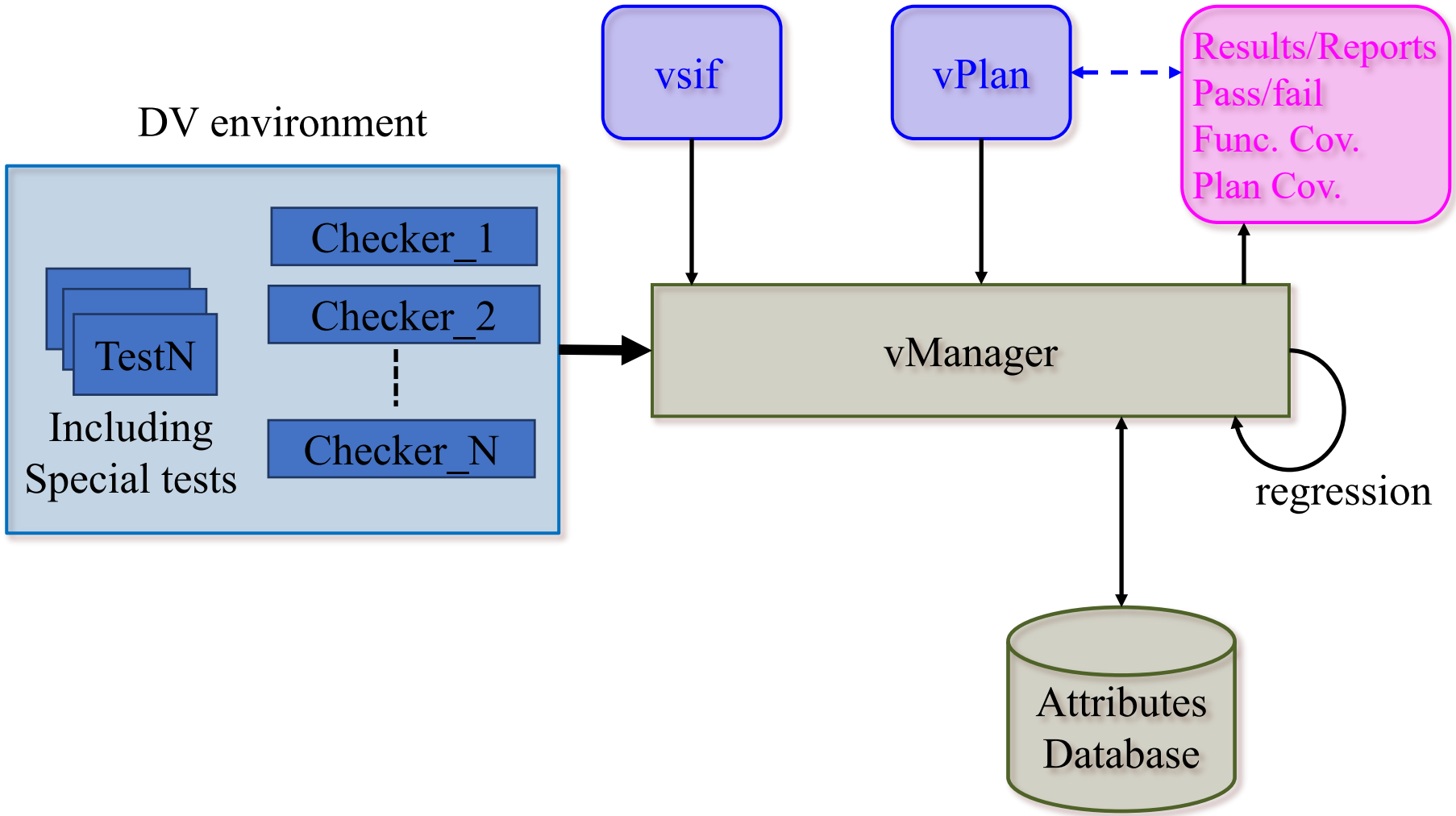
- Need agnostic method to check all checkers remain functional;
- Problem:
  - **Need to be deliberately producing errors**;
  - Standard processing of regression results reports tests as:
    - FAIL – if they contain one or more errors;
    - PASS – if they have no errors;
- New methodology:
  - On selected tests, introduce predefined number of deliberate errors – provoke checkers;
  - Revised processing of regression results reports tests as:
    - FAIL – if they do not match the predefined deliberate errors **or** have others errors;
    - PASS – if they match the predefined number / style of deliberate errors **and** have no other errors;

Processing of regression results

- The regression setup will study two main aspects of the simulation data:
  1. The functional coverage data;
  2. The simulation log files containing error, warning and other information;

- The methodology proposed involves:
  - Using and adapting the Cadence® vManager™ flow;
  - Adding extra intelligence to the parsing of the simulation log files;

Simplified vManager use-case flow



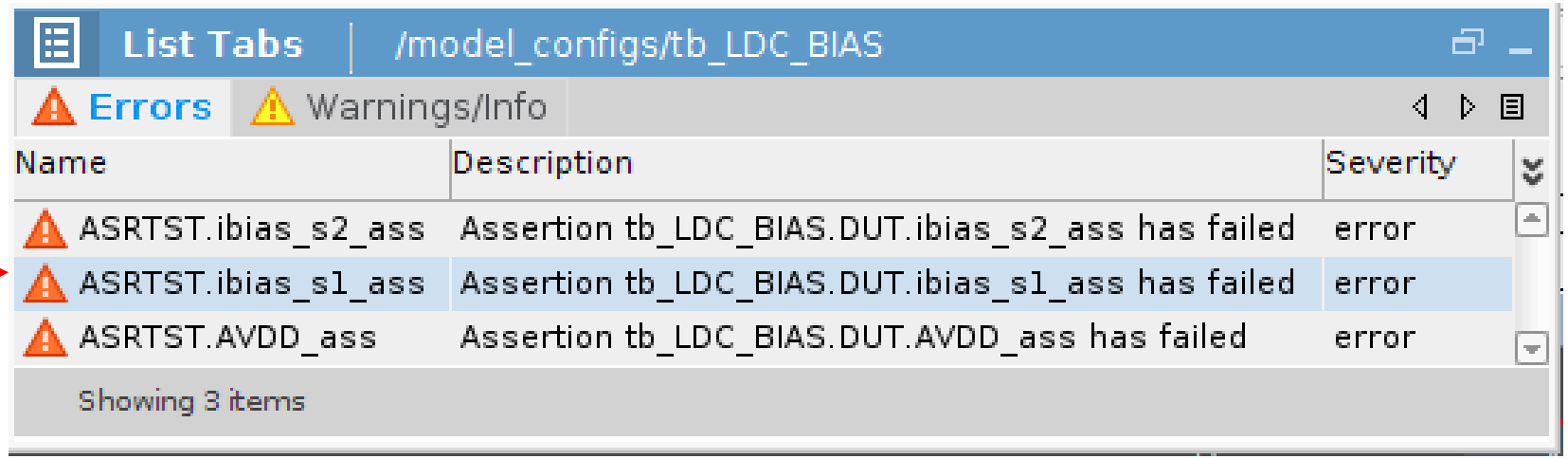
Assert : source code → log file

```
always @*
  begin
    if (en == 1) begin
      #1;
      ibias_s1_ass : assert (correct_ibias_s1)
        else $error (" ibias_0u5 out of range %f at time %tus",ibias_0u5, $time);
    end //En==1
  end //always
```

```
Time:        30.138 us        Info: en = 1; ibias = 0. This deliberately
provokes and proves a model-based assertion.
ncsim: *E,ASRTST (/PATHNAME/PROJECTX/LDC_BIAS_V1/systemVerilog/verilog.sv,152)
: (time 30139 NS) Assertion tb_LDC_BIAS.DUT.ibias_s1_ass has failed
 ibias_0u5 out of range 0.000000 at time        30.139 usus
Time:        36.147 us        Info: en = 1; ibias = 0.5e-6.
```

Assert : log file → attributes in vManager DB

```
Time:        30.138 us        Info: en = 1; ibias = 0. This deliberately
provokes and proves a model-based assertion.
ncsim: *E,ASRTST (/PATHNAME/PROJECTX/LDC_BIAS_V1/systemVerilog/verilog.sv,152)
: (time 30139 NS) Assertion tb_LDC_BIAS.DUT.ibias_s1_ass has failed
 ibias_0u5 out of range 0.000000 at time        30.139 usus
Time:        36.147 us        Info: en = 1; ibias = 0.5e-6.
```



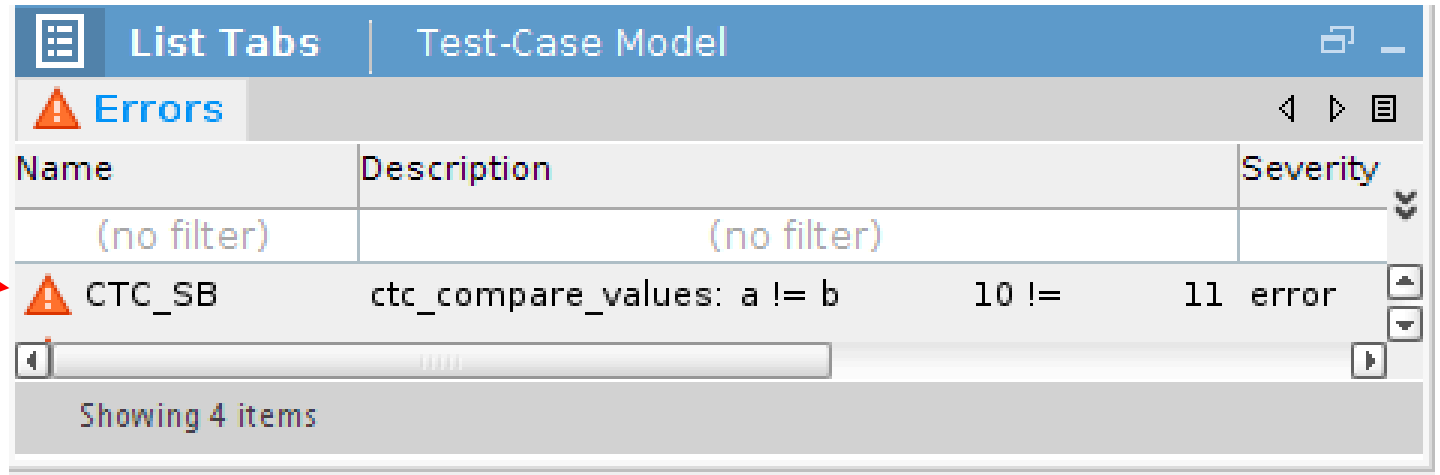
uvm_error : source code → log file

```
class ctc_sb extends uvm_scoreboard;
...
task ctc_compare_values();
    if (act_val != exp_val)
      `uvm_error("CTC_SB",$sformatf("ctc_compare_values: act_val != exp_val\n"))
endtask
...
```

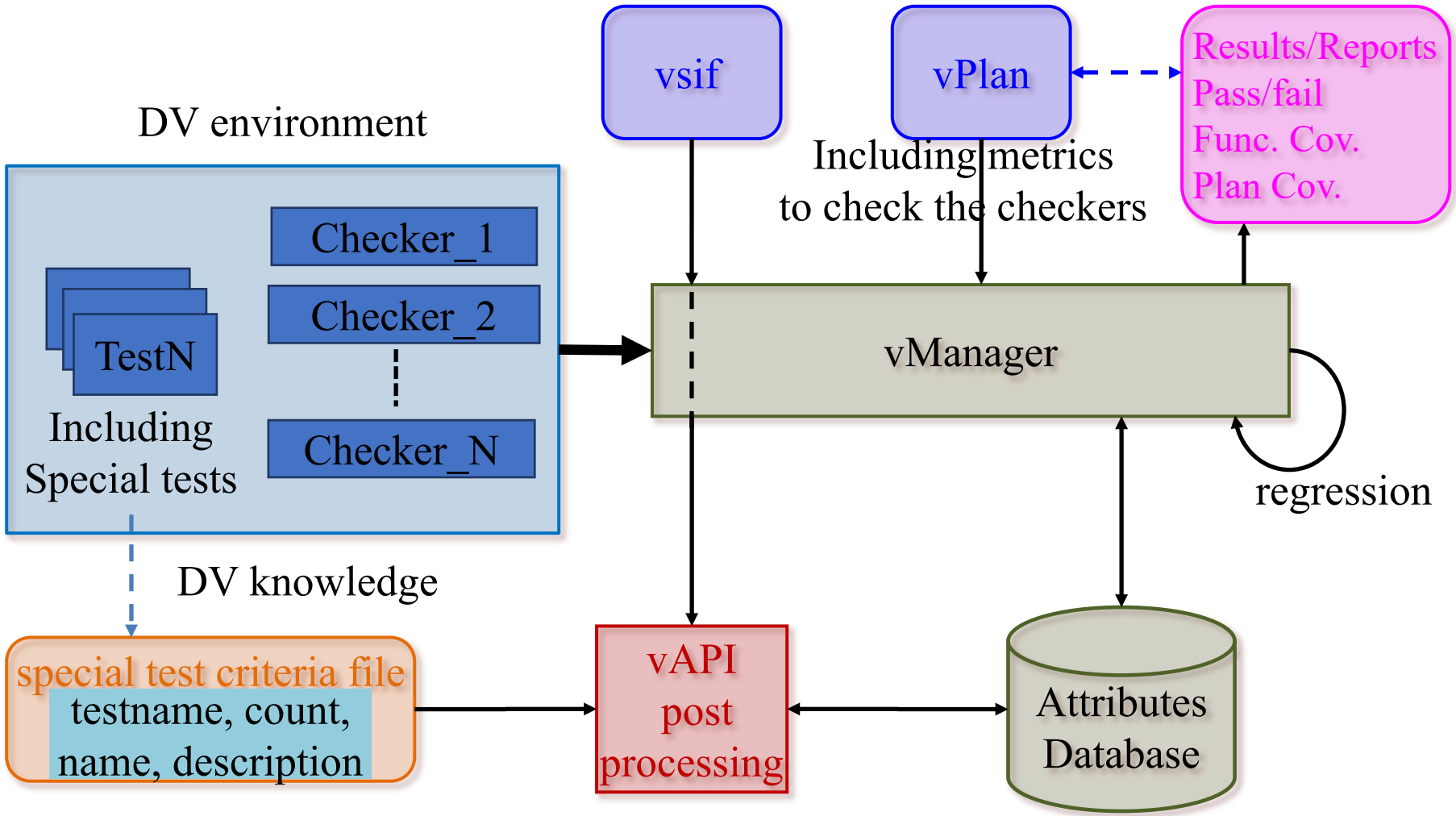
```
reporter [RNTST] Running test ctc_single_error_test...
...
UVM_ERROR /PATHNAME/ctc_uvm.sv(28) @ 100: uvm_test_top.m_ctc_env.m_ctc_sb
[CTC_SB] ctc_compare_values: act_val != exp_val
...
```

uvm_error : log file → attributes in vManager DB

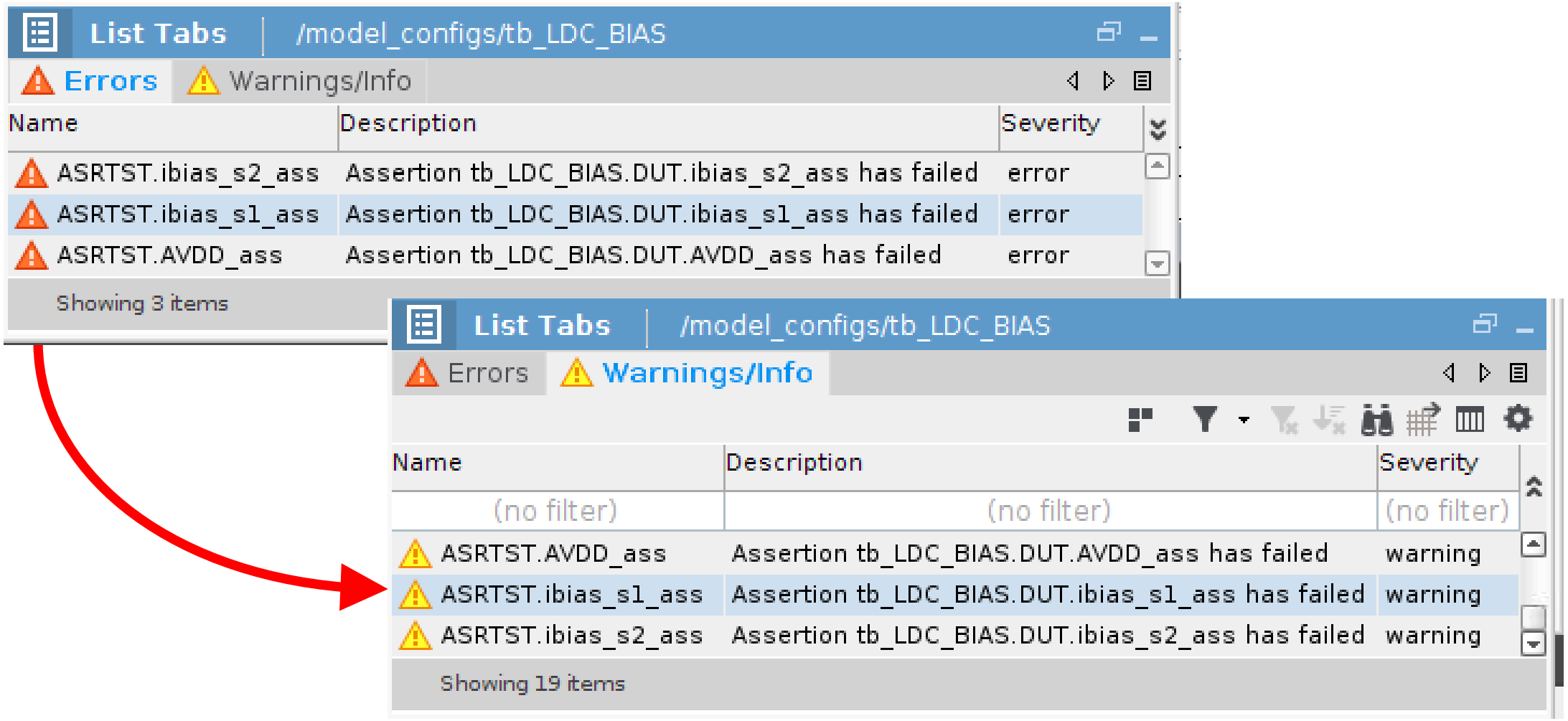
```
reporter [RNTST] Running test ctc_single_error_test...
...
UVM_ERROR /PATHNAME/ctc_uvm.sv(28) @ 100: uvm_test_top.m_ctc_env.m_ctc_sb
[CTC_SB] ctc_compare_values: act_val != exp_val
...
```



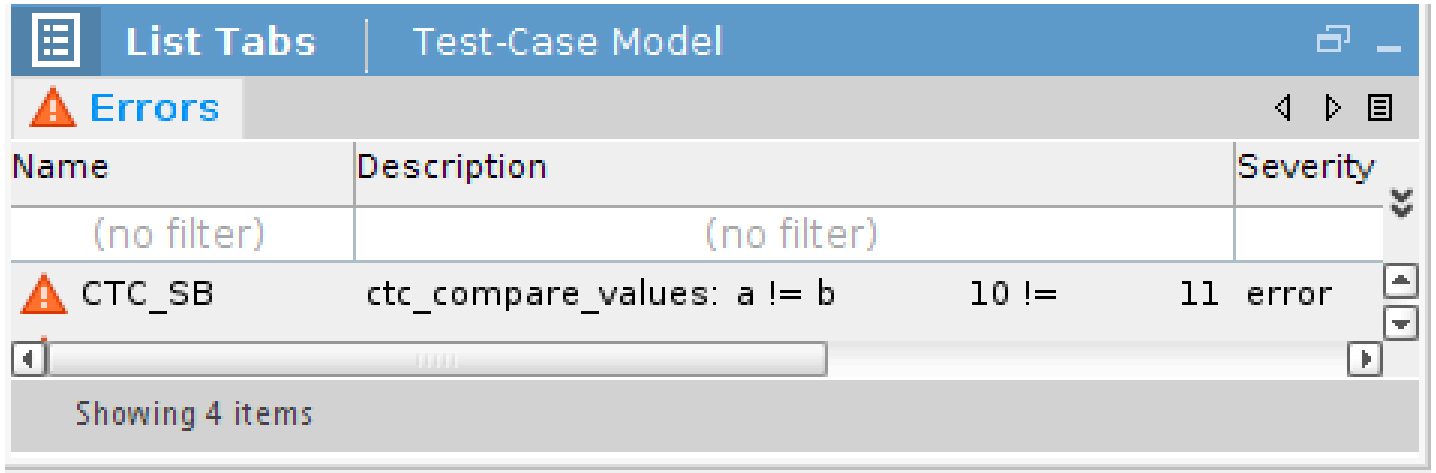
Adapted vAPI vManager use-case flow

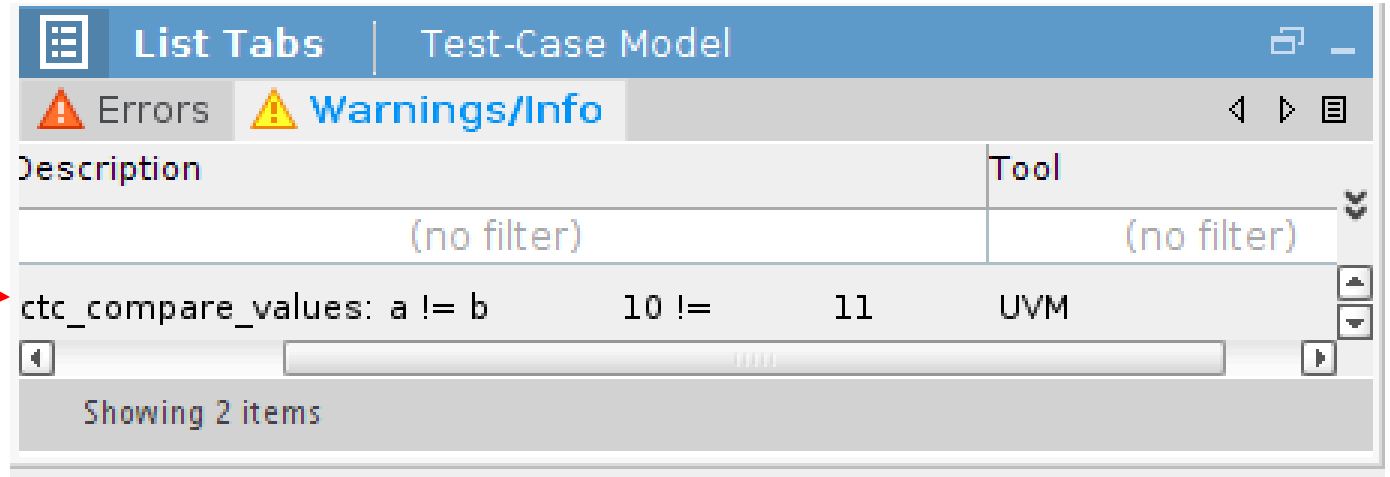


Down-grading of attributes in vManager DB

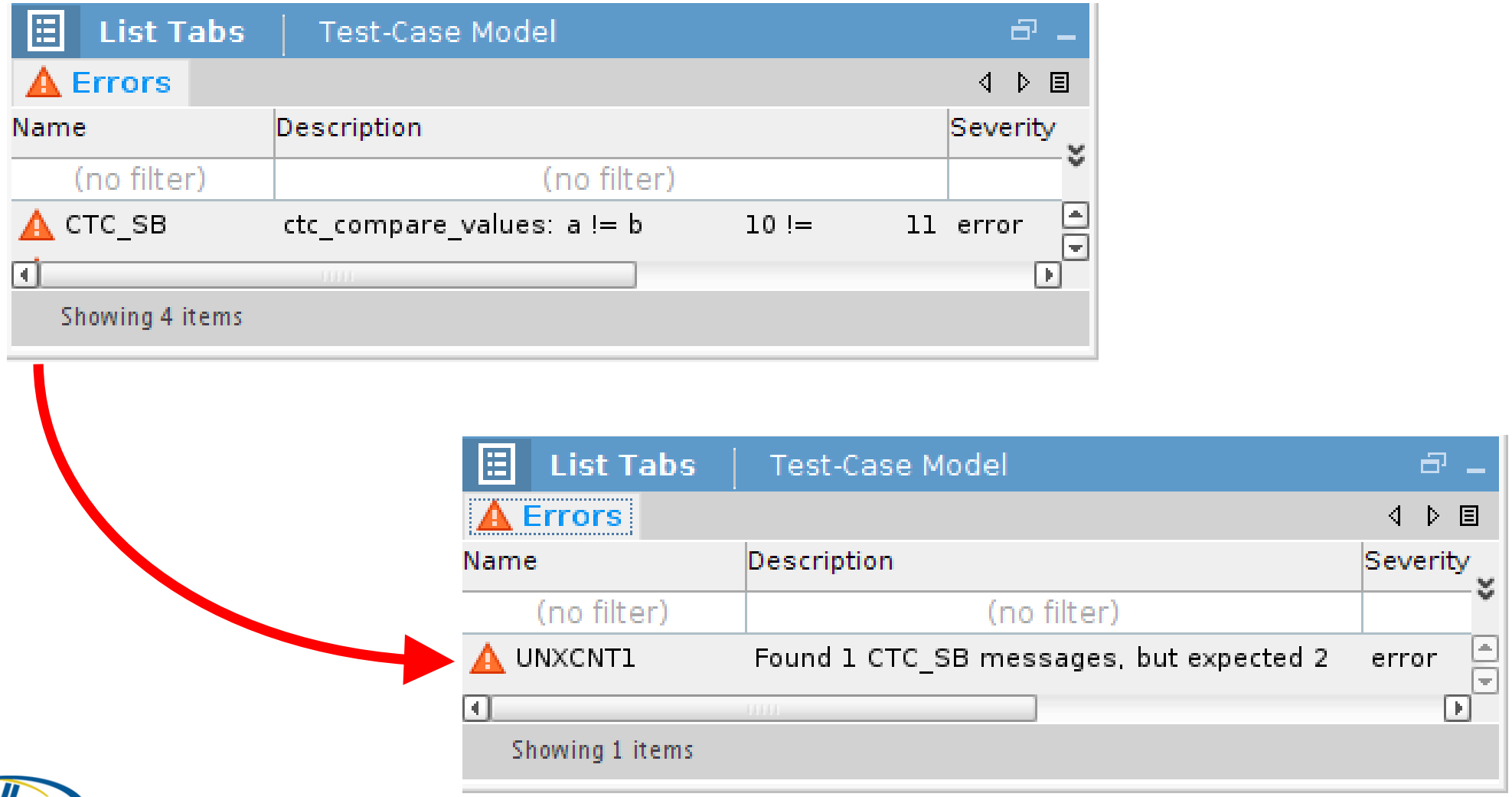


Down-grading of attributes in vManager DB





New attributes in vManager DB – count mismatch



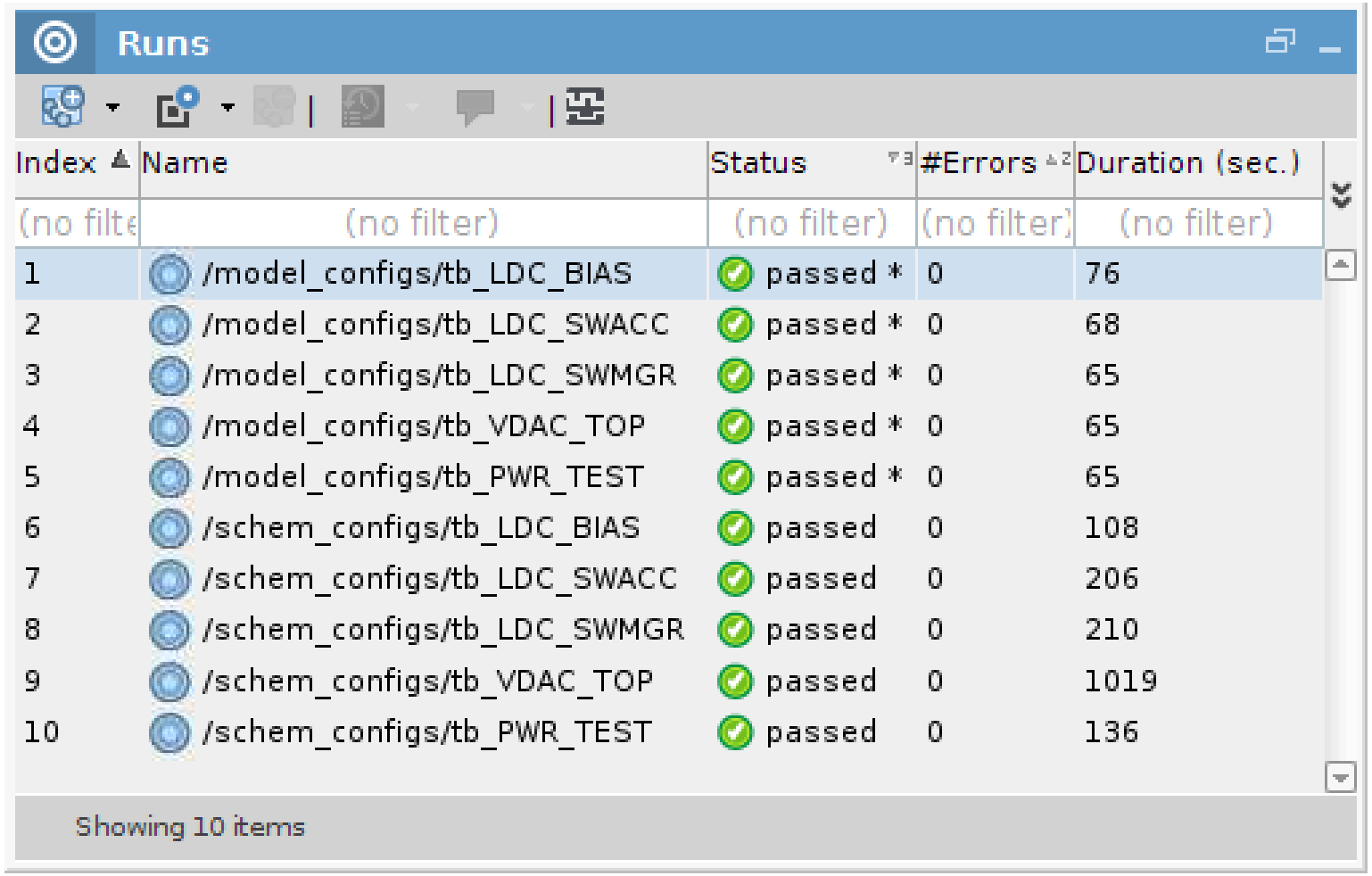
vManager DB – regression analysis, prior to vAPI

Runs

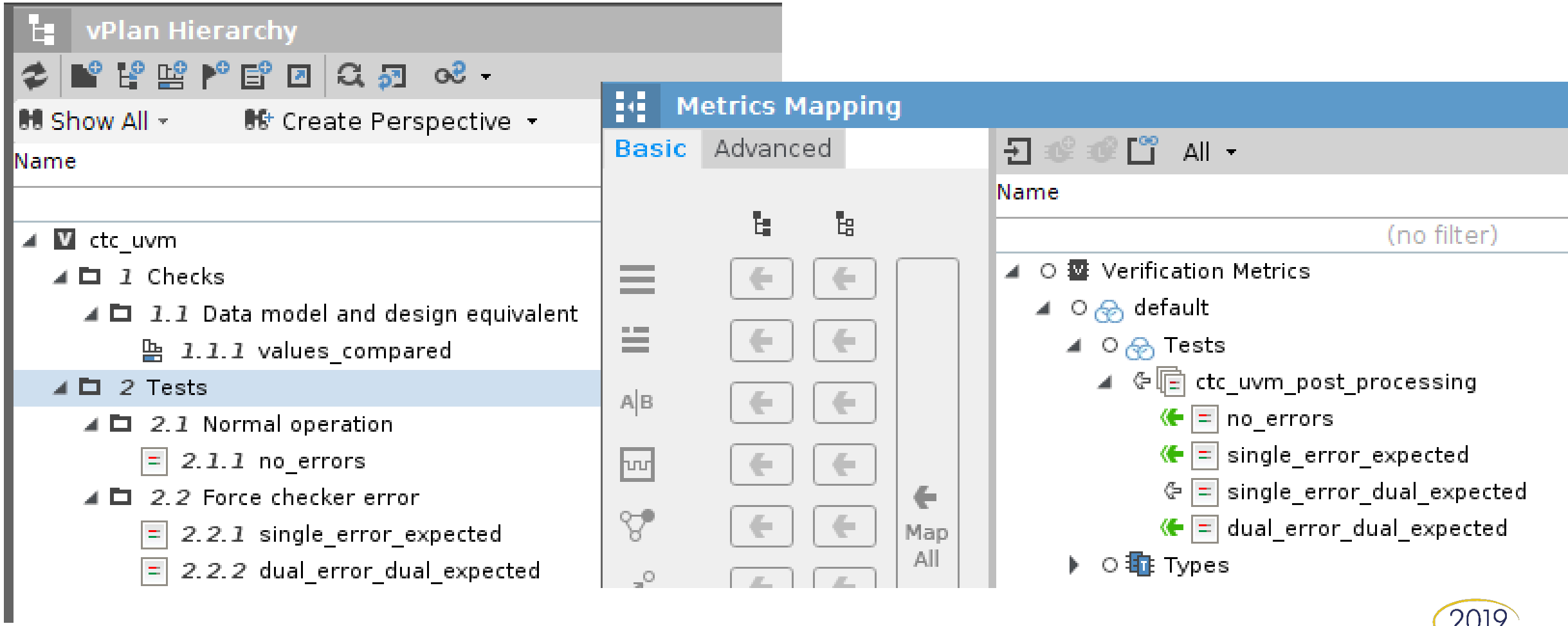
Index	Name	Status	#Errors	Duration (sec.)
(no filte	(no filter)	(no filter)	(no filter	(no filter)
1	/model_configs/tb_LDC_BIAS	failed	3	72
2	/model_configs/tb_LDC_SWACC	failed	2	65
3	/model_configs/tb_LDC_SWMGR	failed	1	63
4	/model_configs/tb_VDAC_TOP	failed	4	65
5	/model_configs/tb_PWR_TEST	failed	1	65
6	/schem_configs/tb_LDC_BIAS	passed	0	126
7	/schem_configs/tb_LDC_SWACC	passed	0	251
8	/schem_configs/tb_LDC_SWMGR	passed	0	260
9	/schem_configs/tb_VDAC_TOP	passed	0	1336
10	/schem_configs/tb_PWR_TEST	passed	0	171

Showing 10 items

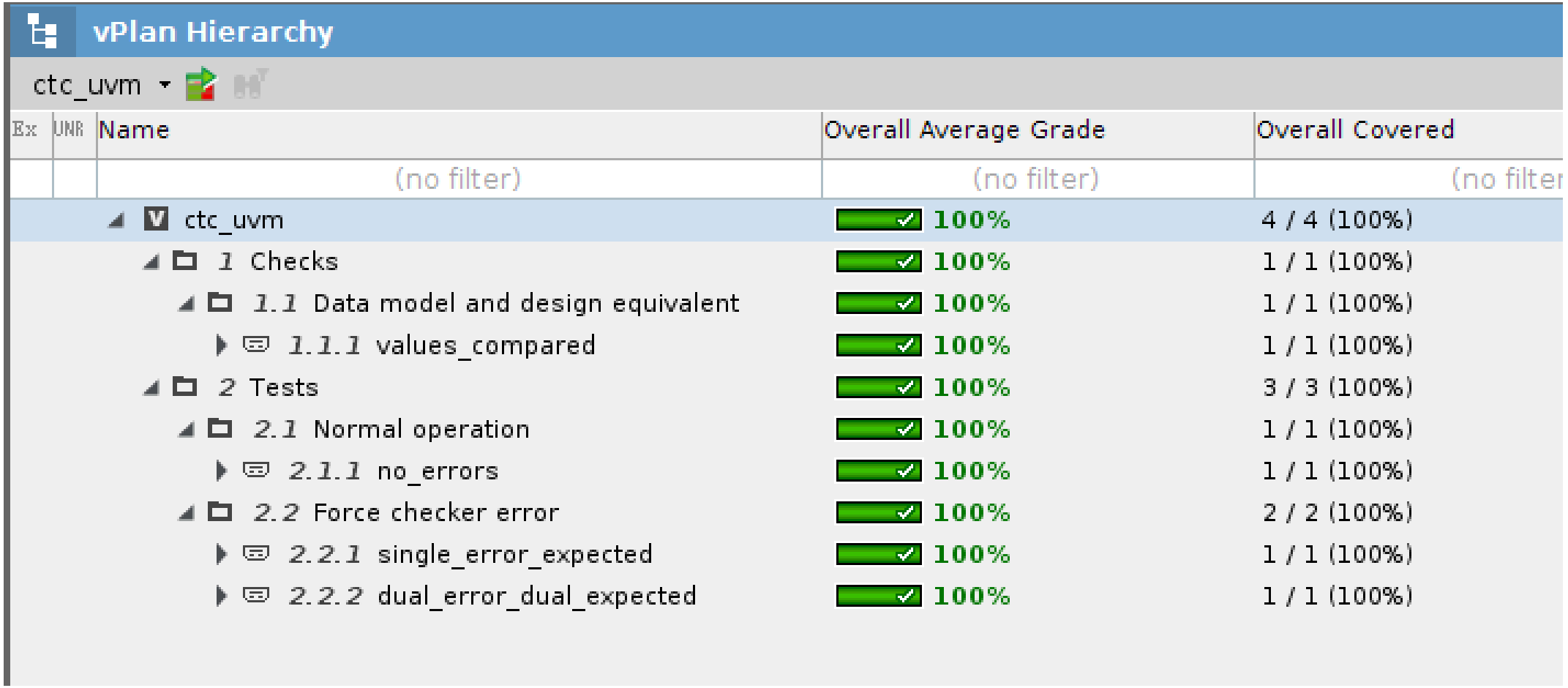
vManager DB – regression analysis, after vAPI



vManager – vPlan metrics for special tests



vManager – vPlan coverage for special tests



vAPI post-processing

- Controlled invocation from the vsif flow – “post session”;

- A python script is used – but other formats, Perl, are alternatives;
  1. Read and parse the special test criteria file;
  2. Using REST / JSON: traverse the attributes database – loop through all runs and all errors;
  3. Down-grade error attributes which match special test criteria; keep tally;
  4. If tally mismatches, introduce new error;
  5. Write back down-grades and new errors to the attributes database;

Special test criteria file – *.csv format – Ex. #1

test_name	count	name	description
tb_LDC_BIAS	1	ASRTST.AVDD_ass	Assertion tb_LDC_BIAS.DUT.AVDD_ass has failed
tb_LDC_BIAS	1	ASRTST.ibias_s1_ass	Assertion tb_LDC_BIAS.DUT.ibias_s1_ass has failed
tb_LDC_BIAS	1	ASRTST.ibias_s2_ass	Assertion tb_LDC_BIAS.DUT.ibias_s2_ass has failed
tb_LDC_SWACC	2	ASRTST.vdd_ass	Assertion tb_LDC_SWACC.DUT.vdd_ass has failed
tb_LDC_SWMGR	1	ASRTST.vdd_ass	Assertion tb_LDC_SWMGR.DUT.vdd_ass has failed
tb_VDAC_TOP	1	ASRTST.ibdacen_ass	Assertion tb_VDAC_TOP.DUT.ICORE.ibdacen_ass has failed
tb_VDAC_TOP	1	ASRTST.dacref_ass	Assertion tb_VDAC_TOP.DUT.ICORE.dacref_ass has failed
tb_VDAC_TOP	1	ASRTST.vref_ass	Assertion tb_VDAC_TOP.DUT.ICORE.vref_ass has failed
tb_VDAC_TOP	1	ASRTST.vdd_ass	Assertion tb_VDAC_TOP.DUT.ICORE.vdd_ass has failed
tb_PWR_TEST	1	ASRTST.vdd_ass	Assertion tb_PWR_TEST.DUT.vdd_ass has failed

Special test criteria file – *.csv format – Ex. #2

test_name	count	name	description
single_error_expected	1	CTC_SB	ctc_compare_values
single_error_dual_expected	2	CTC_SB	ctc_compare_values
dual_error_dual_expected	2	CTC_SB	ctc_compare_values

Conclusions

- We started explaining the need to check the checkers;
- We showed that introducing deliberate errors will explicitly demonstrate that the checkers are active;
- We showed that a cleverer PASS/FAIL definition can understand real- and deliberate-errors;
- This flow is now being actively used on a live project at ams;

Questions

Miscellaneous supplementary slides

- vsif flow – script excerpt;
- vAPI python – several script excerpts;

Miscellaneous – vsif script excerpt

```
session MINI__Projectx_model_vs_schem {
  top_dir : $ENV(SIM_DIR)/../vmgr_sessions;
  ...
  post_session_script: "check_the_checker.py";

  group model_configs {
    run_script:  "runams -cell $ATTR(test_name) -view model_config -simulate batch ..."
    special_test_criteria_file : "$ENV(DV_DIR)/bin/vsifs/MINI__model_vs_schem__special-test-criteria-file.csv";
    ...

    test tb_LDC_BIAS;
    ...
  };

  group schem_configs {
    run_script:  "runams -cell $ATTR(test_name) -view schem_config -simulate batch ...";
    ...

    test tb_LDC_BIAS;
    ...
  };

};
```

Miscellaneous – vAPI.py – REST definitions

```
set_server(vmgr_project="vmgr",server='https://'+vmgr_server,user=vmgr_username,passwd=vmgr_password)
headers = {
        'content-type':'application/json',
        'X-VMGR-Routing-Finalize' : 'true'
}
request={
         "filter" : { "@c"       : ".AttValueFilter",
                      "attName"  : "parent_session_name",
                      "operand"  : "EQUALS",
                      "attValue" : session_name
                    },
          "projection" : {
                       "selection" : ["id", "test_name", "special_test_criteria_file", "index",
"failures_count", "failed_runs_count", "errors_count", "sv_seed", "log_file"]
                    }
        }
runs_response = post(url='/runs/list',request=request,headers=headers)
runs_response_list = runs_response.json()
```

Miscellaneous – vAPI.py – loop through runs

```
# Loop over all the runs of the current session
for run_resp in runs_response_list :
  if (run_resp.has_key('special_test_criteria_file')) :
    print "   =       P O S T   P R O C E S S I N G     ="
    print "   Run: special_test_criteria_file = " + run_resp['special_test_criteria_file']

    exists = os.path.isfile(run_resp['special_test_criteria_file'])
    if exists :
      with open(run_resp['special_test_criteria_file'], 'rb') as csvfile:
        spamreader = csv.reader(csvfile, delimiter=',', quotechar='/')
        for row in spamreader:
          if (HDR_found & HDR_real_data) :
            expected_test_name       = row[0]
            expected_count           = int(row[1])
            expected_name            = row[2]
            expected_description     = row[3]
  else :
    print "   = N O   P O S T   P R O C E S I N G     ="
```

Miscellaneous – vAPI.py – loop thru error attributes

```
# Read severe message for this run
            severe_msg_response = post(url='/severe-messages/list',request=request,headers=headers)
            severe_msg_response_list = severe_msg_response.json()

            # Loop over all severe messages of this run.
            for severe_msg_resp in severe_msg_response_list :
              ...
```

Miscellaneous – vAPI.py – matching algorithm

```
if (re.match(expected_name, severe_msg_resp['name'])) :
  if (re.match('(.*)'+expected_description+'(.*)' , severe_msg_resp['description'])) :
    if (re.match('(.*)'+expected_test_name+'(.*)' , run_resp['test_name'])) :
      sim_log_count += 1
      if (sim_log_count == 1) :
        print "\n   match for " + expected_name + " - downgraded error to warning"
      ...                 ...
```

Miscellaneous – vAPI.py – write-back attributes DV

```
request={
                        "update": {"severity": "warning",
                                    "comment": "downgraded error to warning"
                                   },
                        "rs":      {"filter":{"@c": ".AttValueFilter",
                                              "attName": "id",
                                              "operand": "EQUALS",
                                              "attValue": severe_msg_resp['id']
                                             }
                                   }
                    }
severe_msg_update = post(url='/severe-messages/update',request=request,headers=headers)
```

Guidelines (1)

- Please keep the default font size for main lines at 28pt (or 26pt)
  - And use 24pt (or 22pt) font size for the sub bullets
- Use the default bullet style and color scheme supplied by this template
- Limited the number of bullets per page.
- Use keywords, not full sentences
- Please do not overlay Accellera or DVCon logo's
- Check the page numbering

Guidelines (2)

- Your company name and/or logo are only allowed to appear on the title page.
- Minimize the use of product trademarks
- Page setup should follow on-screen-show (4:3)
- Do not use recurring text in headers and/or footers
- Do not use any sound effects
- Disable dynamic slide transitions
- Limit use of animations (not available in PDF export)

Guidelines (3)

- Use clip-art only if it helps to state the point more effectively (no generic clip-art)
- Use contrasting brightness levels, e.g., light-on-dark or dark-on-light. Keep the background color white
- Avoid red text or red lines
- Use the MS equation editor or MathType to embed formulas
- Embed pictures in vector format (e.g. Enhanced or Window Metafile format)