SCIENCE
PASSION
TECHNOLOGY

FMU Simulation Environment for Cyber-Physical Systems Co-Design

Introduction & Use Case

David Kaufmann, MSc

SS22 04/04/2022

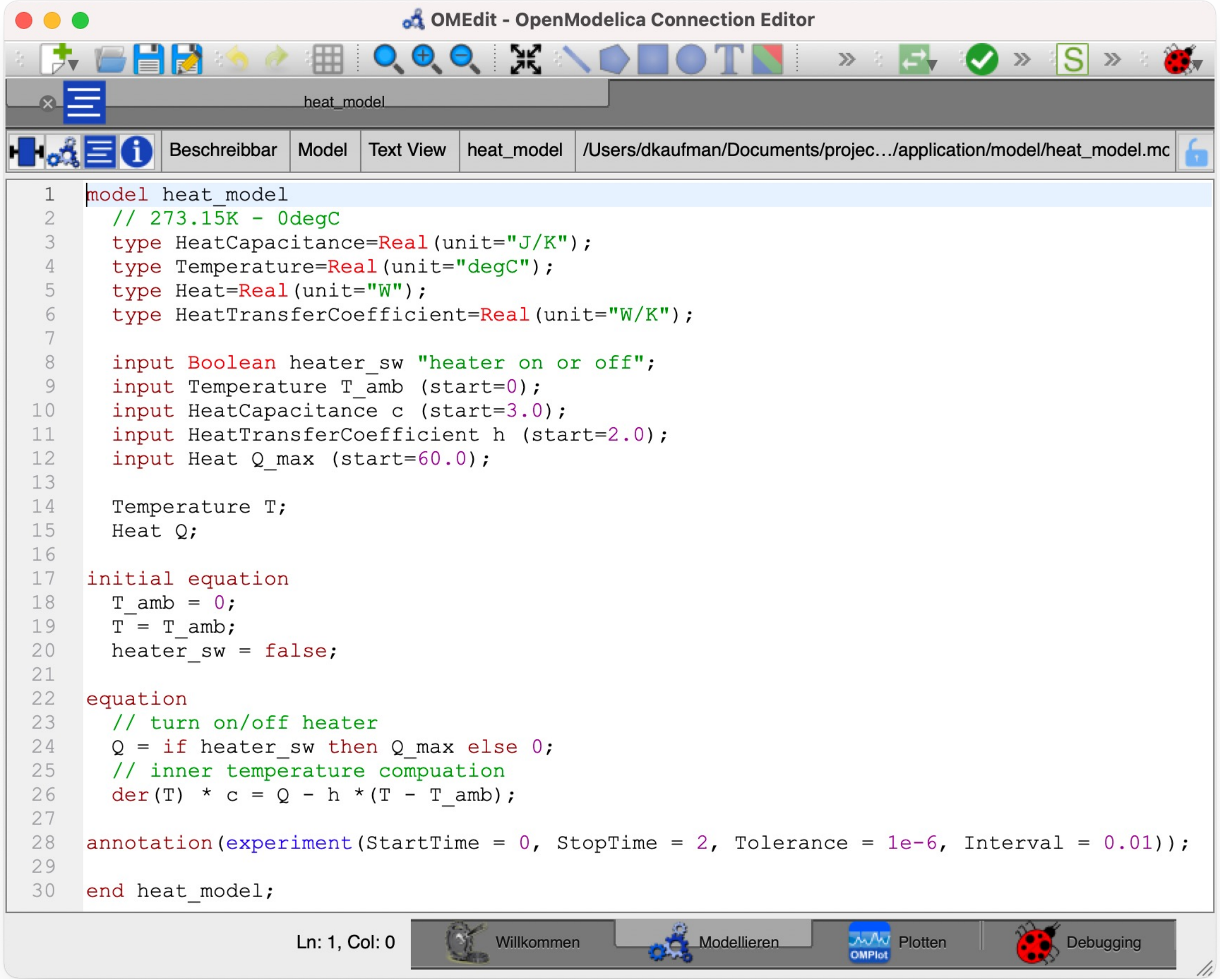
David.Kaufmann@ist.tugraz.at

Overview

- Modeling in OpenModelica
- Functional Mock-up Unit (FMU) Standard
- FMU Simulation Environment
- Use Case Heater Model
- ASP Diagnose Tool

Modeling in OpenModelica

- OMEdit editor for creating models
- Use **Linux** pre-built OpenModelica VirtualBox:
  - Link to Linux VB release
- Model the system according to OpenModelica standard
- Input
  - Define inputs with “input” statement to have access in FMU model.
- Output:
  - All used variables can be read as output signals.


OMEdit - OpenModelica Connection Editor
heat_model
Beschreibbar
Model
Text View
heat_model
/Users/dkaufman/Documents/projec…/application/model/heat_model.mc
Ln: 1, Col: 0
Willkommen
Modellieren
OMPlot
Plotten
Debugging

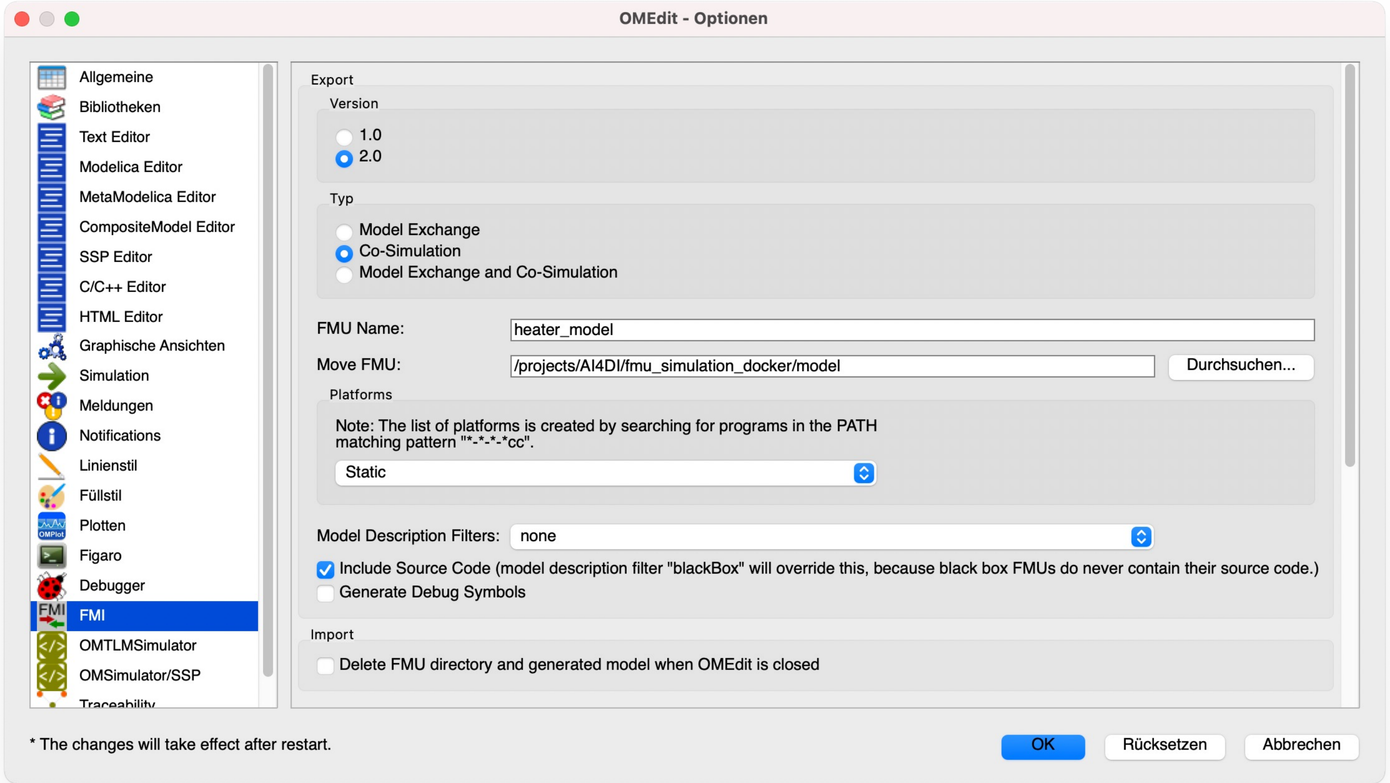
```
model heat_model
  // 273.15K - 0degC
  type HeatCapacitance=Real(unit="J/K");
  type Temperature=Real(unit="degC");
  type Heat=Real(unit="W");
  type HeatTransferCoefficient=Real(unit="W/K");

  input Boolean heater_sw "heater on or off";
  input Temperature T_amb (start=0);
  input HeatCapacitance c (start=3.0);
  input HeatTransferCoefficient h (start=2.0);
  input Heat Q_max (start=60.0);

  Temperature T;
  Heat Q;

initial equation
  T_amb = 0;
  T = T_amb;
  heater_sw = false;

equation
  // turn on/off heater
  Q = if heater_sw then Q_max else 0;
  // inner temperature compuation
  der(T) * c = Q - h *(T - T_amb);

annotation(experiment(StartTime = 0, StopTime = 2, Tolerance = 1e-6, Interval = 0.01));

end heat_model;
```

Modeling in OpenModelica

Co-Simulation:

- FMUs contain local solvers
- Enables step by step simulation

Export FMU:



OMEdit - Optionen

Export

Version

○ 1.0
● 2.0

Typ

○ Model Exchange
● Co-Simulation
○ Model Exchange and Co-Simulation

FMU Name: heater_model

Move FMU: /projects/AI4DI/fmu_simulation_docker/model

Platforms

Note: The list of platforms is created by searching for programs in the PATH matching pattern "*-*-*-*cc".

Static

Model Description Filters: none

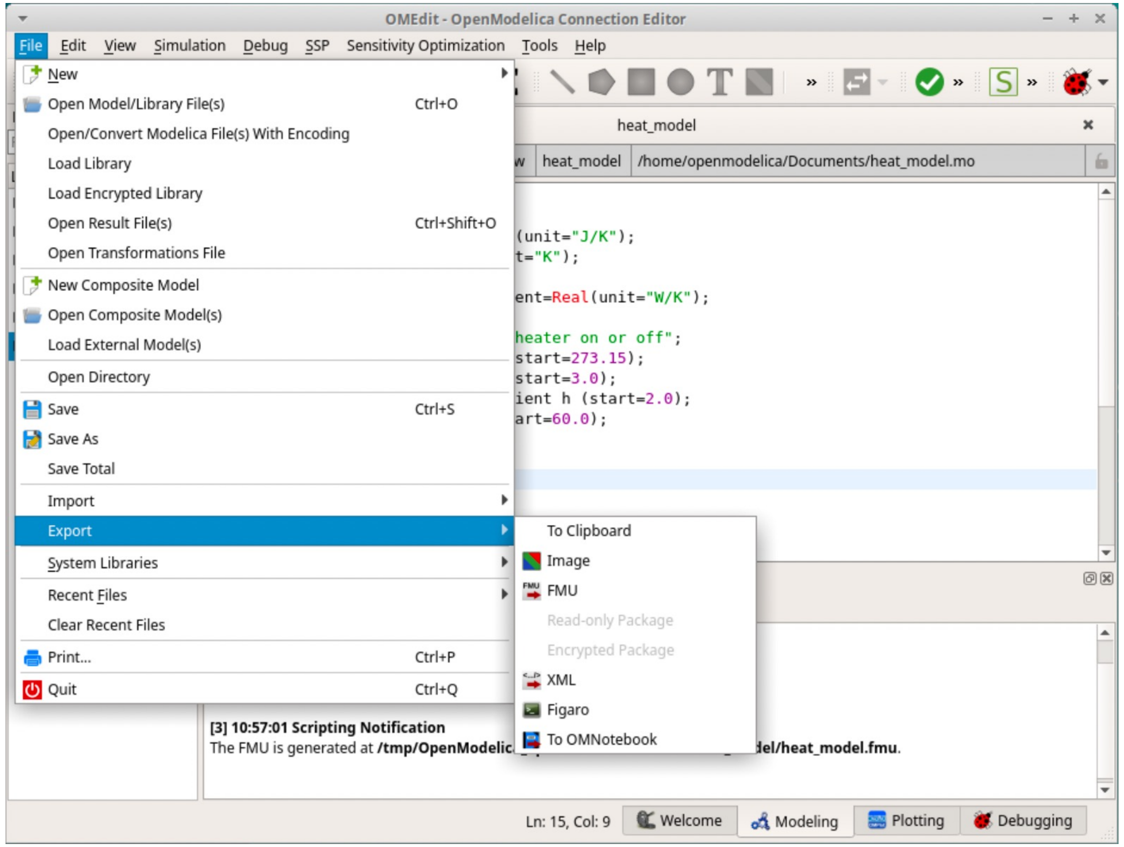
☑ Include Source Code (model description filter "blackBox" will override this, because black box FMUs do never contain their source code.)

□ Generate Debug Symbols

Import

□ Delete FMU directory and generated model when OMEdit is closed

* The changes will take effect after restart.



OMEdit - OpenModelica Connection Editor

heat_model /home/openmodelica/Documents/heat_model.mo

```
(unit="J/K");
t="K");

ent=Real(unit="W/K");

heater on or off";
start=273.15);
start=3.0);
ient h (start=2.0);
art=60.0);
```

[3] 10:57:01 Scripting Notification
The FMU is generated at /tmp/OpenModelic... del/heat_model.fmu.

Ln: 15, Col: 9

Functional Mock-up Unit (FMU)

The Functional Mock-up Interface (or FMI) defines a standardized interface to be used in computer simulations to develop complex cyber-physical systems.

FMI defines an interface that is implemented by an executable called a Functional Mock-up Unit (FMU)

An FMU contains following:

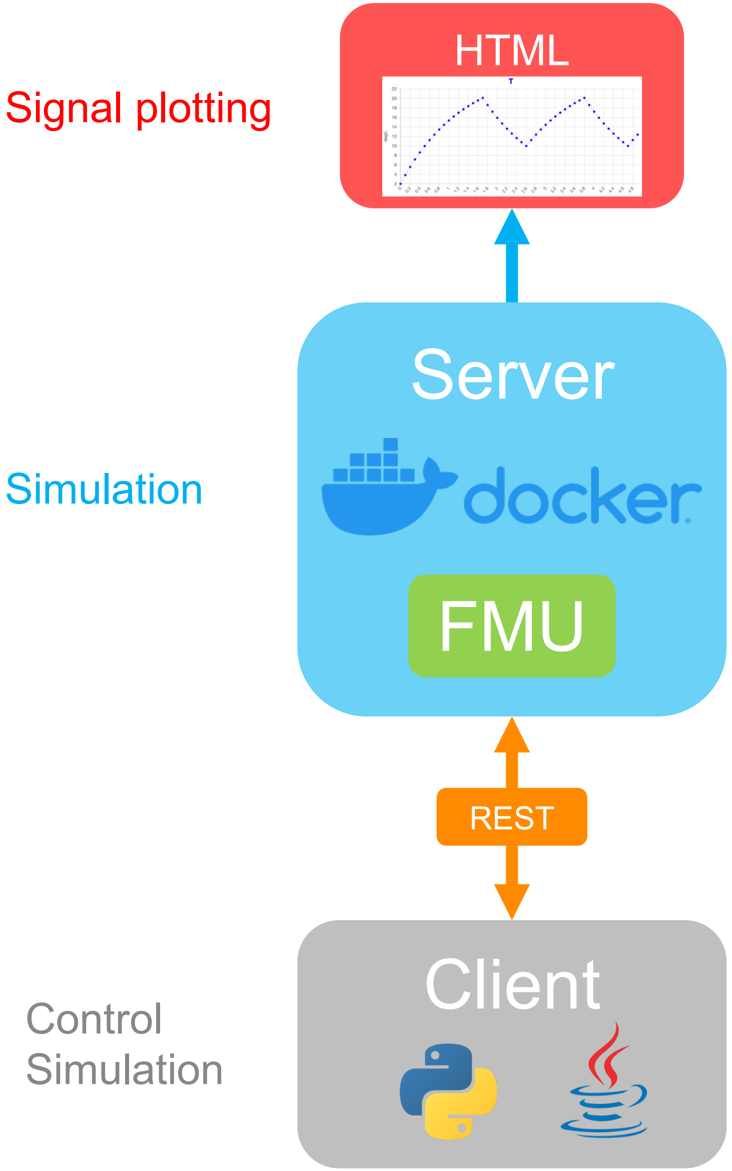
- A model description XML file:
  - Model information.
  - Variable definitions as type, unit and description.
  - General model information as name, generation tool and FMI version.
- Model equations:
  - Differential equations, algebraic relations and discrete equations.
  - Represented in C functions and binaries.

fmi Functional Mock-Up Interface

FMU Simulation Environment

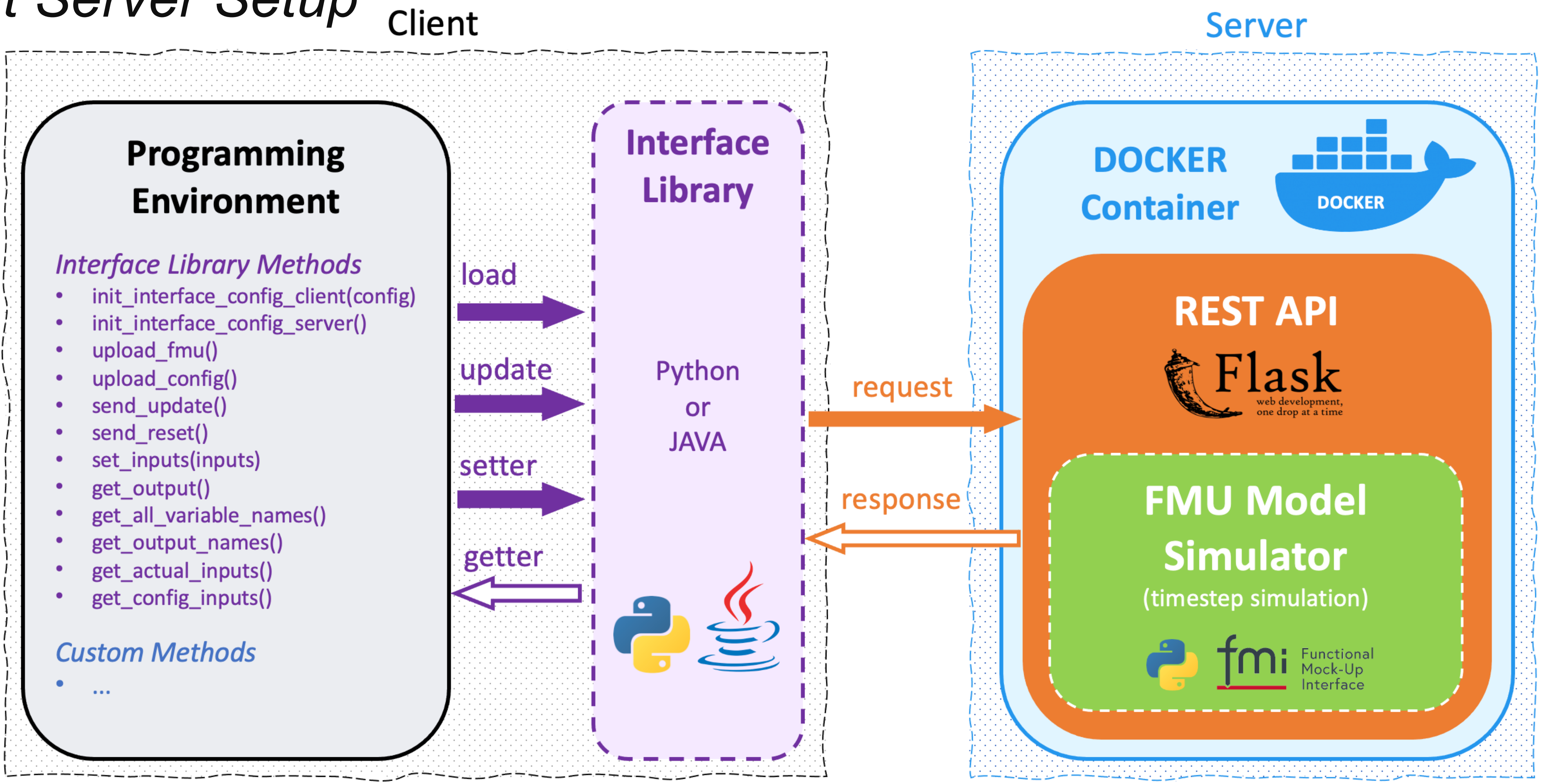
General Idea

- Provide user friendly simulation environment for FMUs.
- Enable a step-by-step simulation (Co-Simulation FMU).
- Enable coupling of multiple simulated FMU models.
- Build a client server environment to access from different programming environments (Python or Java).
- Dockerise simulation environment as server with a REST API.
- Provide a client interface to communicate with the server.
- Provide signal chart drawing.
- Clone Gitlab repository:
  - git clone https://ss22_mts:x-sykzfA_Ld1PVArrwsz@git.ist.tugraz.at/MBR/AI4DI/fmu_simulation_environment



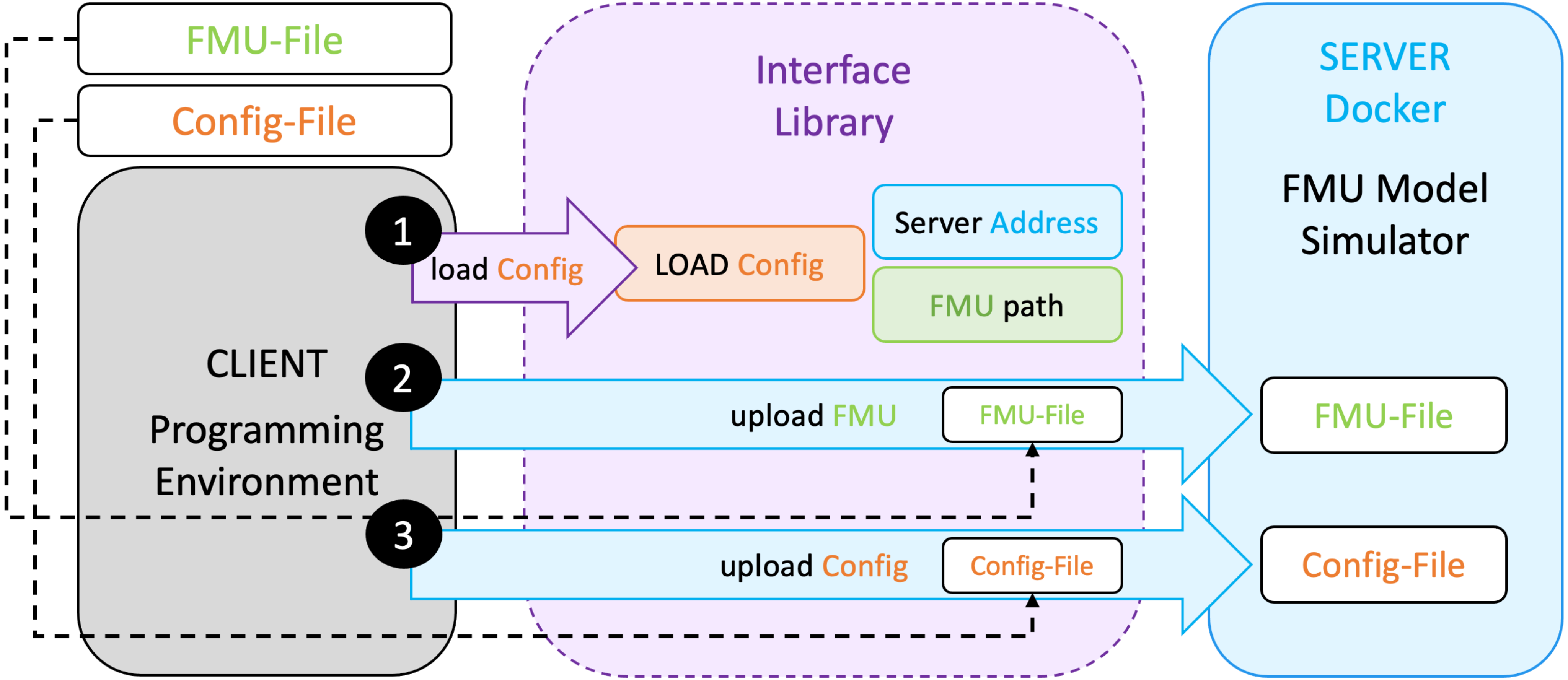
FMU Simulation Environment

Client Server Setup



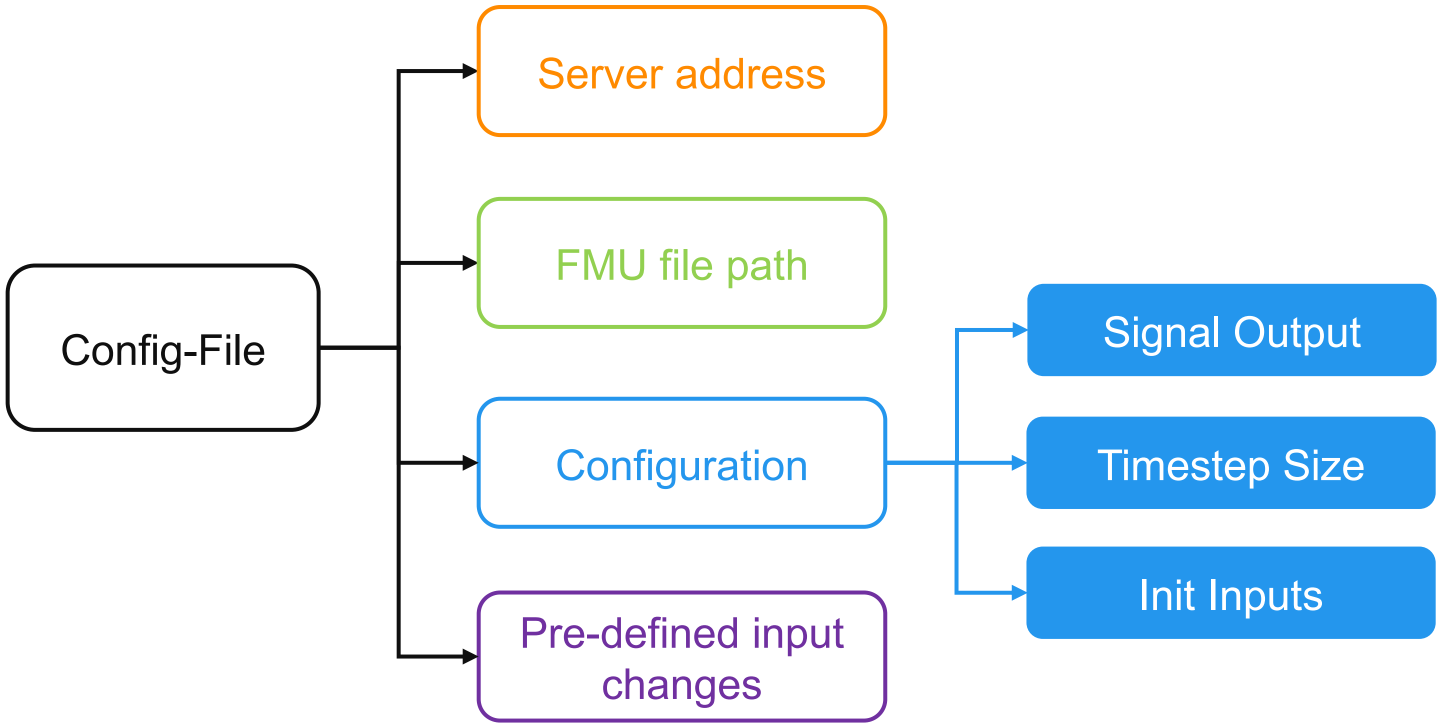
FMU Simulation Environment

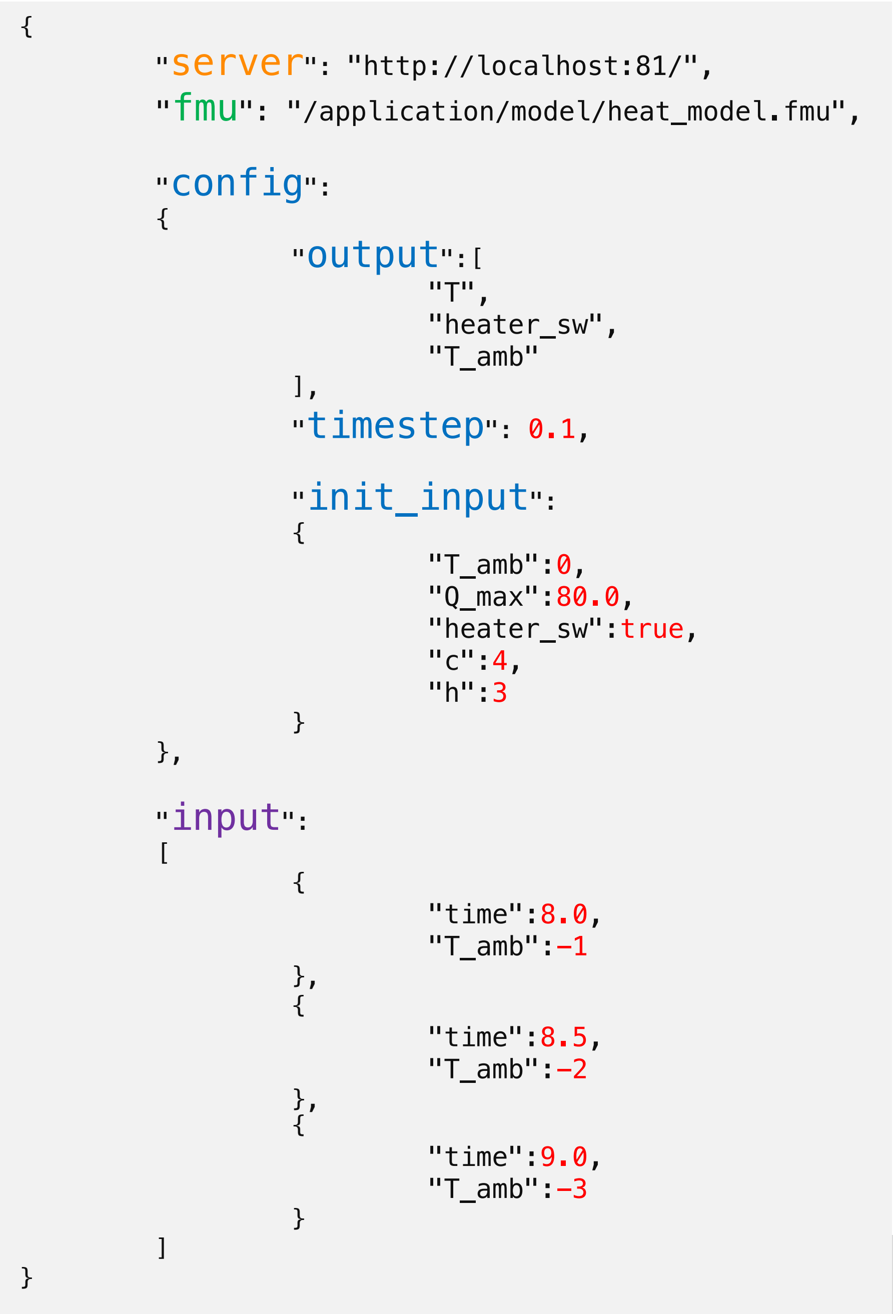
Configure and Upload Simulation Model



FMU Simulation Environment

Model Configuration Structure





```
{
        "server": "http://localhost:81/",
        "fmu": "/application/model/heat_model.fmu",

        "config":
        {
                "output":[
                        "T",
                        "heater_sw",
                        "T_amb"
                ],
                "timestep": 0.1,

                "init_input":
                {
                        "T_amb":0,
                        "Q_max":80.0,
                        "heater_sw":true,
                        "c":4,
                        "h":3
                }
        },

        "input":
        [
                {
                        "time":8.0,
                        "T_amb":-1
                },
                {
                        "time":8.5,
                        "T_amb":-2
                },
                {
                        "time":9.0,
                        "T_amb":-3
                }
        ]
}
```

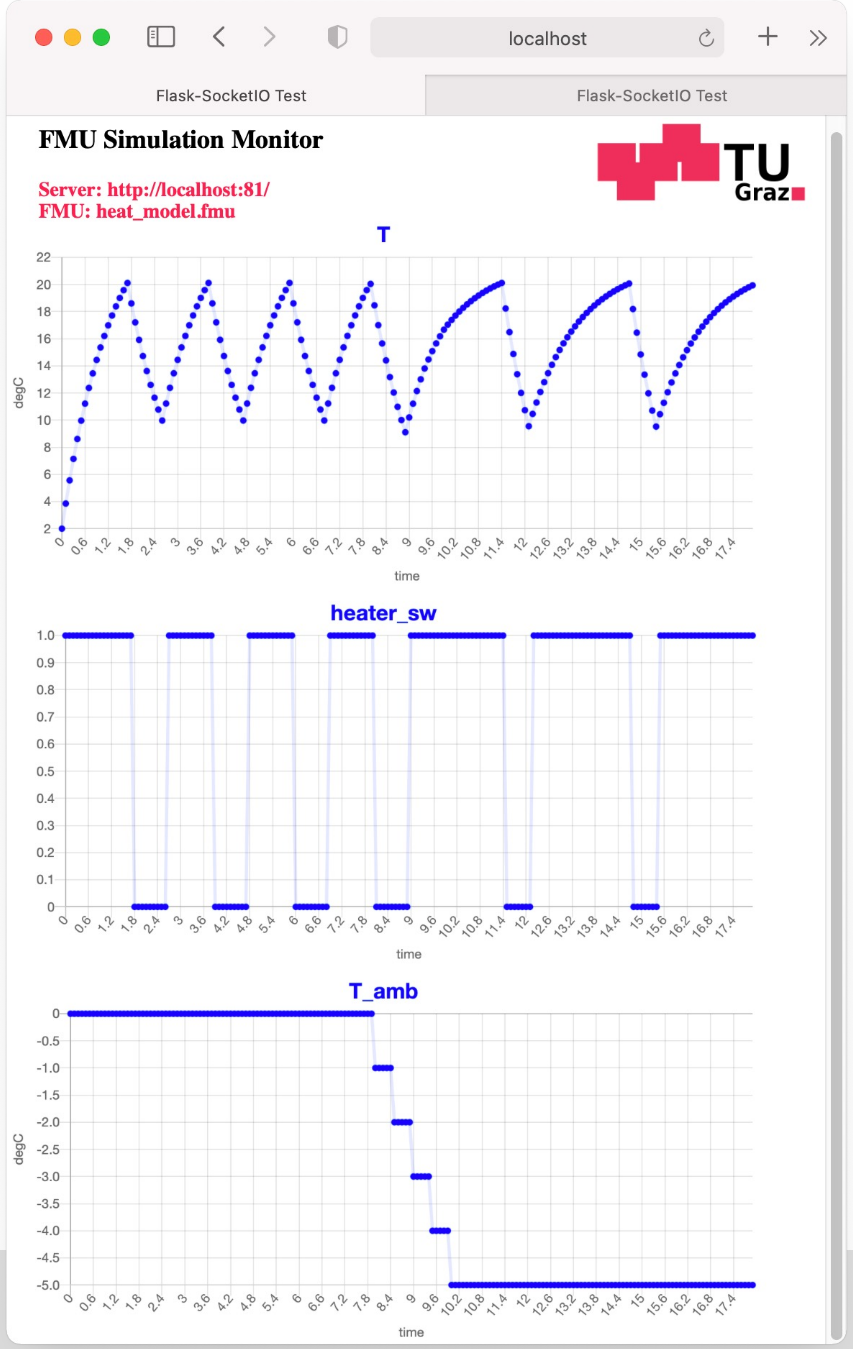
Use Case Heater Model

- Simple Heater Model
  - First order differential equation
  - Heater panel to heat up a medium
  - Surrounding environment with a global ambient temperature
  - Constant heat capacitance,
  - Constant heat transfer coefficient
  - Medium temperature is observed
- Temperature Control Unit
  - External placed 2-point controller

Heater FMU	heater_model.fmu
Heater CONFIG	config_heater_model.json
Heater Simulation Controller	heater_intern_control.py

Start Heater Model Docker Container

Name: model
Address: localhost:81



FMU Simulation Monitor

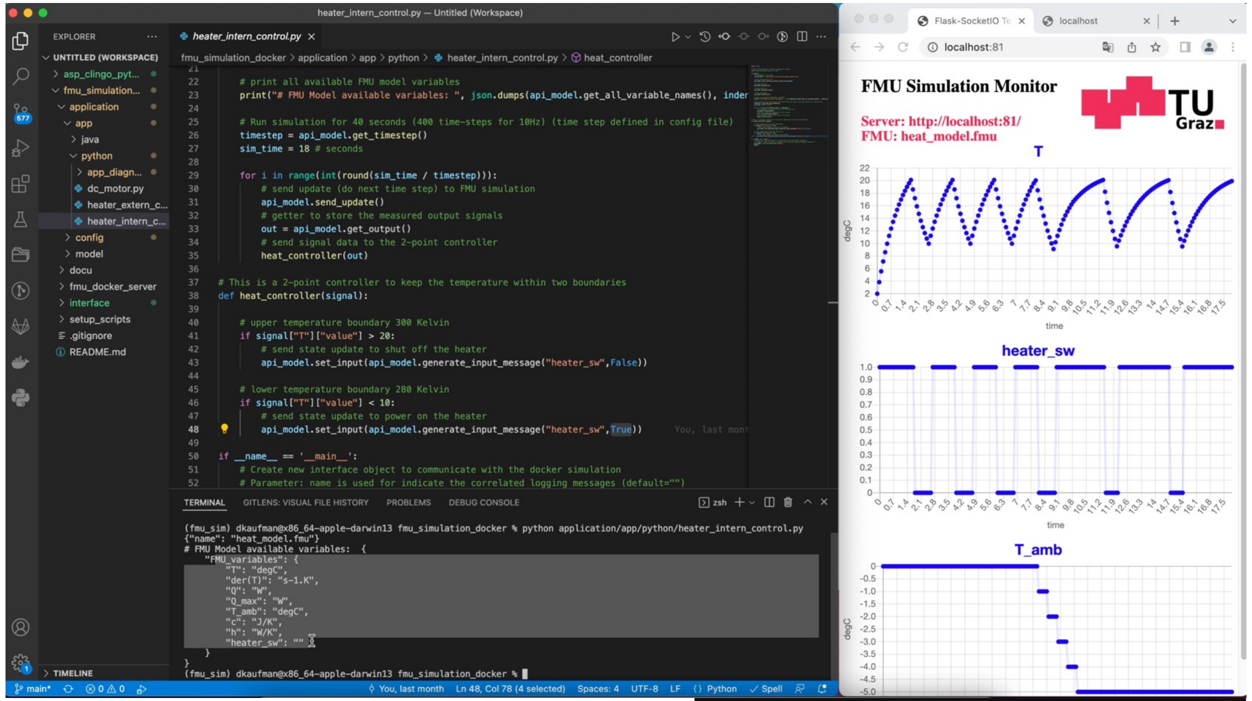
Server: http://localhost:81/
FMU: heat_model.fmu

Use Case Heater Model

1. Build Docker Image

2. Start Docker Container
   - Name
   - Address & port

3. Simulation Controller
   1. Initialize
      - Communication interface (server address)
      - Upload FMU model to server
      - Upload Configuration file
   2. Trigger simulation update by given timestep
   3. Get output signals and send to 2-point control unit
   4. Get 2-point control unit output and send as input signal to server



```python
    # print all available FMU model variables
    print("# FMU Model available variables: ", json.dumps(api_model.get_all_variable_names(), inder

    # Run simulation for 40 seconds (400 time-steps for 10Hz) (time step defined in config file)
    timestep = api_model.get_timestep()
    sim_time = 18 # seconds

    for i in range(int(round(sim_time / timestep))):
        # send update (do next time step) to FMU simulation
        api_model.send_update()
        # getter to store the measured output signals
        out = api_model.get_output()
        # send signal data to the 2-point controller
        heat_controller(out)

# This is a 2-point controller to keep the temperature within two boundaries
def heat_controller(signal):

    # upper temperature boundary 300 Kelvin
    if signal["T"]["value"] > 20:
        # send state update to shut off the heater
        api_model.set_input(api_model.generate_input_message("heater_sw",False))

    # lower temperature boundary 280 Kelvin
    if signal["T"]["value"] < 10:
        # send state update to power on the heater
        api_model.set_input(api_model.generate_input_message("heater_sw",True))      You, last mont

if __name__ == '__main__':
    # Create new interface object to communicate with the docker simulation
    # Parameter: name is used for indicate the correlated logging messages (default="")
```

```
(fmu_sim) dkaufman@x86_64-apple-darwin13 fmu_simulation_docker % python application/app/python/heater_intern_control.py
{"name": "heat_model.fmu"}
# FMU Model available variables:  {
    "FMU_variables": {
        "T": "degC",
        "der(T)": "s-1.K",
        "Q": "W",
        "Q_max": "W",
        "T_amb": "degC",
        "c": "J/K",
        "h": "W/K",
        "heater_sw": ""
    }
}
(fmu_sim) dkaufman@x86_64-apple-darwin13 fmu_simulation_docker %
```

FMU Simulation Monitor

Server: http://localhost:81/
FMU: heat_model.fmu

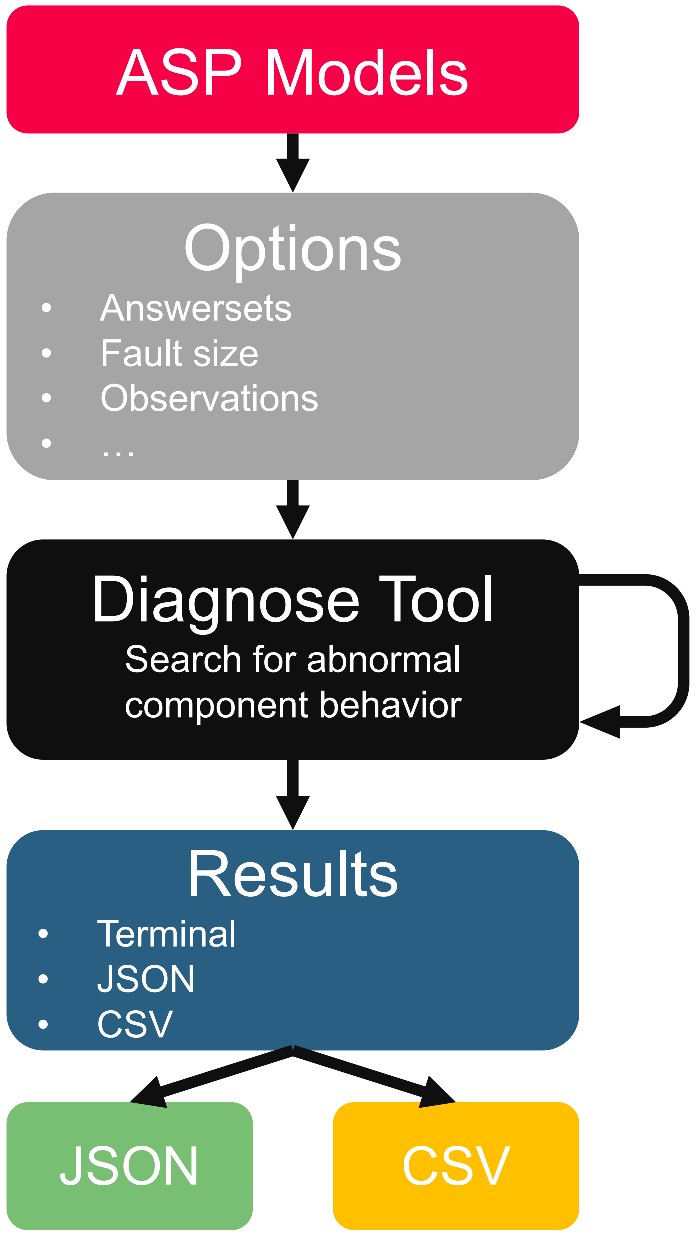
LIVE DEMO

ASP Diagnose Tool

- Clone Gitlab Repository:
  - git clone https://ss22_mts:-ZbnH7jrYe3JzrT1dod9@git.ist.tugraz.at/MBR/AI4DI/asp_clingo_python_tool.git
- Answer Set Programming (ASP) model diagnose tool
- Theorem solver: **Clingo 5.4.1**
- Modeling cyber physical systems in ASP
- Searching for abnormal behavior of components

- **Setup options:**
  - ASP File or directory to perform diagnose
  - Number of required answersets
  - Definition of fault size (e.g.: 2 -> 0 (as reference), 1, 2 searched)
  - Automatic constraint adding after each diagnose run
  - Additional observation file

- **Result options:**
  - **Terminal**
    - file, fault size, observations, diagnose, time for diagnose
  - **JSON**
    - index, file, fault size, observations, diagnose, time for diagnose
  - **CSV**
    - index, file, time for diagnose, number of diagnosis found, total time



ASP Models

Options
- Answersets
- Fault size
- Observations
- …

Diagnose Tool
Search for abnormal component behavior

Results
- Terminal
- JSON
- CSV

JSON

CSV

SCIENCE
PASSION
TECHNOLOGY

Thank you for your attention!

David Kaufmann, MSc

04/04/2022

David.Kaufmann@ist.tugraz.at