The International Workshop on Edge Artificial Intelligence for Industrial Applications (EAI4IA)

ANDANTE

AI
ECSEL JU

AI4DI

TEMPO

AI

Vienna, Austria

25-26 July 2022

The International Workshop on Edge Artificial Intelligence for Industrial Applications (EAI4IA)

AI
ECSEL JU
ANDANTE
AI4DI
TEMPO

A framework for integrating automated diagnosis into simulation

David Kaufmann, MSc TU Graz, Austria
Prof. DI Dr. Franz Wotawa TU Graz, Austria

Vienna, Austria 25-26 July 2022

Presentation Outline

- Introduction
- Framework Design and Method Implementation
- Use Case
  - CPS Simulation Model
  - ASP Diagnose Model
  - Demonstration
- Conclusions

Introduction

- How to prevent a systems from breaking?
  - Regular maintenance
  - Lifetime predictions
  - **Diagnosis** during runtime

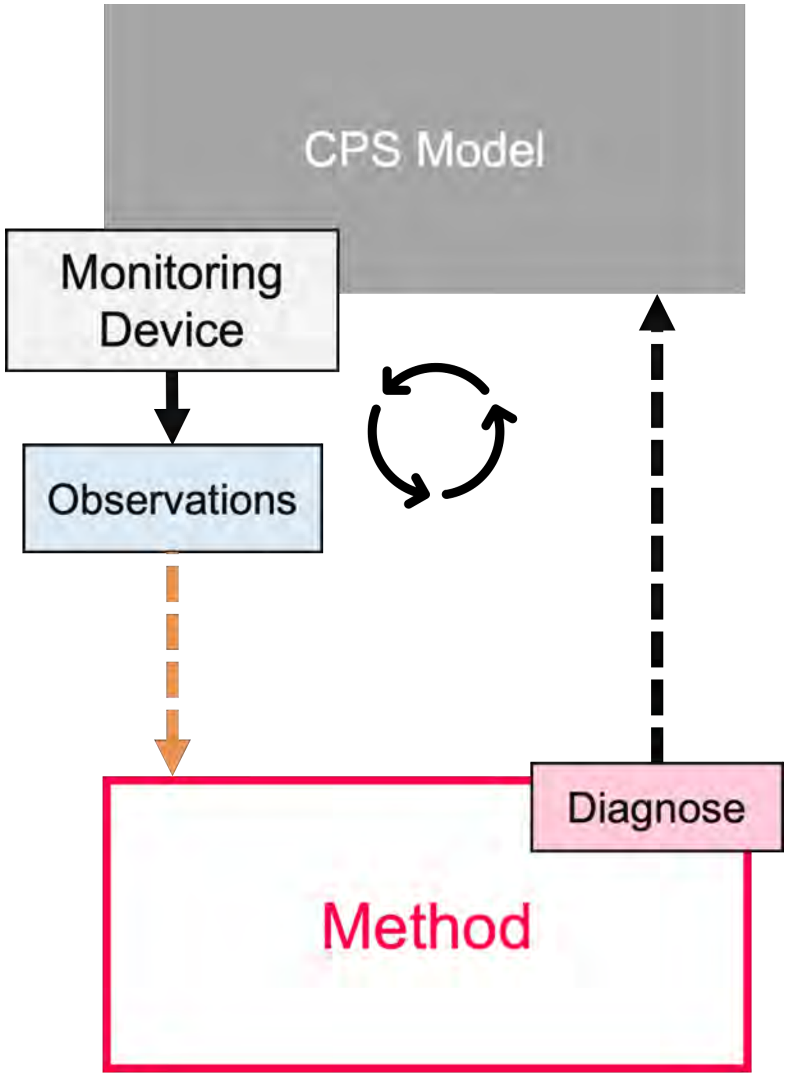
- Why performing advanced diagnosis on cyber-physical-systems?
  - Detection of failures during runtime
  - Localization of root cause
  - Initiating repair actions
  - Keep the system operational under safe conditions

Introduction

- Requirements for **testing** and **validation** of an advanced diagnosis method:
  - Co-simulation environment framework
  - Standardized simulation environment
  - Step-by-step simulation
  - Fault injection during runtime
  - Interface for information flow to and from the method under test
  - Different programming environments

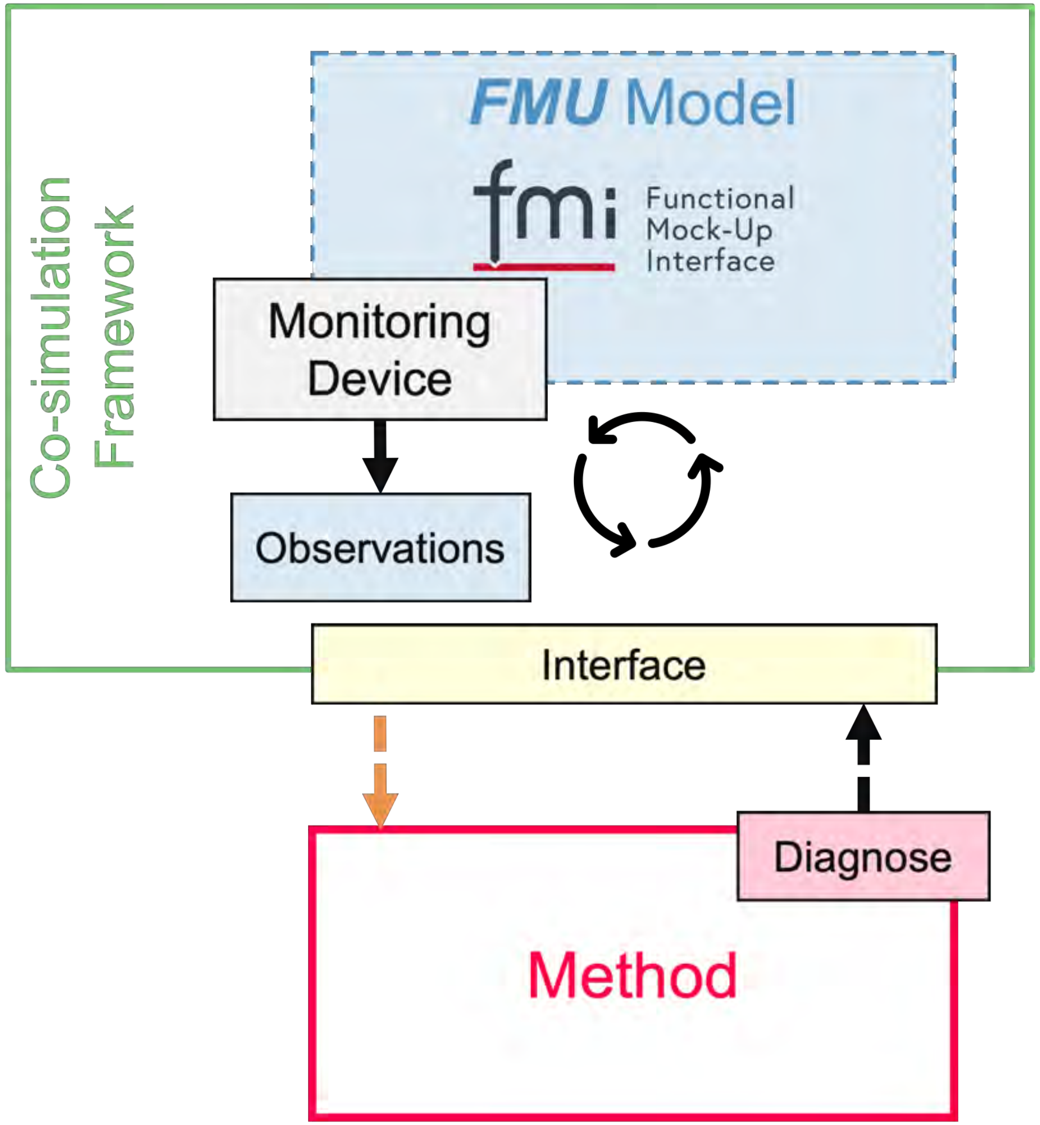
Framework Design and Method Implementation

- Cyber-Physical-System (CPS) model
- Monitoring for observations
- Method under test, e.g.:
  - Model-based diagnosis
  - Simulation-based diagnosis
  - ML-based diagnosis
- Fault detection & localization of root cause
- Diagnose feedback for failure mitigation



Framework Design and Method Implementation

- Co-simulation framework
- Use standardized format FMU as models
    - Solver integrated in co-simulation FMU
    - Perform step-by-step simulation
- Simple interface for linking tools
- Different programming environments
- Test & validation environment for tools



Framework Design and Method Implementation

Framework:

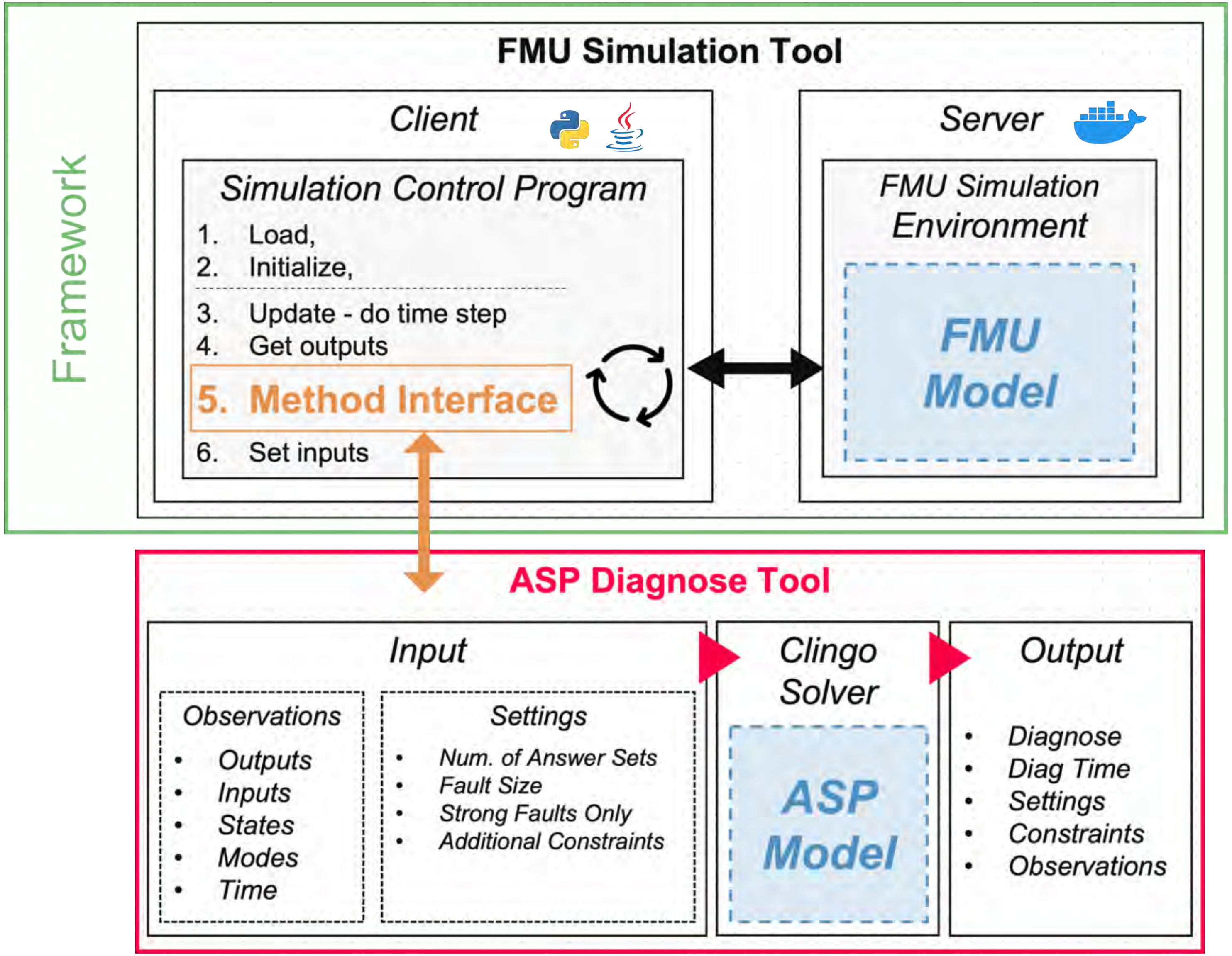
- FMU Simulation Tool
- Client-Server solution
- Dockerized simulation environment
- Multiple models in co-simulation
- PyFMI library
- REST API communication

Method:

- ASP Diagnose Tool
- Theorem solver Clingo 5.4.1
- Input: observations & settings
- Output format: JSON, CSV

Models:

- FMU Model (generated from CPS)
- ASP Model (abstract model of CPS)



Framework Design and Method Implementation

Framework:

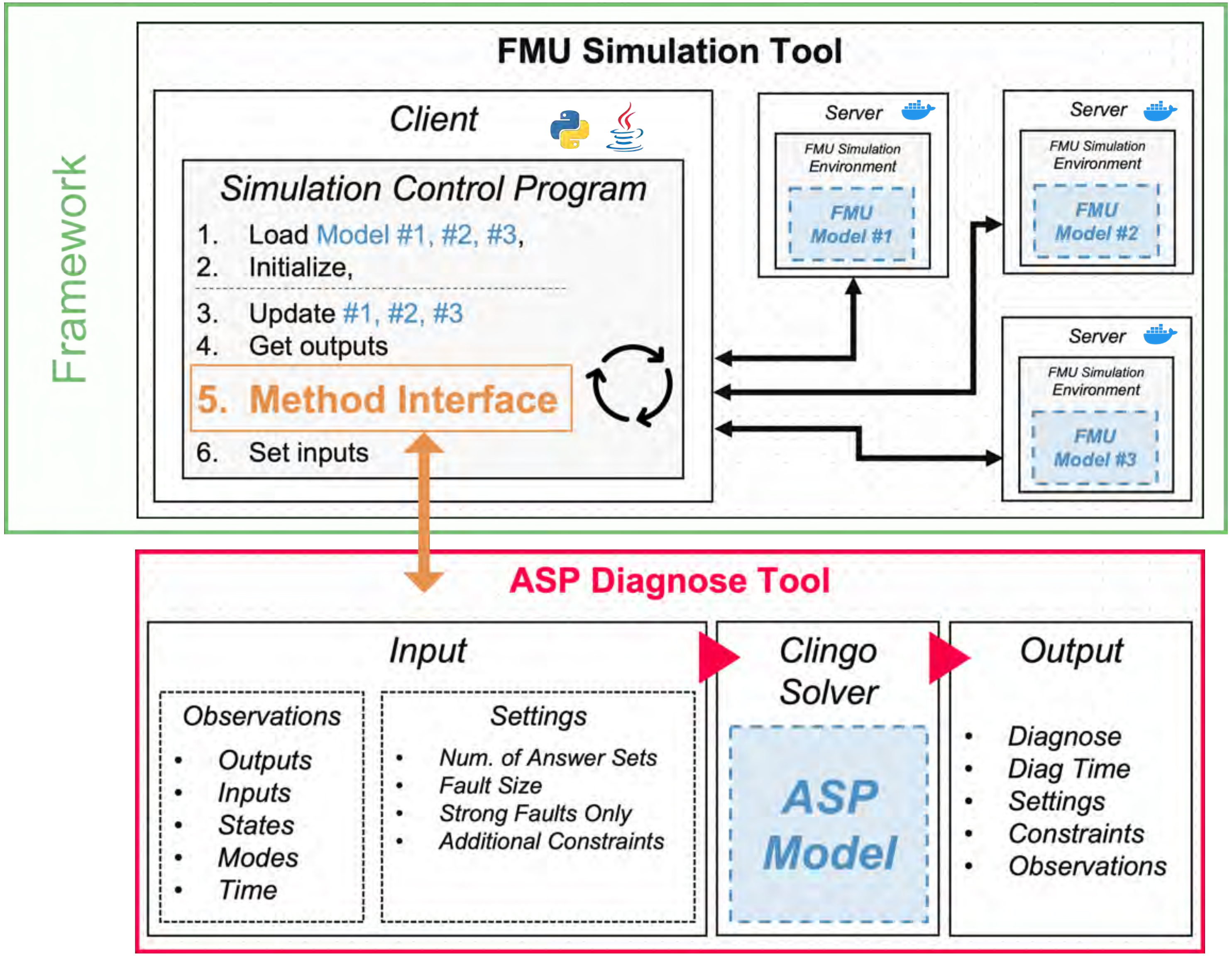
- FMU Simulation Tool
- Client-Server solution
- Dockerized simulation environment
- Multiple models in co-simulation
- PyFMI library
- REST API communication

Method:

- ASP Diagnose Tool
- Theorem solver Clingo 5.4.1
- Input: observations & settings
- Output format: JSON, CSV

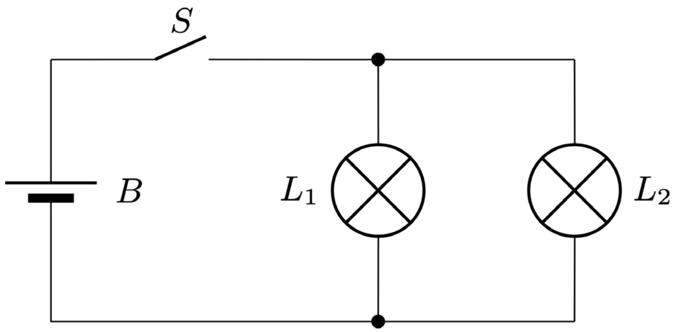
Models:

- FMU Model (generated from CPS)
- ASP Model (abstract model of CPS)



Use Case – CPS Simulation Model

- OpenModelica OMEdit
- Electrical circuit model
- Components:
  - Battery
  - Switch
  - Lamp 1 & 2
- Component fault types
- **FMU generation**



Component	*State*	*Description*
light bulb (bulb), switch (sw)	ok broken short	ordinary behaviour open connection in eletrical circuit short in the electrical circuit
battery (bat)	ok empty	ordinary behaviour empty battery fault

circuit components & connections

```
Two_Lamp_Circuit
PhysicalFaultModeling.PFM_Bulb bulb1(r = 100.0);
PhysicalFaultModeling.PFM_Bulb bulb2(r = 100.0);
PhysicalFaultModeling.PFM_Switch sw;
PhysicalFaultModeling.PFM_Ground gnd;
PhysicalFaultModeling.PFM_Battery bat(vn = 5.0);

connect(gnd.p, bat.m);
connect(bat.p, sw.p);
connect(sw.m, bulb1.p);
connect(sw.m, bulb2.p);
connect(bulb1.m, gnd.p);
connect(bulb2.m, gnd.p);
    Two_Lamp_Circuit;
```

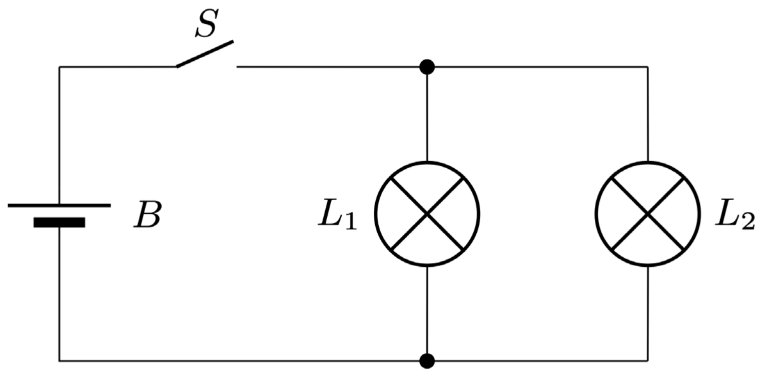
define components state input

```
Two_Lamp_Circuit_Testbench
PhysicalFaultModeling.Two_Lamp_Circuit sut;
input FaultType bat_state(start=FaultType.ok);
input OperationalMode switch_mode(start=OperationalMode.close);
input FaultType switch_state(start=FaultType.ok);
input FaultType bulb1_state(start=FaultType.ok);
input FaultType bulb2_state(start=FaultType.ok);

sut.sw.mode = switch_mode;
sut.bat.state = bat_state;
sut.sw.state = switch_state;
sut.bulb1.state = bulb1_state;
sut.bulb2.state = bulb2_state;
    Two_Lamp_Circuit_Testbench;
```

Use Case – ASP Diagnose Model

- Answer Set Programming
- Model behavior in first order logic representation
- Components:
  - Battery
  - Switch
  - Lamp 1 & 2
- Component connection



Lamp state (on, off) logic

```
val(light(X),on) :- type(X,lamp), val(in_pow(X),nominal), nab(X).
val(in_pow(X),nominal) :- type(X,lamp), val(light(X),on).
val(light(X),off) :- type(X,lamp), val(in_pow(X),zero), nab(X).
```

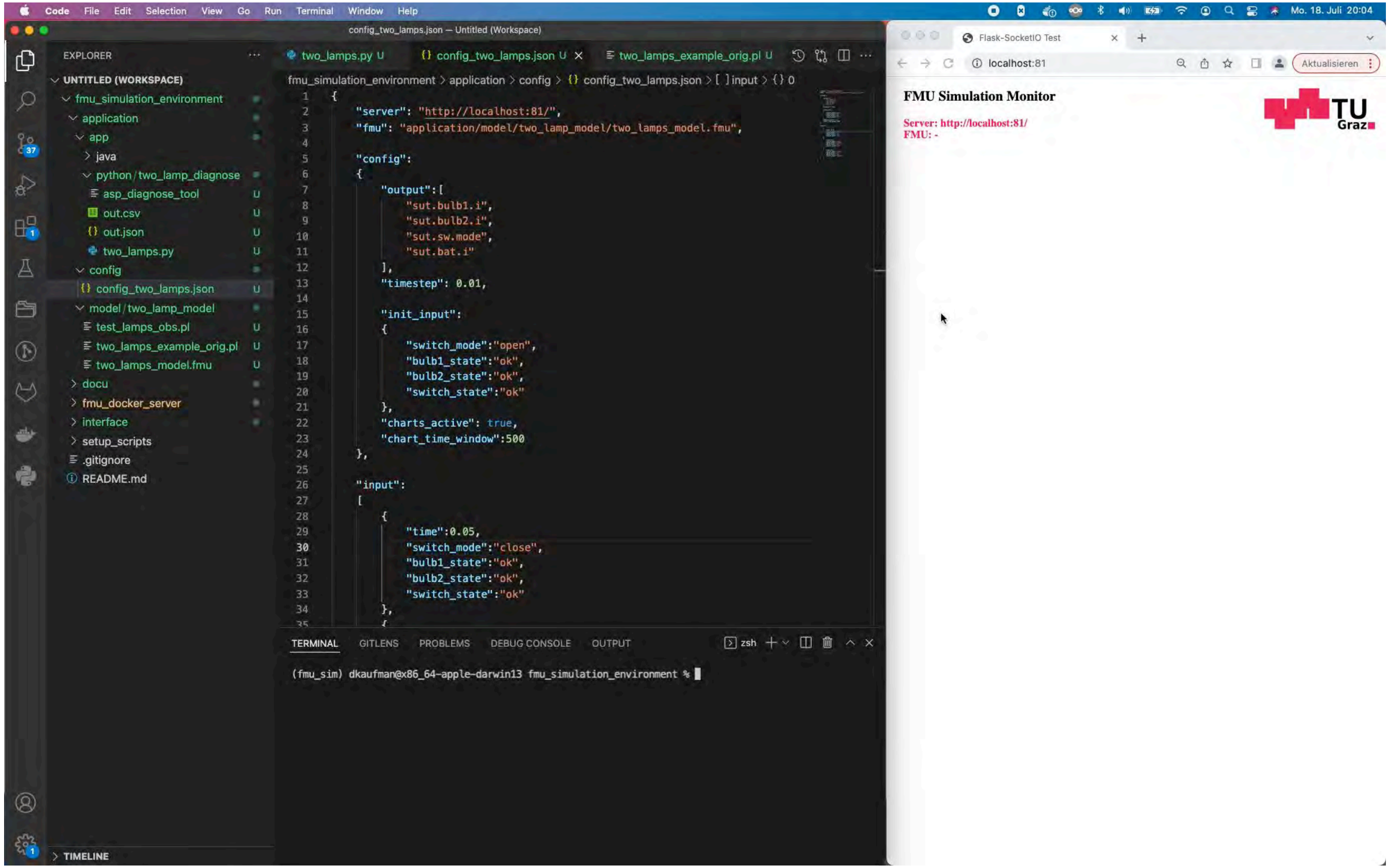
Component connection logic

```
val(X,V) :- conn(X,Y), val(Y,V).
val(Y,V) :- conn(X,Y), val(X,V).
:- val(X,V), val(X,W), not V=W.

type(b, bat).
type(s, sw).
type(l1, lamp).
type(l2, lamp).

conn(in_pow(s), pow(b)).
conn(out_pow(s), in_pow(l1)).
conn(out_pow(s), in_pow(l2)).
```

Use Case - Demonstration

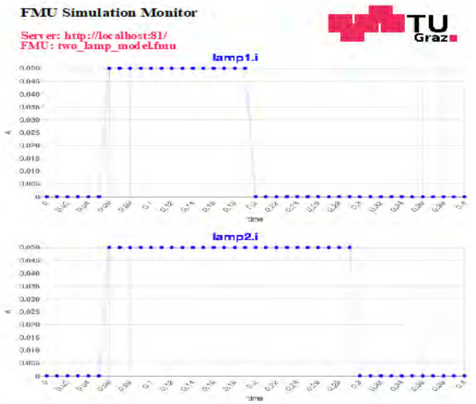


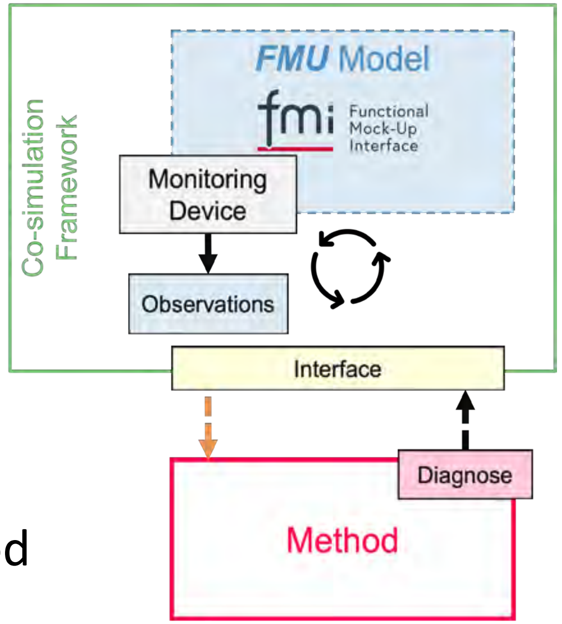
Conclusion

- Co-simulation framework for FMU models
- Simulations are safe and cost efficient
- Example method: ASP Diagnose Tool
- Usability tested by students:
  - Simple setup
  - Fast and productive tool to gather results
  - Use Cases: Low-Pass-Filter, H-Bridge, Heater-Panel, Javelin-Throw, etc.

Projection in the future:

- Test and validate diagnosis methods for AI systems
- Test planning tools for mitigating
- Simulation framework interface allows to add any method
- Diagnosis is part of self adaptive system with growing interest in autonomous driving
- Distribute co-simulation framework to the community





Time		0.3 sec
Input	switch_mode	close
	bulb1_state	ok
	bulb2_state	ok
	switch_state	broken
Output	sut.bulb1.i	0.00 A
	sut.bulb2.i	0.00 A
	sut.bat.i	0.00 A

Time		0.3 sec
Status	Model status	model satisfied
	Diagnose time	0.003007 sec
	Fault size	1
	Diagnose found	2
	Diagnose	ab(s) or ab(b)
	Diagnose time	0.002818 sec
	Fault size	2
	Diagnose found	1
	Diagnose	ab(l1) & ab(l2)
	Observation	val(light(l1), off). val(light(l2), off). on(s).

Thank You

For your attention

@ David.Kaufmann@ist.tugraz.at

Event Organisers

The Key Digital Technologies Joint Undertaking - the Public-Private Partnership for research, development and innovation – funds projects for assuring world-class expertise in these key enabling technologies, essential for Europe's competitive leadership in the era of the digital economy. KDT JU is the successor to the ECSEL JU programme. www.kdt-ju.europa.eu

The AI4DI project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826060. The JU receives support from the European Union’s Horizon 2020 research and innovation programme and the national authorities. www.ai4di.eu

The TEMPO project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826655. The JU receives support from the European Union’s Horizon 2020 research and innovation programme and Belgium, France, Germany, The Netherlands, Switzerland. www.tempo-ecsel.eu

The ANDANTE project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 876925. The JU receives support from the European Union’s Horizon 2020 research and innovation programme and Belgium, France, Germany, The Netherlands, Portugal, Spain, Switzerland. www.andante-ai.eu