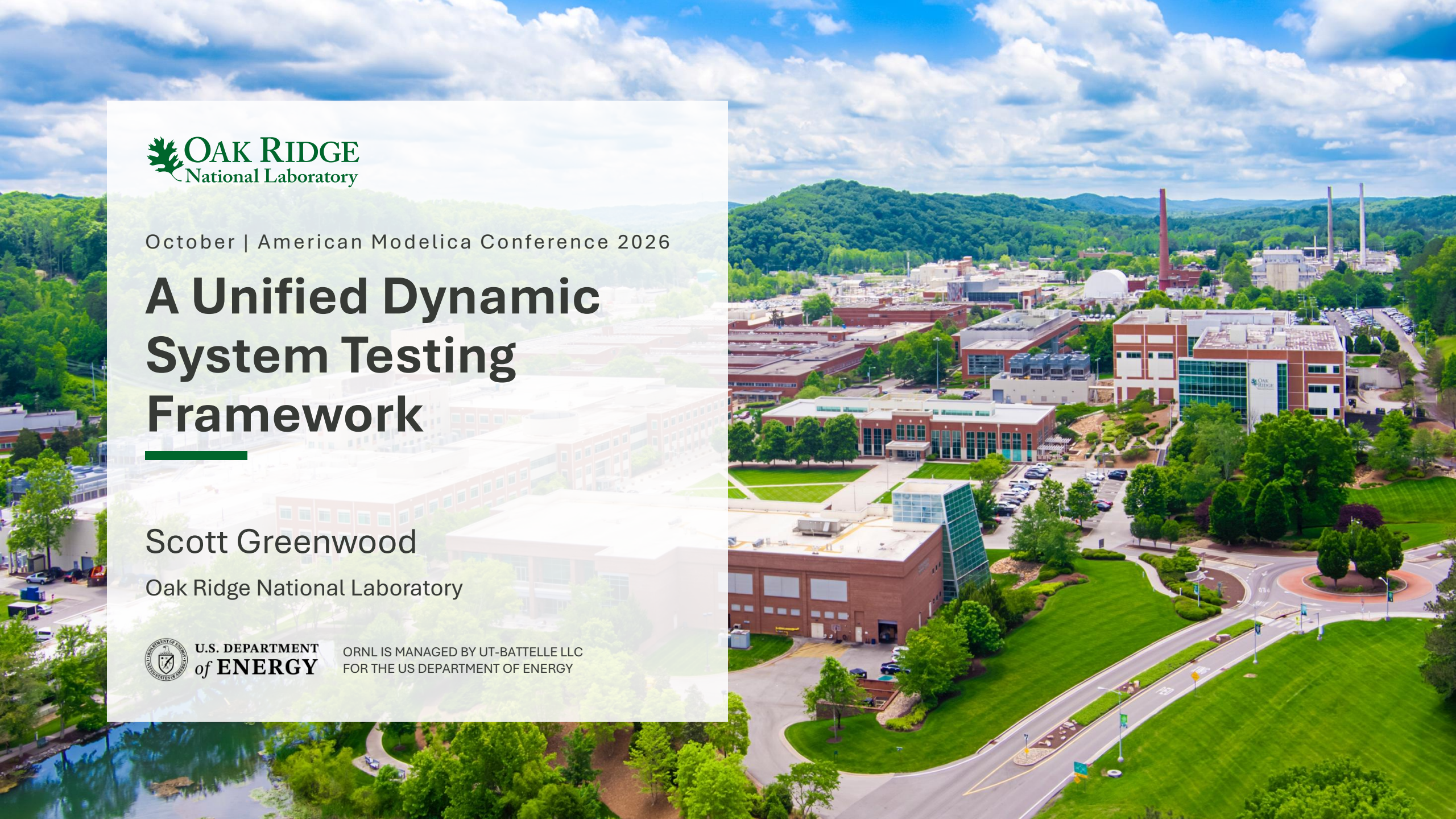


OAK RIDGE National Laboratory

October | American Modelica Conference 2026

A Unified Dynamic System Testing Framework

Scott Greenwood

Oak Ridge National Laboratory

U.S. DEPARTMENT of ENERGY

ORNL IS MANAGED BY UT-BATTELLE LLC
FOR THE US DEPARTMENT OF ENERGY

What is DSTF?

A test runner for time-dependent behavior. It finds the tests in a library, runs them, and scores each trajectory against a stored baseline.

```
$ uv run dstf --config testing.json run --backend OpenModelica --report
```

Test report

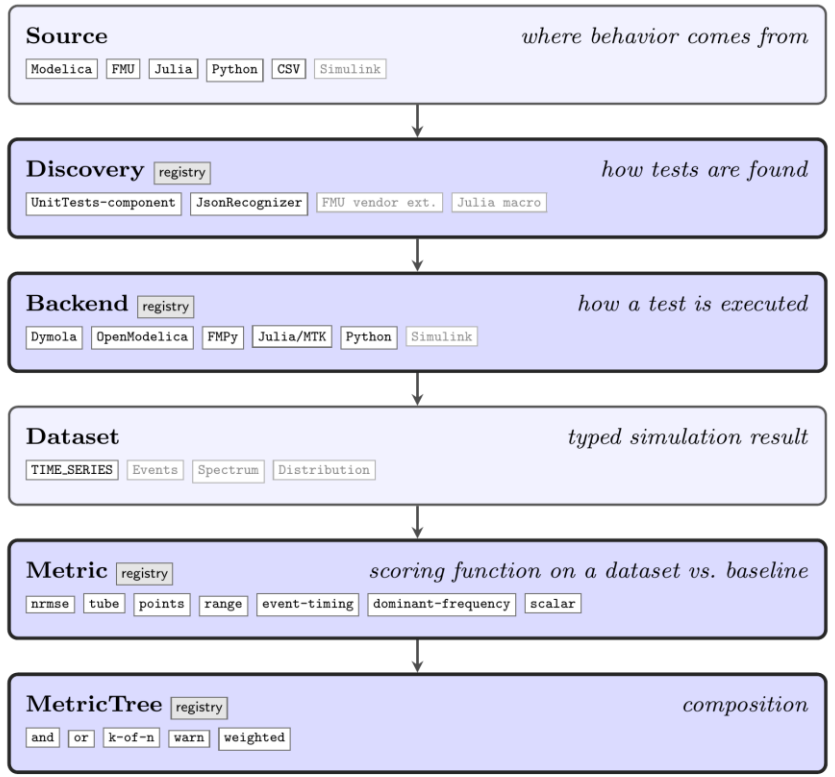
OpenModelica (OpenModelica 1.27.1~2-g6db4671) · MSL 4.1.0 · linux · DSTF 1.0.0 · 2026-10-13 08:20

Final report · 13 tests · ran for 11s · finished 8:21:03 AM

QUEUED	RUNNING	PASS	FAIL	SIM_FAIL	NO_REF	TIMED OUT	NOT RUN
0	0	12	1	0	0	0	0

.* Aa

Key	Reference	Test	Status	Resolution	Worker	Elapsed (s)	Worst NRMSE	Criteria
Pipe	*Pipe*	*Pipe*		*Pipe*	*Pipe*	>1e-4	>1e-4	>1e-4
test_0001	ref_0001	ModelicaTestingLib.Examples.ConstantTest	PASS	mixed	W0	0.7	0.0000e+00	2/2
test_0002	ref_0002	ModelicaTestingLib.Examples.EventTest	PASS	mixed	W0	0.5	0.0000e+00	2/2
test_0003	ref_0006	ModelicaTestingLib.Examples.FrequencyTest	PASS	mixed	W0	0.6	–	1/1
test_0004	ref_0012	ModelicaTestingLib.Examples.HeatRodTest	PASS	mixed	W0	0.6	–	21/21
test_0005	ref_0003	ModelicaTestingLib.Examples.IntervalTest	PASS	mixed	W0	0.6	0.0000e+00	2/2
test_0006	ref_0007	ModelicaTestingLib.Examples.MetricTreeTest	FAIL	mixed	W0	0.6	4.2639e-02	2/4
test_0007	ref_0010	ModelicaTestingLib.Examples.MultiFrequencyTest	PASS	mixed	W0	0.6	–	1/1



How do I mark a model as a test (internal to model)?

Add one component and list the variables to track. DSTF finds it by scanning the library.

MetricTreeTest.mo

```
model MetricTreeTest
  Real x(start=0, fixed=true);
  Real y(start=0, fixed=true);

  Components.UnitTests unitTests(
    n=2,
    x={x, y});

equation
  der(x) = (1 - x)/0.5;
  der(y) = (x - y)/1.0;

  annotation (
    experiment(StopTime=10, Tolerance=1e-6,
      __Dymola_Algorithm="Dassl"),
    ...);
end MetricTreeTest;
```

terminal

```
$ uv run dstf --config testing.json discover
Model ID                                    Vars  StopTime  Method  Source
ModelicaTestingLib.Examples.ConstantTest       2        10  Dassl   both
ModelicaTestingLib.Examples.EventTest          2        10  Dassl   both
  ...
ModelicaTestingLib.Examples.MetricTreeTest     2        10  Dassl   both
  ...
Total: 13 tests
  UnitTests variables: 14
  ...
```

Stop time and tolerance come from the experiment annotation.

How do I run it, and what comes back?

One command. It prints a verdict per test and writes a dashboard, a status file, and a page per test.

terminal

```
$ uv run dstf --config testing.json run --backend OpenModelica
Running 13 tests via persistent OpenModelica workers (parallel=1, timeout=60s/test)
Backend: OpenModelica · linux · DSTF 1.0.0
  ...
Persistent OpenModelica run complete: 13 ok, 0 failed, 0 timed out (12s wall, 9s total
work, 0.7x parallel speedup)
Comparing 13 tests against references...
  ...
  PASS  [0001] ModelicaTestingLib.Examples.ConstantTest
  PASS  [0002] ModelicaTestingLib.Examples.EventTest
  ...
  PASS  [0007] ModelicaTestingLib.Examples.MetricTreeTest
  ...
Results: 13/13 passed
```

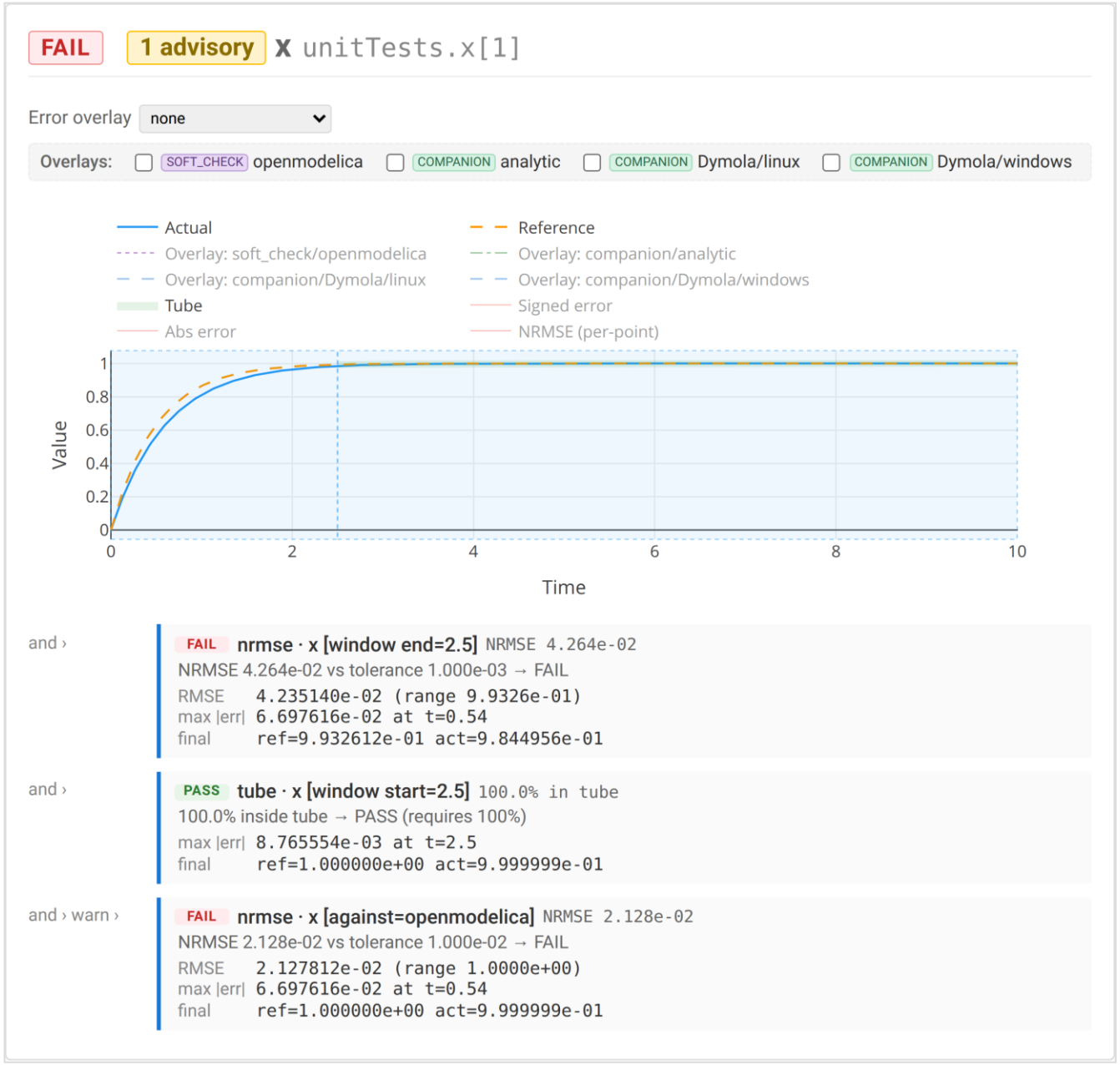
what it wrote

```
testing_output/ModelicaTestingLib/
    OpenModelica/linux/
  dashboard.html
  status.json
  verdicts.json
  reports/ref_0007/
    interactive.html
    comparison_data.json
  test_0006/
    omc_stdout.txt
    result_res.mat
    ...
```

13 tests, 12 s on OpenModelica.

What does a failure tell me?

I changed one time constant from 0.5 s to 0.6 s. The start-up check fails. The steady-state tube still passes.

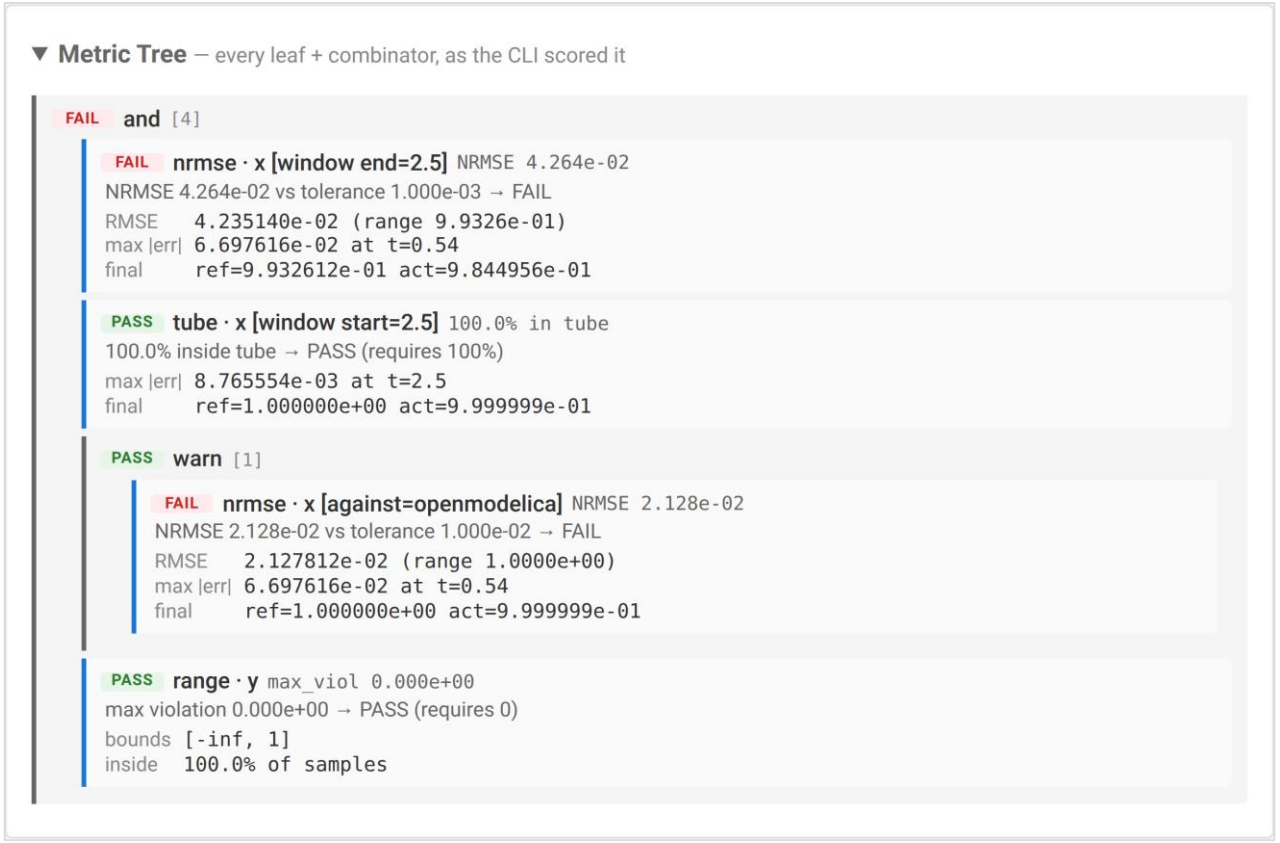


What if one tolerance is not enough?

A test can hold several criteria in a tree. Each one names its metric, its time window and its baseline.

test_spec.json (one entry, reflowed)

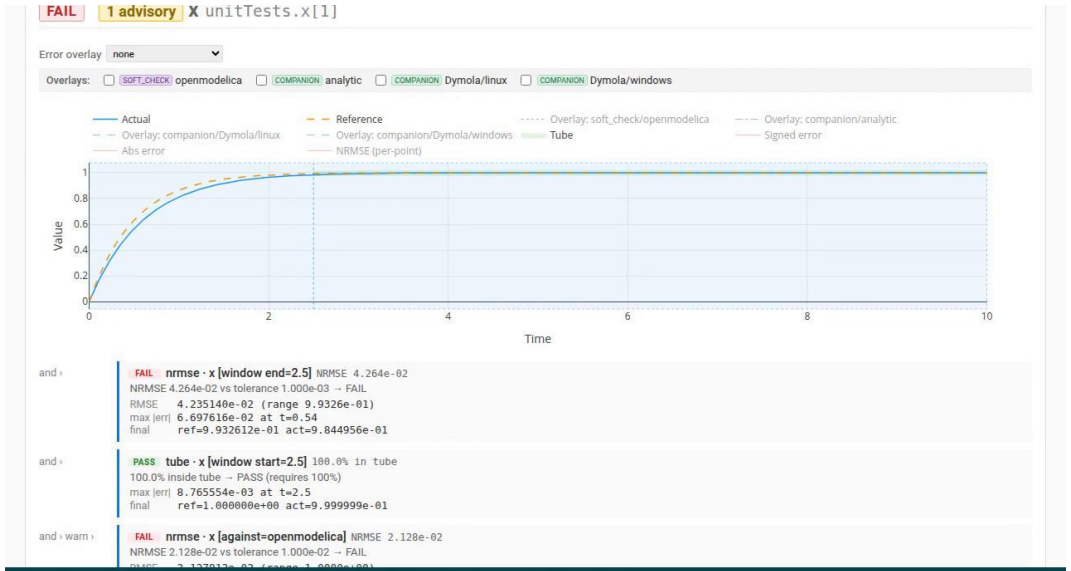
```
"model": "ModelicaTestingLib.Examples.MetricTreeTest",
"metrics": {
  "combinator": "and",
  "children": [
    {"metric": "nrmse",
     "variable": "unitTests.x[1]",
     "tolerance": 0.001, "window": {"end": 2.5}},
    {"metric": "tube",
     "variable": "unitTests.x[1]",
     "tube_width_mode": "rel", "tube_rel": 0.02,
     "tube_min_width": 0.005, "window": {"start": 2.5}},
    {"combinator": "warn",
     "children": [
       {"metric": "nrmse",
        "variable": "unitTests.x[1]",
        "against": "openmodelica", "tolerance": 0.01}]},
    {"metric": "range",
     "variable": "unitTests.x[2]",
     "max_value": 1.0}
  ]
}
```



Demo: break a model, read the failure, edit a criterion

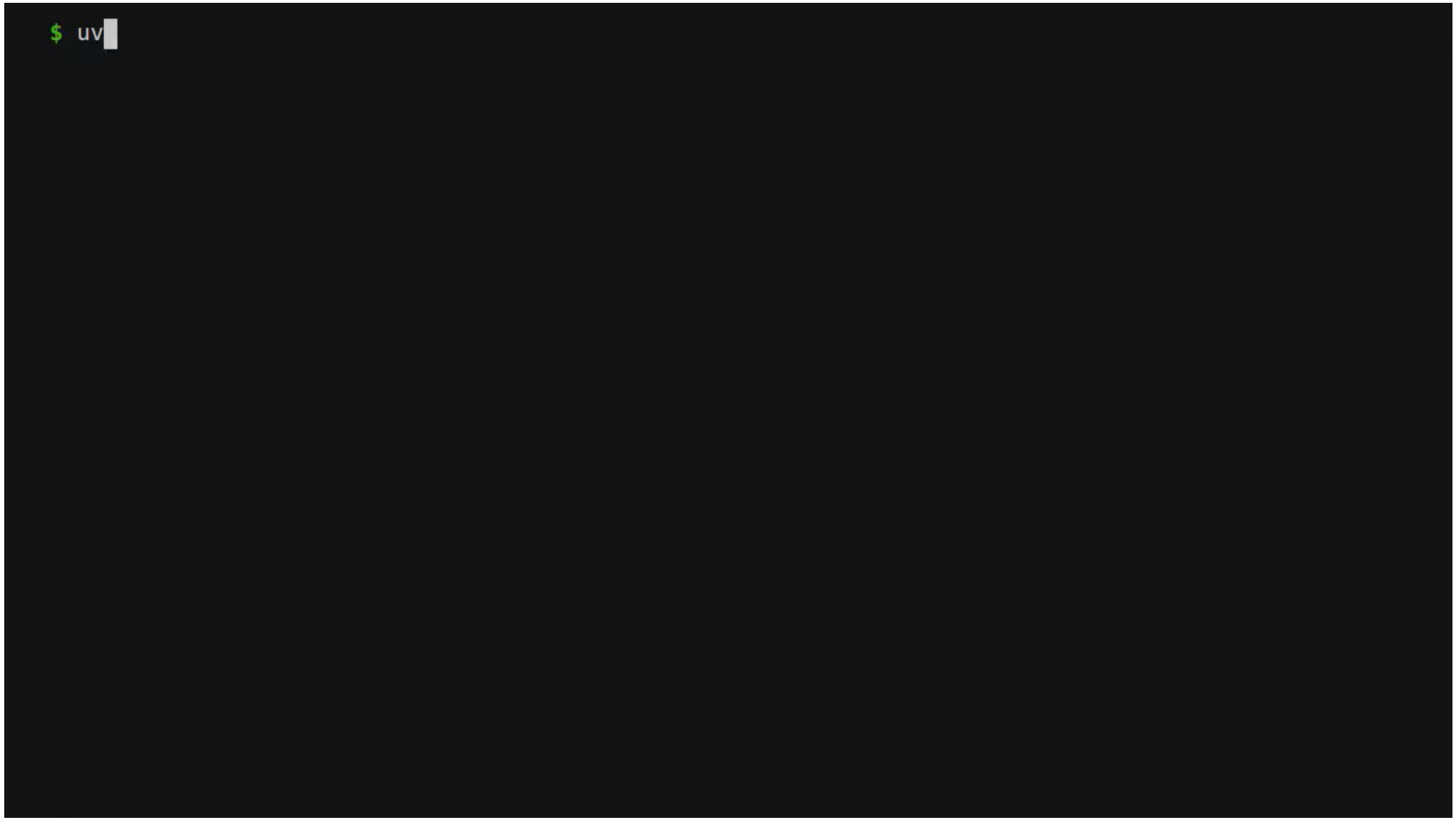
the steps, in order (next slide for pre-recorded)

```
$ uv run dstf --config testing.json discover
# edit MetricTreeTest.mo: (1 - x)/0.5  ->  (1 - x)/0.6
$ uv run dstf --config testing.json run --backend OpenModelica
$ uv run dstf --config testing.json compare --backend OpenModelica --report
# open the test's page from dashboard.html
$ uv run dstf --config testing.json author --backend OpenModelica
# give the range on y a lower bound, review the diff, Save
$ git diff --stat
$ uv run dstf --config testing.json compare --backend OpenModelica
```



Start-up NRMSE fails. The steady-state tube passes. The cross-check warns.

Demo: Pre-recorded



Does it work with my tool?

Five backends run today. The spec, the scoring and the report are the same for all of them.

Backend	Example library	Passed	Wall time
Dymola	ModelicaTestingLib	13 of 13	43 s
OpenModelica	ModelicaTestingLib	13 of 13	12 s
FMPy	3 reference FMUs	3 of 3	under 1 s
Julia / MTK	JuliaMtkTestingLib	8 of 8	31 s
Python	PythonTestingLib	3 of 3	1 s

One run each, one worker, Linux. Not a benchmark.



Can I test against measured data?

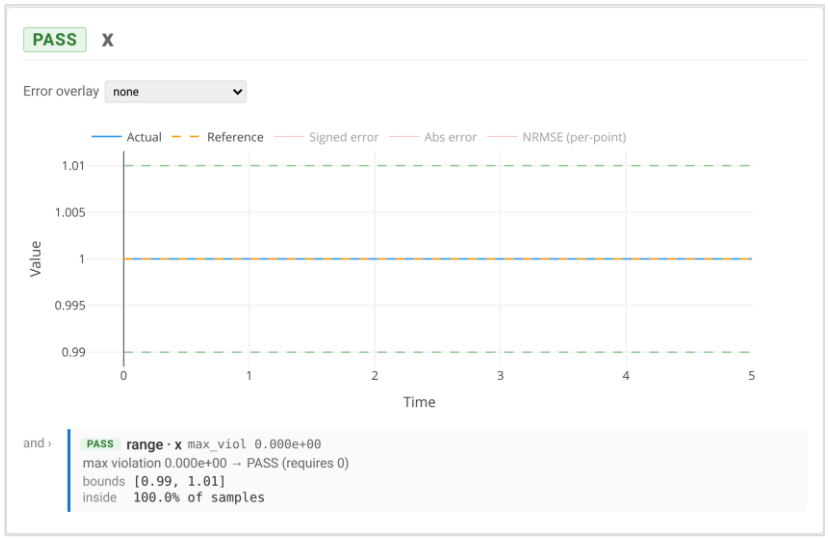
Yes. The Python backend runs any simulate() function. This one reads a CSV and returns its columns.

ConstantCsv.py (comments trimmed)

```python
# examples/python/PythonTestingLib/Examples/ConstantCsv.py
def simulate(stop_time: float, tolerance: float) -> dict:
    del tolerance

    times: list[float] = []
    values: list[float] = []
    with _DATA_FILE.open(encoding="utf-8") as f:
        reader = csv.DictReader(f)
        for row in reader:
            t = float(row["t"])
            if t > stop_time:
                break
            times.append(t)
            values.append(float(row["x"]))

    return {
        "time": times,
        "variables": {"x": values},
    }
```




```csv
# data/constant_experiment.csv
t,x
0.0,1.0
0.25,1.0
0.5,1.0
...
```

What happens to the tests and baselines I already have?

A recognizer tells DSTF what a test looks like in your library. Baselines from another tool import as advisory checks.

a rule, in testing.json

```
// testing.json
"recognizers": [
  {
    "name": "demo:icons-example-as-simulate-only",
    "applies_to": ["modelica"],
    "match": {
      "type": "extends",
      "class_pattern": "*Icons.Example"
    },
    "fields": {
      "simulate_only": {"from": "constant", "value": true},
      "stop_time": {"from": "experiment-annotation", "name": "StopTime"},
      "tolerance": {"from": "experiment-annotation", "name": "Tolerance"}
    }
  }
]
```

a model it matches

```
model SimulateOnlyTest "Pass iff this model simulates — no comparison"
  extends ModelicaTestingLib.Icons.Example;

  Real x(start=0, fixed=true) "Trivial ramp";

equation
  der(x) = 1;

  annotation (
    experiment(StopTime=5, Tolerance=1e-4),
    ...);
```

```
$ uv run dstf --config testing.json run --backend OpenModelica
  ...
  PASS  [new] ModelicaTestingLib.Examples.SimulateOnlyTest

# usage, from --help
$ dstf import-baseline <model_id> <name> <source_path>
```

How does it run in CI?

The exit code carries the verdict, and the run writes JUnit XML, which CI services already display.

the workflow step

```
# README.md, "CI integration"
- run: |
    uv run dstf --config ./testing.json run --report-format junit --output test-results.xml
- uses: actions/upload-artifact@v4
  with:
    name: test-results
    path: test-results.xml
```

exit code 1

when a test or a simulation fails

test-results.xml from the failing run (excerpt)

```
<testsuites tests="13" failures="1" errors="0">
  <testsuite name="ModelicaTestingLib.Examples" tests="13" failures="1" errors="0" skipped="0">
    <testcase name="ModelicaTestingLib.Examples.IntervalTest" classname="ModelicaTestingLib.Examples" />
    <testcase name="ModelicaTestingLib.Examples.MetricTreeTest" classname="ModelicaTestingLib.Examples">
      <failure message="1 criterion failed, with 1 advisory failure" type="RegressionFailure">x (unitTests.x[1]): nrmse · window to 2.5 s · against primary —
NRMSE 4.264e-02 vs tolerance 1.000e-03 → FAIL
      ...</failure>
    </testcase>
    ...
exit code: 1
```

Where does AI fit?

An agent can run DSTF to check its own work, and propose criteria as a patch the tool validates. A person accepts baselines.

from a real agent session (paths shortened)

```
# 1. the agent changes the model, then asks DSTF
$ uv run dstf --config examples/modelica/.../testing.json compare --backend OpenModelica --filter ModelicaTestingLib.Examples.MetricTreeTest
  FAIL  [0007] ModelicaTestingLib.Examples.MetricTreeTest
Results: 0/1 passed

# 2. it adds a criterion as a patch, and the tool applies it
$ uv run dstf --config examples/modelica/.../testing.json spec-update agent_patch.json
Applied 1 op(s) to ModelicaTestingLib.Examples.MetricTreeTest → .../test_spec.json
  add      /metrics/children/-      {'metric': 'range', 'variable': 'unitTests.x[2]', 'min_value': 0.0, 'max_value': 1.01}

# 3. a patch that leaves only warnings is refused
$ uv run dstf --config examples/modelica/.../testing.json spec-update rejected_patch.json
Patch failed: validator rejected the patched tree: metrics: Every tree needs at least one criterion scored against the primary baseline outside a
'warn': a tree of warnings alone can never fail.
```

Full transcript: evidence/agent_transcript.md

How do I try it, and how can I help?

It is open source under the MIT license. These four lines run the example library. (PyPI release soon)

```
git clone https://github.com/ORNL-Modelica/DynamicSystemsTestingFramework
cd DynamicSystemsTestingFramework
uv sync --extra dev --extra om --extra fmpy --extra author --locked
uv run dstf --config examples/modelica/ModelicaTestingLib/Resources/ReferenceResults/testing.json run
```

1

Run it on a library you maintain, and open an issue for what breaks.

2

Write a backend or a recognizer for your tool. A backend is one Python class.

3

Tell me what is missing before you would use it.

the repository

How is this different from other regression options?

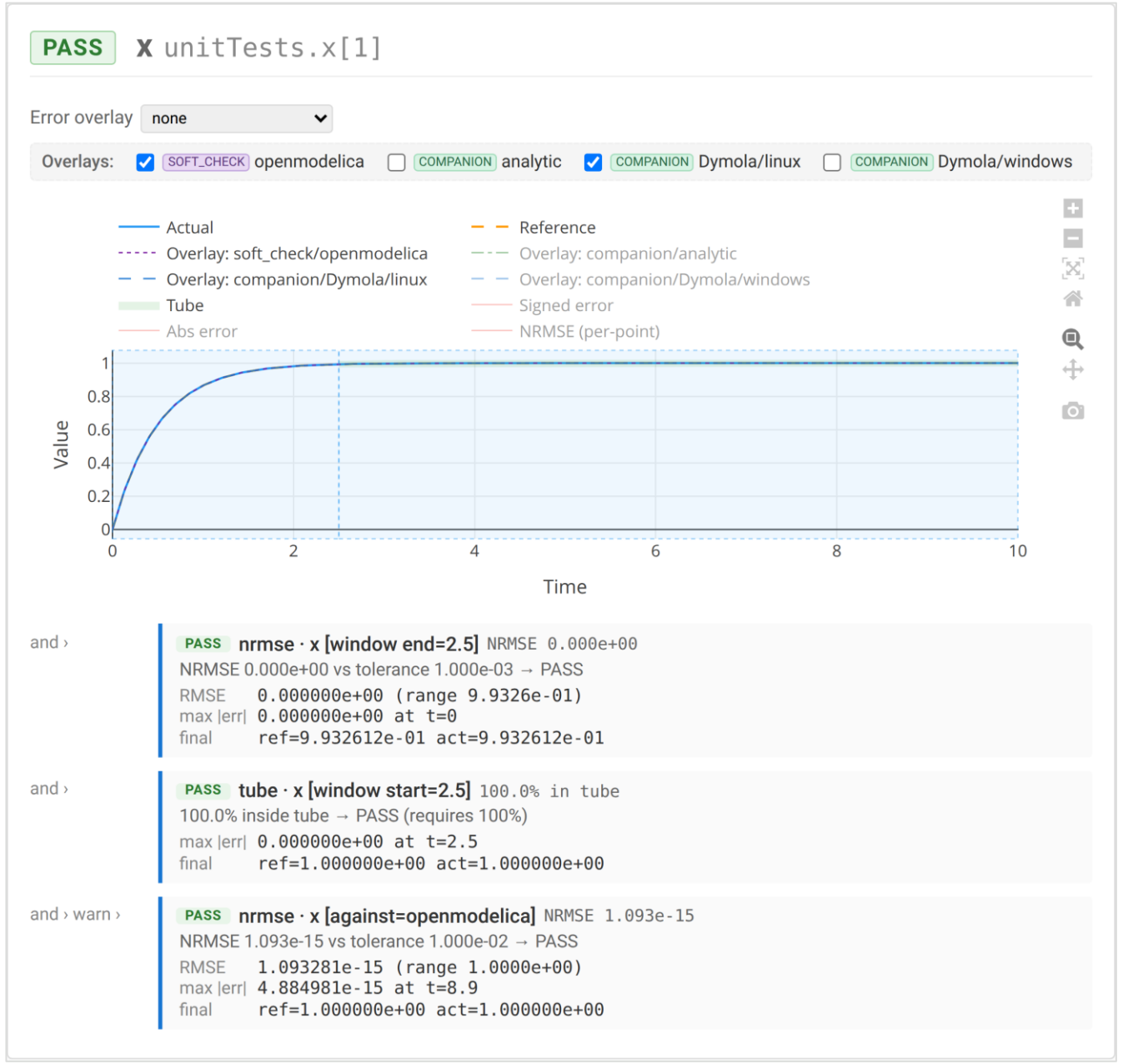
Each of those tests one ecosystem against one reference under one rule. DSTF is built to run beside them.

Tool	What it runs	How it compares
BuildingsPy	Buildings library, stored Dymola results	pyfunnel tube, axis-wise tolerances
csv-compare	Modelica Standard Library	tube, widened in time at discontinuities
OpenModelica library testing	a corpus of libraries, OpenModelica runtime	one reference, one rule
ReSim	one library across several Modelica tools	one reference, one rule
DSTF	**Modelica, FMU, Julia, Python, recorded data**	**a tree of criteria, three baseline roles**

As described in the paper's related-work section.

What kinds of baseline can a test have?

Three. A primary can fail the test. A soft check only warns. A companion is drawn and never scored.



Can it catch a regression the trajectory does not show?

Each baseline stores the translation statistics. A run warns when they change, and a scalar criterion can make one fail the test.

terminal: one state added, trajectories unchanged

```
$ uv run dstf --config testing.json run --backend Dymola --filter "*MetricTreeTest"
  ...
  PASS  [0005] ModelicaTestingLib.Examples.MetricTreeTest [WARN]

Results: 1/1 passed
  Structural warnings: 1

Structural Warnings:
  ModelicaTestingLib.Examples.MetricTreeTest
    Continuous states: 2 -> 3
    Scalar unknowns: 4 -> 5
    Scalar equations: 4 -> 5
```

Structural Warnings (3)

Field	Reference	Current
Continuous states	2	3
Scalar unknowns	4	5
Scalar equations	4	5

a criterion on one number

```
// test_spec.json, SimpleTest
{
  "metric": "scalar",
  "variable": "unitTests.x[1]",
  "quantity": {"reduce": "final"},
  "tolerance": 0.001,
  "tolerance_mode": "abs"
}
```

How exactly are two trajectories compared?



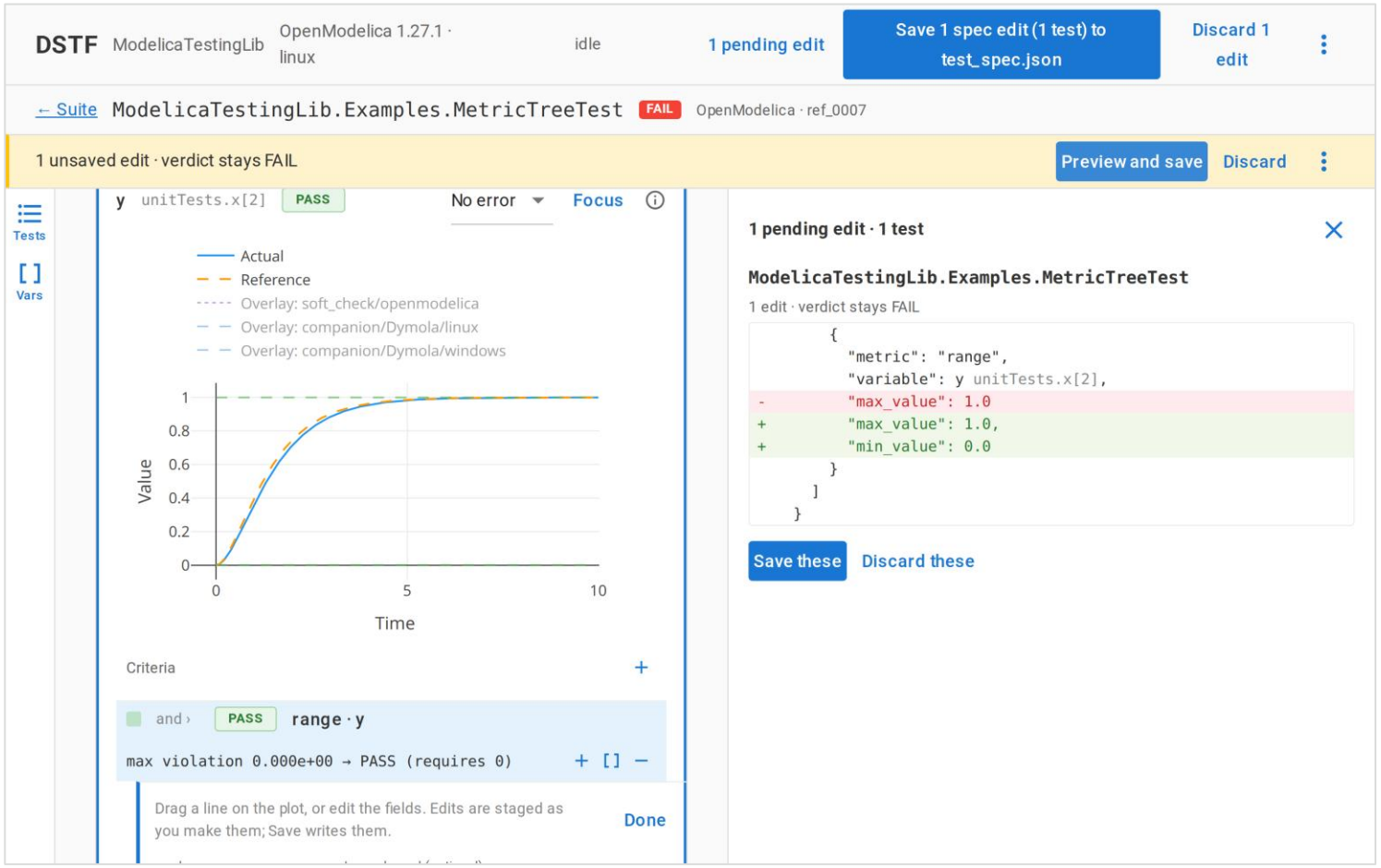
1 The run is interpolated onto the reference time grid.

2 Comparison is piecewise between events, so a jump is never averaged across.

3 A tube bounds y only. Time shifts are scored by points and event-timing.

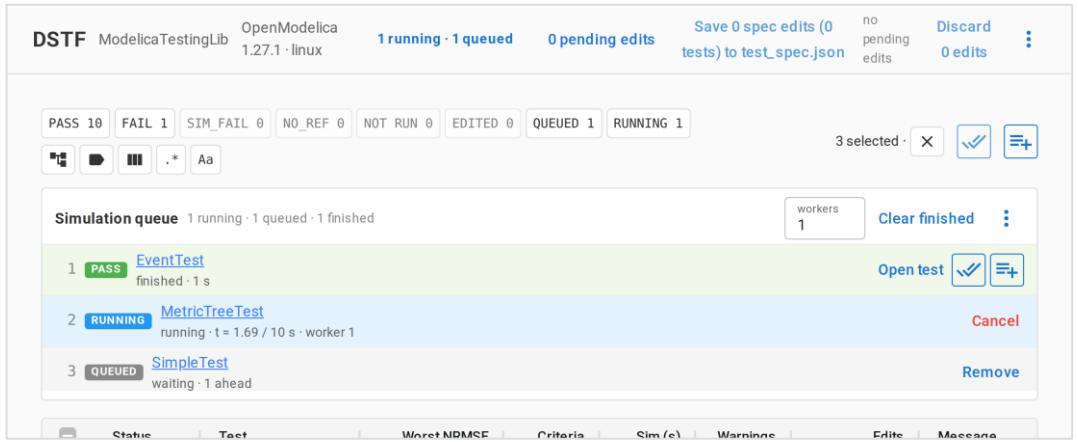
How do I change a criterion without hand-editing JSON?

dstf author opens the last run in a browser. You edit a criterion beside its plot, review the diff, and save.

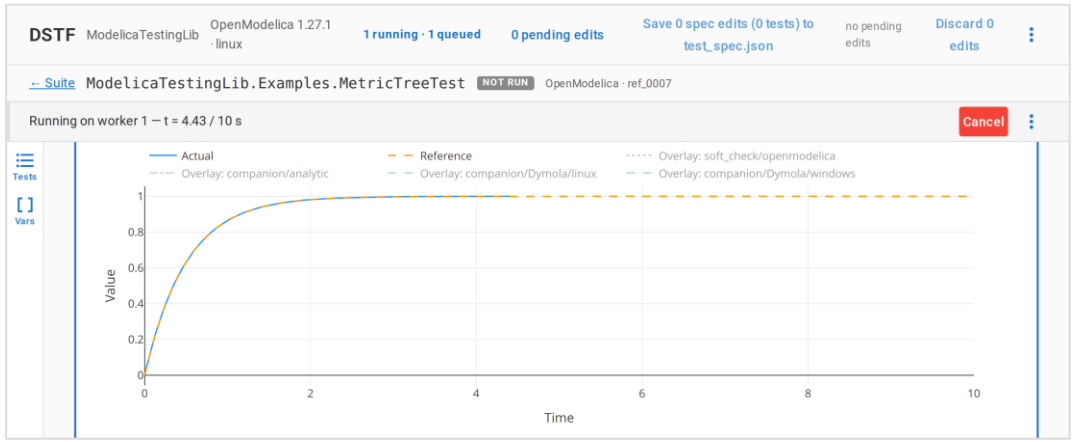


What is new since the paper?

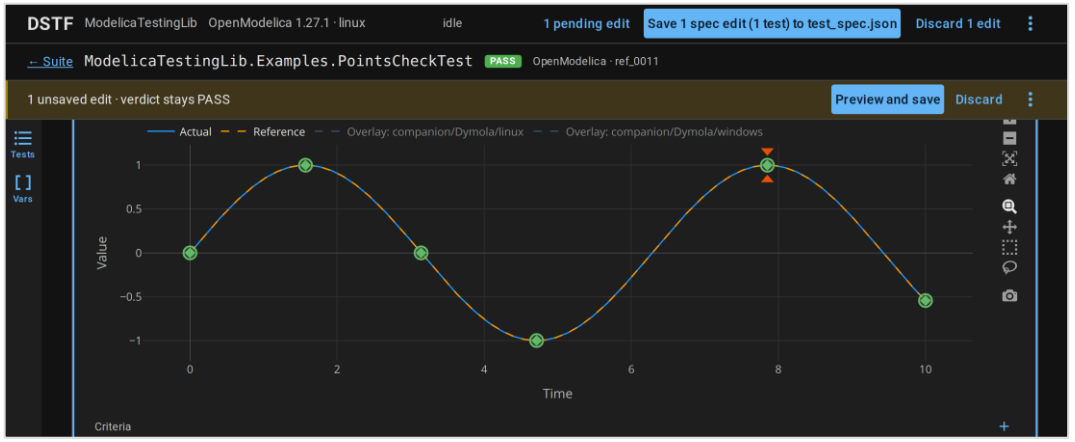
A queue of simulations



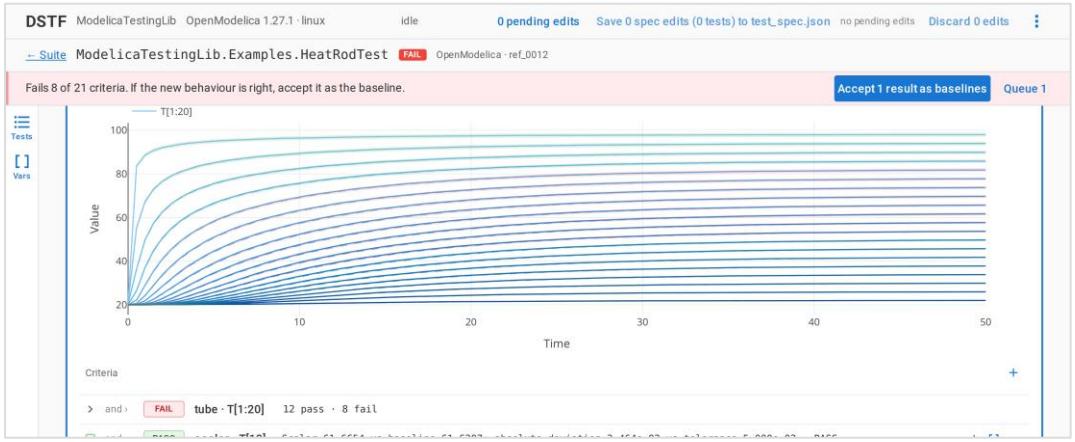
Drawn while the simulator runs



Ctrl+click places a checkpoint



One criterion over an array



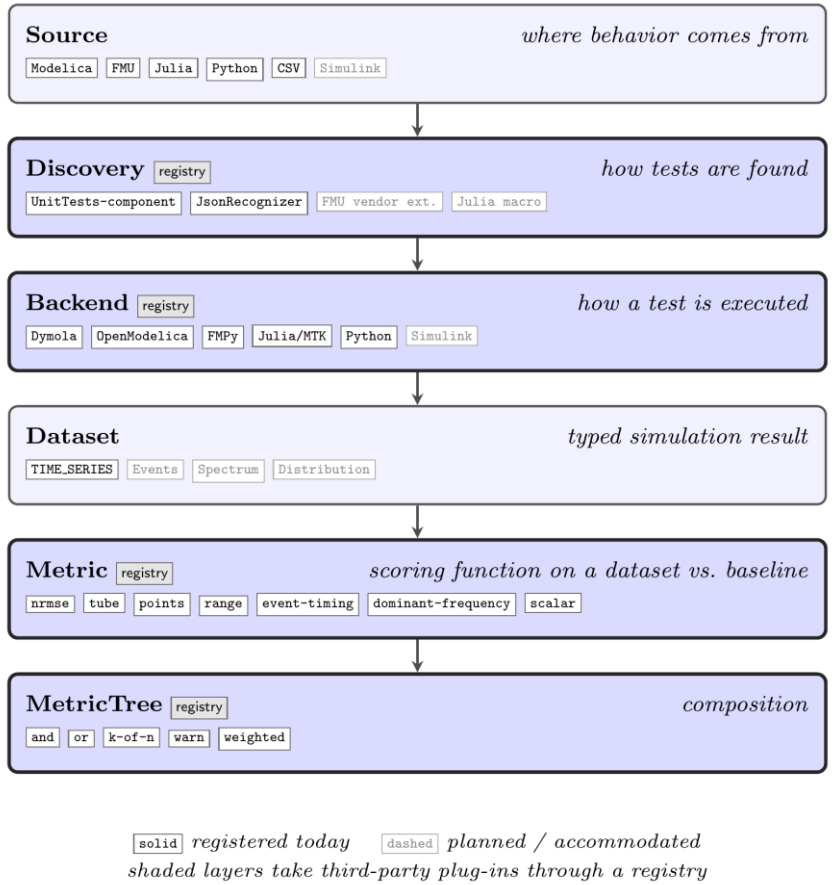
How do I add a simulator or a metric?

Backends, recognizers, metrics and combinators register by name. A separate package adds one through an entry point.

a plug-in package declares

```
# docs/extensibility.md: your package's pyproject.toml
[project.entry-points."dstf.plugins"]
acme-backend = "acme_dstf.backend"
acme-metrics = "acme_dstf.metrics:register"
```

Registry	What you add
Recognizer	what a test looks like in a library
Backend	how a test is run
Metric	how one criterion scores
Combinator	how criteria roll up



Has it been used on anything real, and what is next?

TRANSFORM's test suite runs on it. Three things come next.

```
$ dstf --config TRANSFORM-UnitTests/ReferenceResults/testing.json discover
  ...
Total: 331 tests
  UnitTests variables: 877
  ...
  Source 'unit_tests': 313
```

331

tests in TRANSFORM, found from its own UnitTests components

A versioned schema for every file DSTF writes, so tools and agents can read results without guessing.

Cross-checks between every pair of backends, without an author asking for each one.

Metrics for ensembles: a quantile band, a Kolmogorov-Smirnov test.

What else does it do?

terminal

```
# four workers: 13 tests in 9 s here
$ uv run dstf --config testing.json run --parallel 4
Persistent OpenModelica run complete: 13 ok, 0 failed, 0 timed out (9s wall, 20s total work, 2.2x parallel speedup)

# rerun only what failed, and merge it into the last report
$ uv run dstf --config testing.json run --rerun failed --merge --report
--rerun failed: selected 1 of 13 tests

# rescore after a criteria change, with no simulation
$ uv run dstf --config testing.json compare
```

Also: baselines split by tool and operating system, a live dashboard during a run, a dark theme.

What would it take for you to try it?

github.com/ORNL-Modelica/DynamicSystemsTestingFramework

```
git clone https://github.com/ORNL-Modelica/DynamicSystemsTestingFramework
cd DynamicSystemsTestingFramework
uv sync --extra dev --extra om --extra fmpy --extra author --locked
uv run dstf --config examples/modelica/ModelicaTestingLib/Resources/ReferenceResults/testing.json run
```

Run it on your library and tell me what breaks.

greenwoodms@ornl.gov

Paper: A Unified Dynamic System Testing Framework, American Modelica Conference 2026