BERKELEY LAB
Bringing Science Solutions to the World

IBPSA Project 2
BOPTEST

U.S. DEPARTMENT OF ENERGY
Office of Science

IBPSA Project 2 Expert Meeting
Task 3: Test Cases

Ettore Zanetti ezanetti@lbl.gov

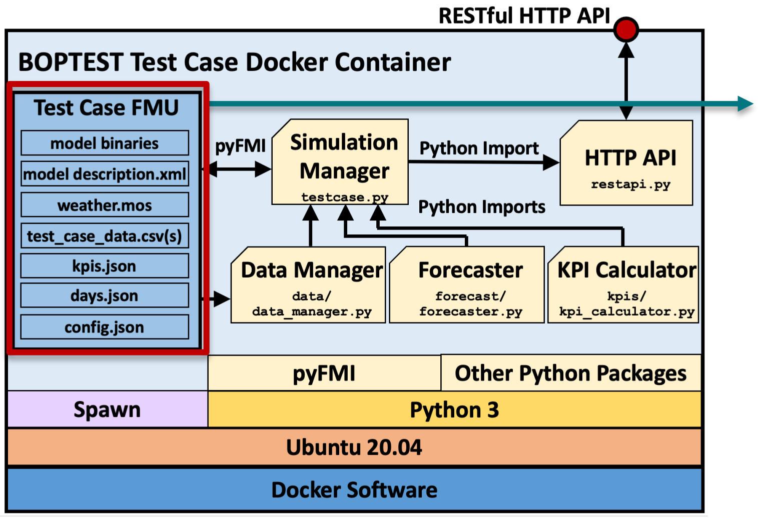
11-20/21-2024

Task 3: Test Cases, Session 2

- Test Case compilation and execution tools (30min)
- Test Case Development and review process (30min)
- Final thoughts (10-15min)

Task 3: Test Cases, Session 2
Test Case compilation and execution tools

History and Current State



- BOPTEST uses co-simulation FMUs to encapsulate test cases
- Most of the test cases are compiled using JModelica as at the time it was the most reliable open source Modelica compiler
- Unfortunately JModelica has been discontinued since Dec 2019
- Currently BOPTEST includes Test Case compilation in the unit tests for most test cases
- This means that most test cases are still using Modelica 3.2.3 and older versions of the Modelica libraries (Buildings, IDEAS)

Task 3: Test Cases, Session 2
Test Case compilation and execution tools

Current Developments

- The only open source compiler available currently is OpenModelica (OM)
- Currently in development branch 422 Test Cases have been updated to Modelica 4.0 and relative version of the libraries, and partially tested with OM

TestCase	Compile Dymola	Simulate Dymola	Compile OMEdit	Simulate OMEdit	OM FMU Compile	OM FMU Simulate in pyfmi
bestest_air	v	v	v	v	x	x
bestest_hydronic	x	v	v	v	o	o
bestest_hydronic_heat_pump	v	v	v	v	o	o
singlezone_commercial_hydronic	v	v	v	v	o	o
multizone_residential_hydronic	v	v	v	v	o	o
multizone_office_simple_air	v	v	v	o	o	o
twozone_apartment_hydronic	v	v	v	o	o	o

- Even though most models compile and simulate in OM, FMU compilation remains an open question because currently OM co-simulation FMUs are under development.

- When Multizone Office Simple Hydronic was added to the available test cases Optimica was made available as compiler option for BOPTEST FMUs

Task 3: Test Cases, Session 2
Test Case compilation and evecution tools

Future Directions

Issues:

- Jmodelica not supported anymore and models stuck at Modelica 3.2.3
- OpenModelica (OM) only experimentally supports co-simulation FMUs
- PyFMI has a bug on event handling for model exchange FMUs

Possible solutions:

Use Model exchange FMUs

- Use PyFMI do step and handle events in BOPTEST
- Change PyFMI with FMPy or OMSimulator and export model exchange FMUs with OM

Keep current implementation

- Use OM when possible
- Use Optimica compilation option for all other models

Task 3: Test Cases, Session 2
Test Case Development and review process

Comments from previous test case development efforts

- Let’s face it, Modelica development is not easy. Especially if multiple tools are considered (Dymola, Optimica, OM)
- Volunteer effort comes in bursts of spare time, and monthly meetings may be too slow to quickly address issues and get feedback
- BOPTEST overhead on top of Modelica model:
  - Boundary conditions for forecasts, typical periods, documentation, interfaces (Bacnet, Semantics),…
  - Test case review
- Data availability and model calibration

Task 3: Test Cases, Session 2
Test Case Development and review process

How can we improve test case development and speed it up?

- Prepare onboarding process for new test case developments to help understand BOPTEST test case nuances
- Have quick feedback meetings outside monthly meetings upon request “office hours” and set up permanent discussion tab on repository
- Proposed cheat sheet with ”typical values” (HVAC sizing, buildings properties, internal gains, etc..)
- Proposed updated test case review document that can be also used by test case developer as general guideline
- Proposed test case stress test script that can be used by developer to quickly ”rattle” test case with different control sequences to check for robustness

Task 3: Test Cases, Session 2
Test Case Development and review process

How can we improve test case development and speed it up?

- **Prepare onboarding process for new test case developments to help understand BOPTEST test case nuances**

Help test case developer navigate BOPTEST test case development depending at what level they start:

- Have test case developer present existing Modelica model or reference case study if Modelica model needs to be developed
- Provide feedback on Modeling requirements using review document and point developer to useful models among Modelica libraries
- Set up discussion tab on GitHub that can be used during development
- Guide developer through BOPTEST reseources, utility scripts and test case structure

Task 3: Test Cases, Session 2
Test Case Development and review process

How can we improve test case development and speed it up?

- **Have quick feedback meetings outside monthly meetings upon request "office hours"**

- Use test case discussion tab to post quick questions and setup meeting in case questions cannot be addressed right away

- **Proposed cheat sheet with "typical values" (buildings properties, internal gains, etc..)**

- Put together lists of "typical" values and best practices to help modelers. Help asked to take data from international / country standard to be put in this document

Task 3: Test Cases, Session 2
Test Case Development and review process

How can we improve test case development and speed it up?

- **Proposed updated test case review document that can be also used by test case developer as general guideline**
- New review document will be used also as checkbox by test case developer
- Review document became more detailed. Some questions might not be relevant for climate or HVAC considered in case study (i.e. moisture condensation in heating only radiant systems)
- Create examples of filled documents that can be shared with new test case developers

Task 3: Test Cases, Session 2
Test Case Development and review process

How can we improve test case development and speed it up?

- **Proposed test case stress test script that can be used by developer to quickly ”rattle” test case with different control sequences to check for robustness**

```
Input Json: "test_case_name": "bestest_air”
            "compare_with_baseline": false
            "simulation_step": 3600
            "show_plots": true
            ”forecast_check": true
```

"scenarios": [{"time_period": "all", "electricity_price": "constant”,“custom_input”=true}, {”start_time": 0, “stop_time”: 86400, "electricity_price": "constant”}]

"input_list": ["con_oveTSetCoo_activate","con_oveTSetCoo_u"]

Input .csv: Time series of custom inputs that should be used in a specific scenarios

Task 3: Test Cases, Session 2
Test Case Development and review process

How can we improve test case development and speed it up?

- **Proposed test case stress test script that can be used by developer to quickly ”rattle” test case with different control sequences to check for robustness**
  - The script will instantiate the test case and collect meta data in inputs needed (min and max)
  - It will generate random sequence for given inputs in .json file using latin hypercubic sampling given time horizon and step to explore control space
  - If model crashes during simulation it will return breaking sequence, FMU log, and .mat file
  - It will generate plots of measurements and inputs. If baseline option is set to true it will compare with baseline
  - If forecast_check set to true it will save forecast data for the scenarios that can be compared with data in Modelica (weather can be done automatically)