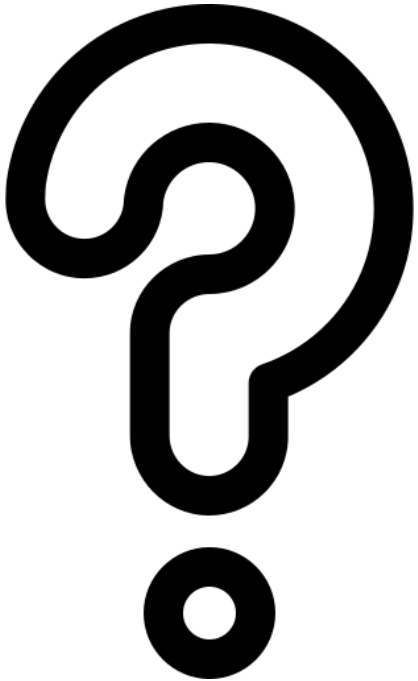
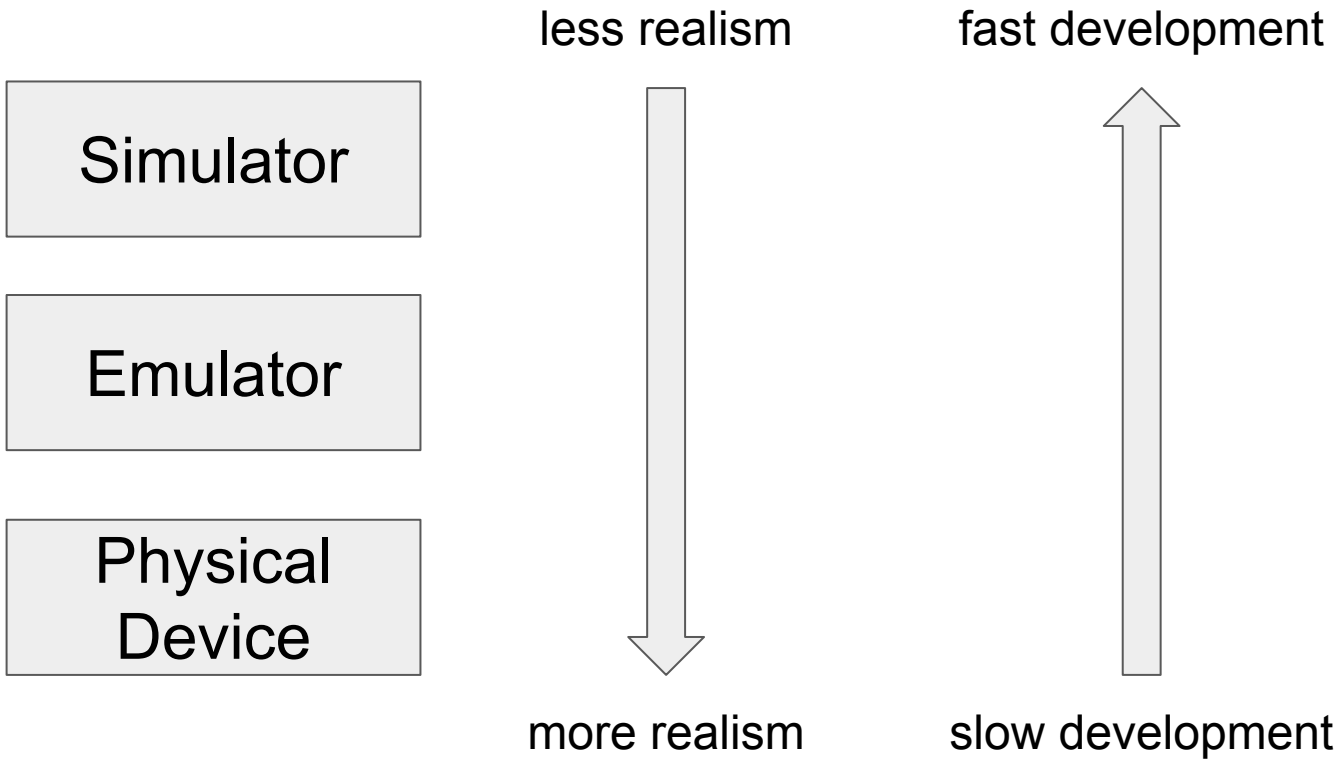
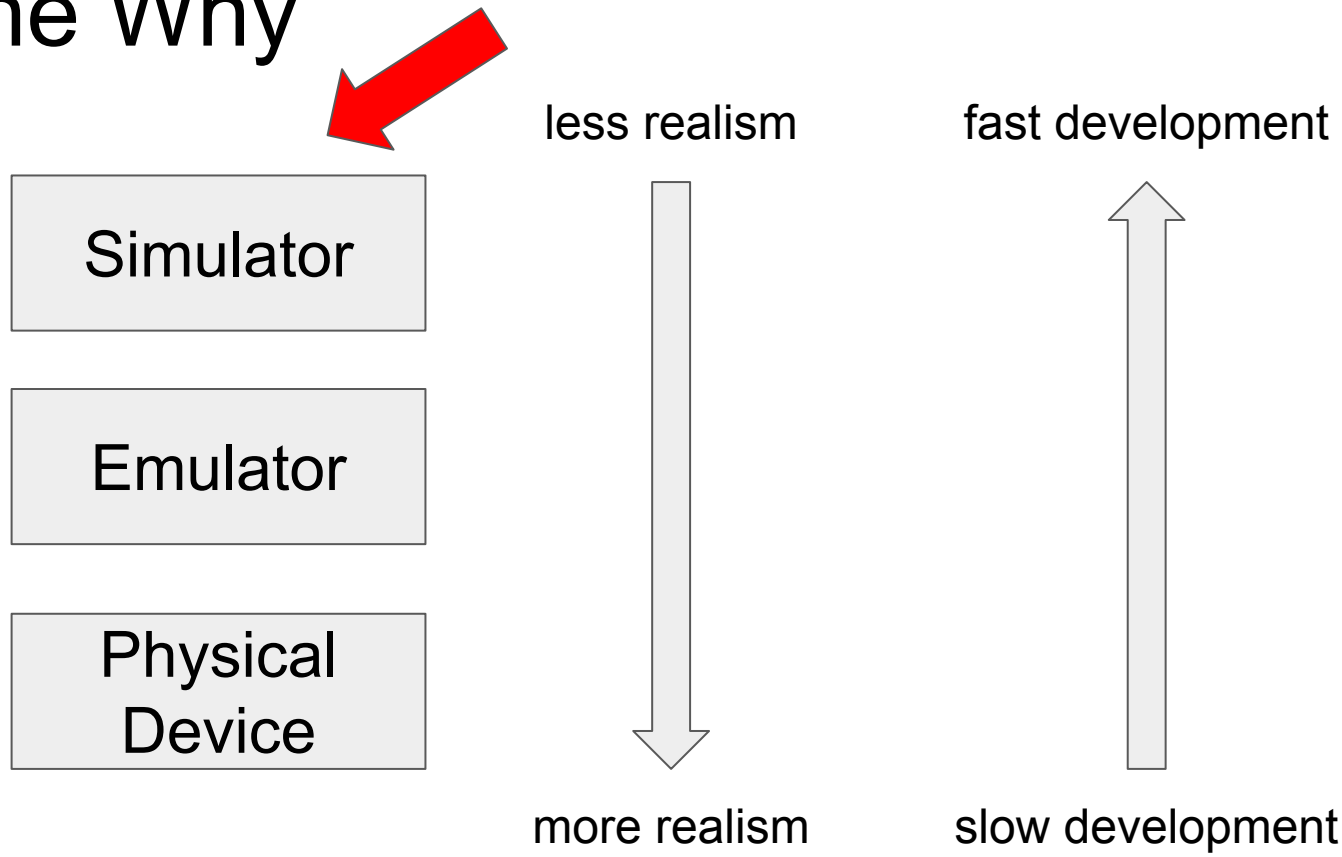
Another paper, but why



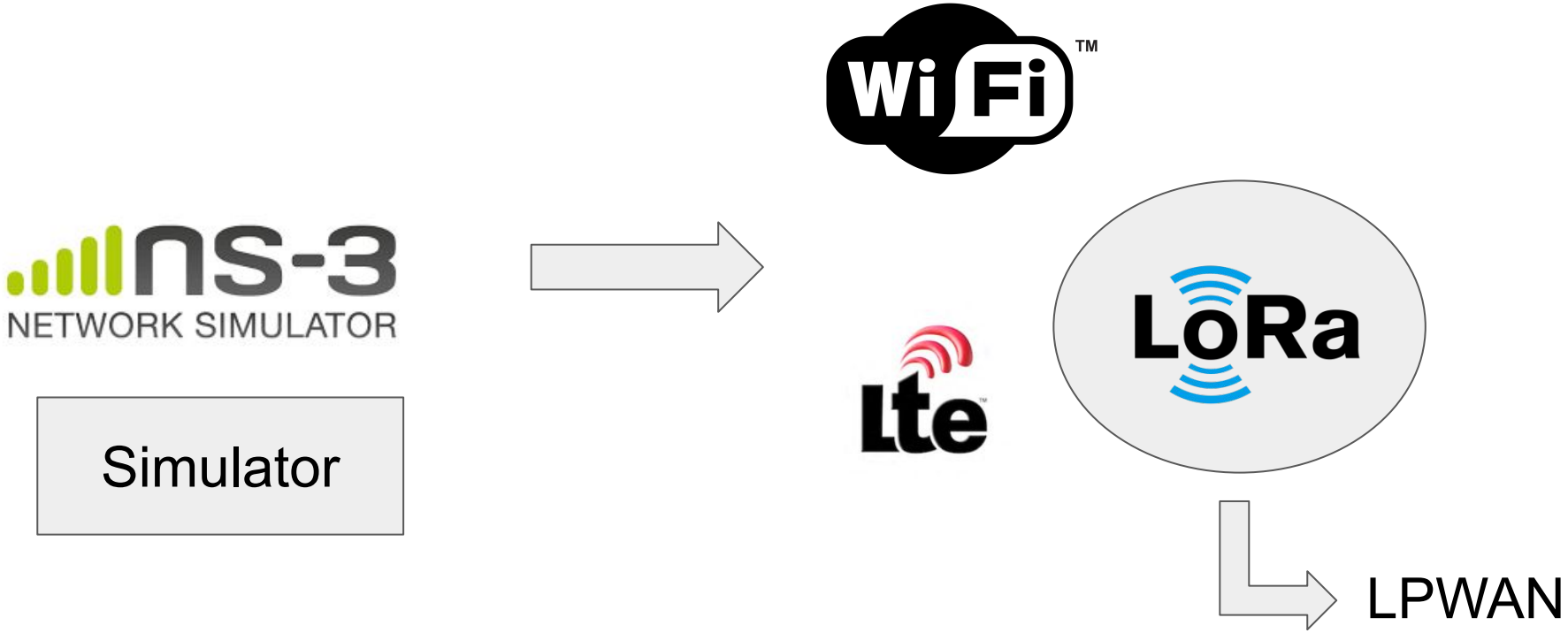
The Why: introduction to network simulation



The Why


Simulator
Emulator
Physical Device
less realism
more realism
fast development
slow development

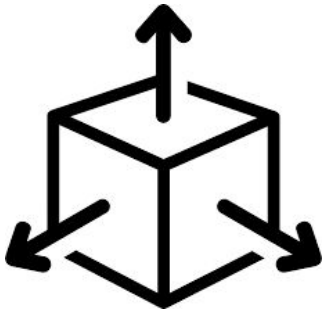
The Why: Wireless technologies



The Why: LPWAN Factors

Battery Lifespan

Scalability



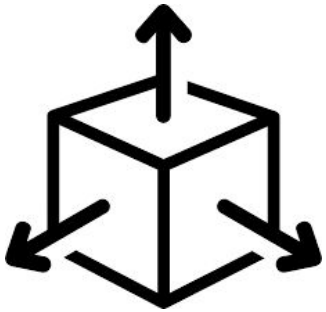
Throughput

Coverage

The Why: LPWAN Features

Battery Lifespan

Scalability



Throughput

Coverage

The Why: Modeling

Battery Lifespan

LPWAN node

ns-3 NETWORK SIMULATOR

Energy Model

getEnergyFraction

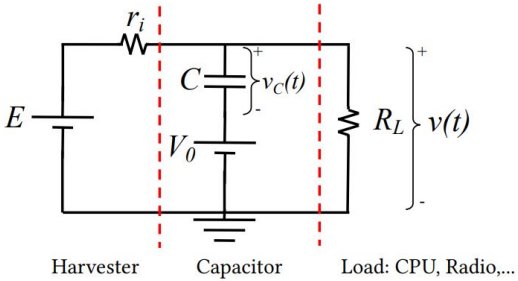
getSupplyVotlage

HandleEnergyDrainedEvent

The Why: Current Workflow

Energy Model

1. Draw Circuit (paper)



2. Solve Circuit

$$v(t,s) = E\frac{R_{eq}(s,t)}{r_i(t)}\left(1 - e^{-\frac{t}{R_{eq}(s,t)C}}\right)$$

3. Write Code

```
double E0 = m_eFull + K + m_internalResistance * m_typCurrent - A;

double E = E0 - K * m_qRated / (m_qRated - it) + A * std::exp(-B * it);
```

The Why: Current Workflow

```
double E0 = m_eFull + K + m_internalResistance * m_typCurrent - A;

double E = E0 - K * m_qRated / (m_qRated - it) + A * std::exp(-B * it);
```

Manual work

Error prone

Time consuming

Too complex

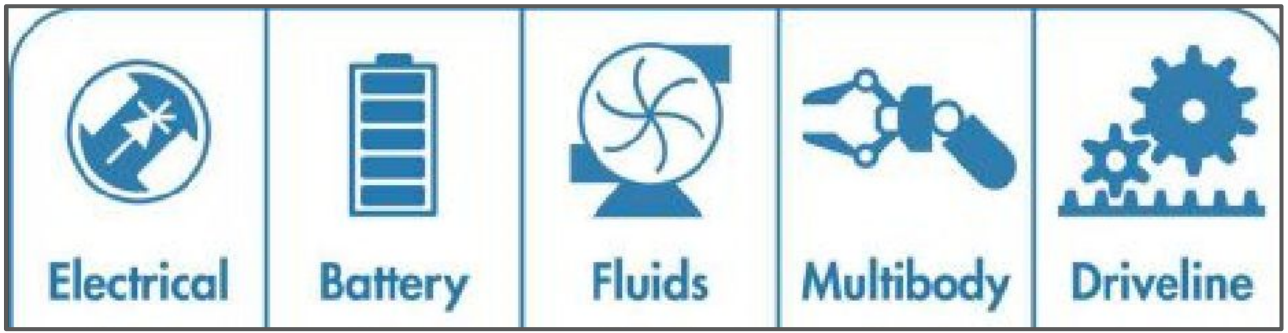
No abstractions

Cost > Benefit

The Why: Physical Modeling

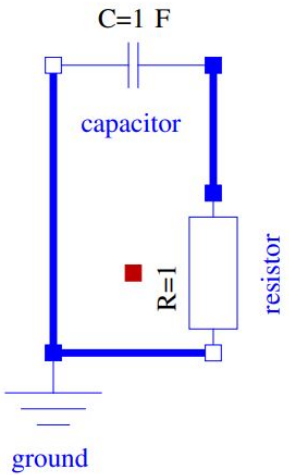
OpenModelica

MathWorks®
Simscape



The Why: New Workflow

1. Draw Circuit (GUI)



Magic

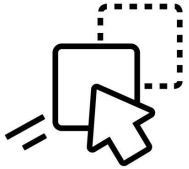
More Magic

Energy Model

The Why: New Workflow

OpenModelica

MathWorks
Simscape



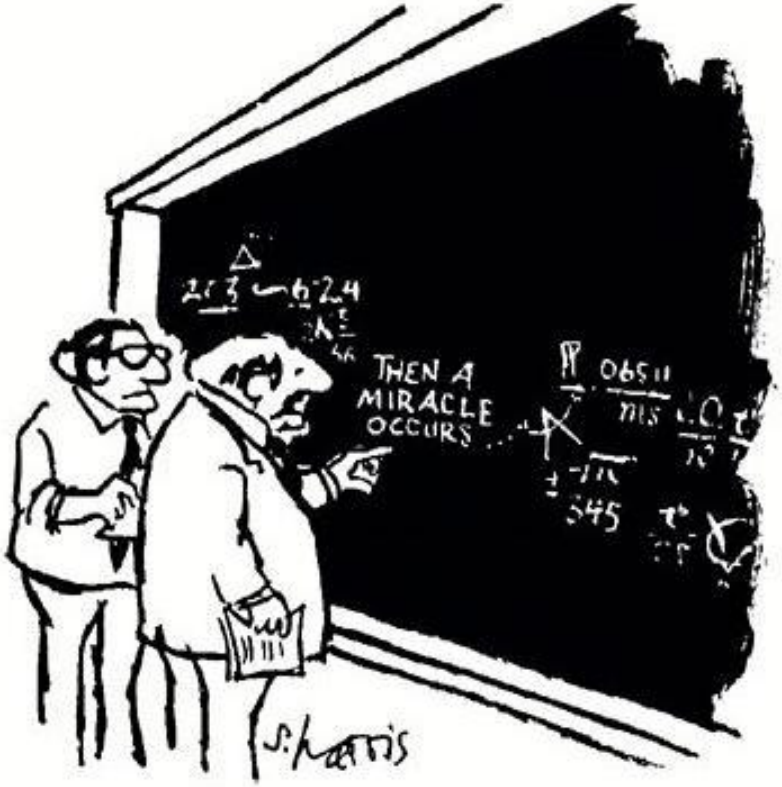
Drag and Drop

GUI

Predefined building blocks

No circuit analysis

The How



"I THINK YOU SHOULD BE MORE EXPLICIT HERE IN STEP TWO."

The How

OpenModelica

ns-3 NETWORK SIMULATOR

MathWorks®
Simscape

 Hybrid Co-Simulation Problem

The How

OpenModelica

ns-3
NETWORK SIMULATOR

Hybrid Co-Simulation Problem

My Paper

Extending Energy Models for Wireless Network Simulation with FMI-based Hybrid Co-Simulation

Lars Moons
Department of Computer Science
University of Antwerp
lars.moons@student.uantwerpen.be

Related Work

Hybrid co-simulation: it's about time

Fabio Cremona[1] · Marten Lohstroh[1] · David Broman[2] · Edward A. Lee[1] · Michael Masin[3] · Stavros Tripakis[1,4]

Step Revision in Hybrid Co-simulation with FMI

Fabio Cremona*‡¶, Marten Lohstroh*, David Broman†, Marco Di Natale‡,
Edward A. Lee*, and Stavros Tripakis*§
{f.cremona,marten,eal,stavros}@eecs.berkeley.edu, dbro@kth.se, marco@sssup.it
*University of California Berkeley, USA †KTH Royal Institute of Technology, Sweden
‡Scuola Superiore Sant'Anna, Italy §Aalto University, Finland ¶ALES — United Technologies Research Center, Italy

Nothing about Energy Models

FMI-based Co-Simulation of Hybrid Closed-loop Control System Models

Edmund Widl, Florian Judex, Katharina Eder
Austrian Institute of Technology, Vienna, Austria
{edmund.widl, florian.judex, katharina.eder}@ait.ac.at

Peter Palensky
Delft University of Technology, Delft, The Netherlands
p.palensky@tudelft.nl

Topics

1. Discrete Event Simulation

2. Physical Simulation

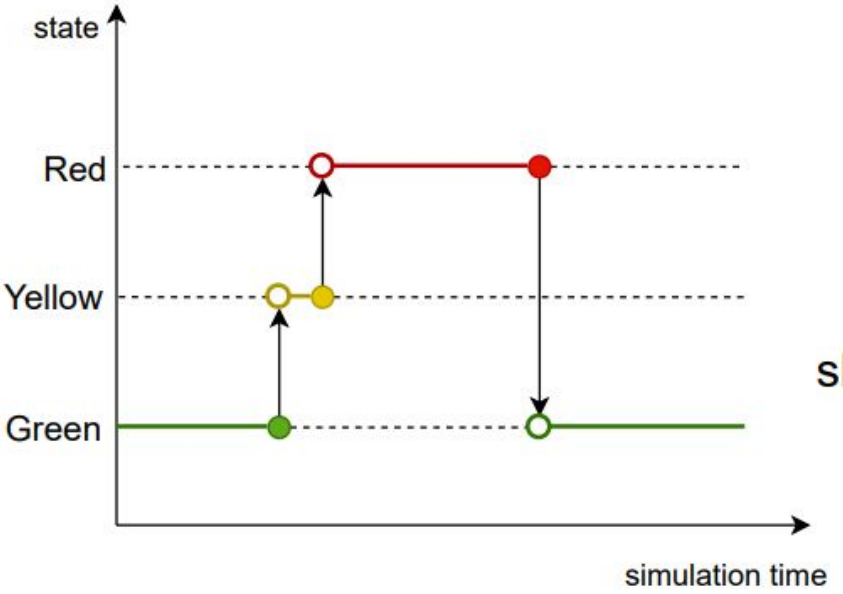
3. FMI standard

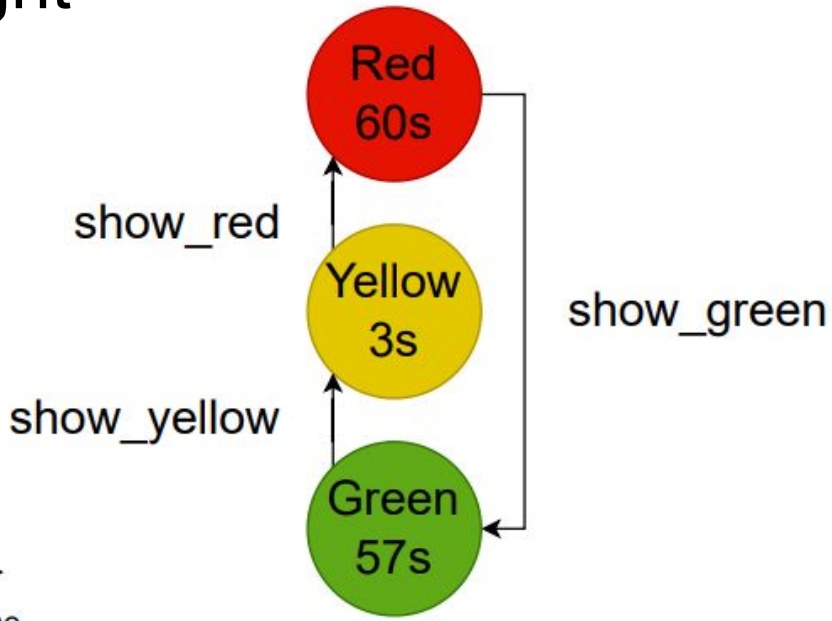
4. Architecture

5. PoC Implementation

Discrete Event Simulation

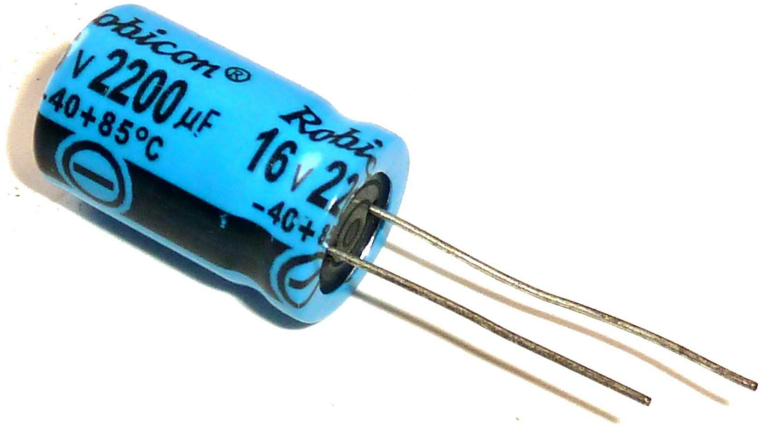
Traffic Light

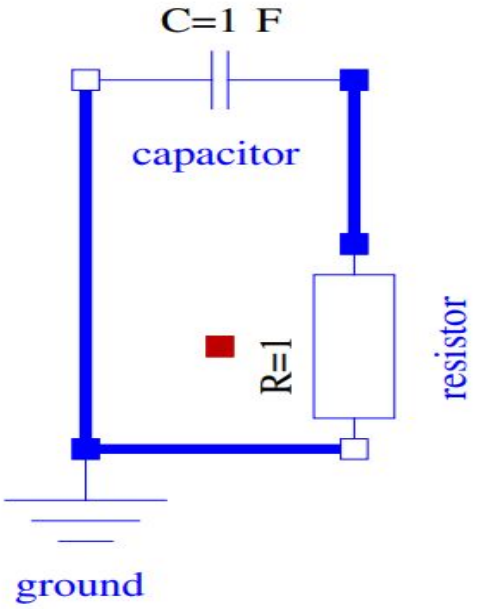




Physical Simulation OpenModelica

Discharging Capacitor

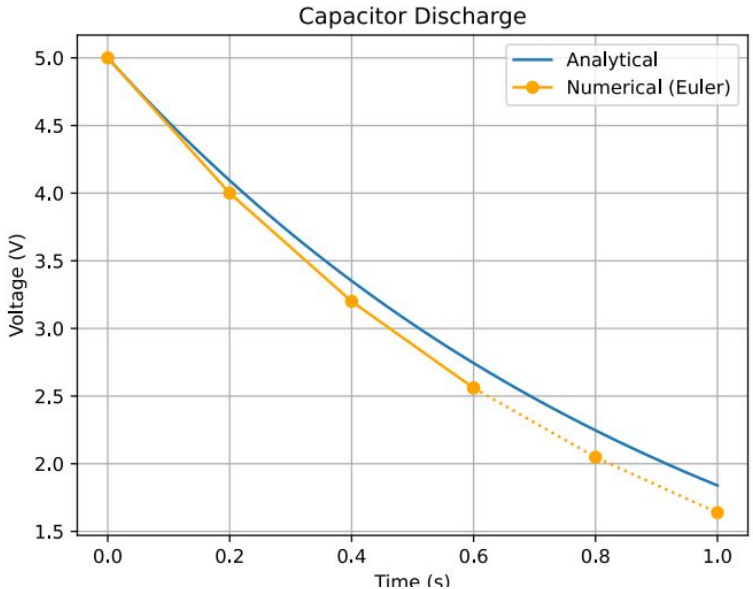




Physical Simulation OpenModelica

Discharging Capacitor

```
model CapacitorDischarge
  parameter Real C = 1;
  parameter Real Resistance = 1;
  parameter Real initialVoltage = 5;
  Real voltage(start=initialVoltage);
equation
  der(voltage) = -voltage / (C * Resistance);
end CapacitorDischarge;
```

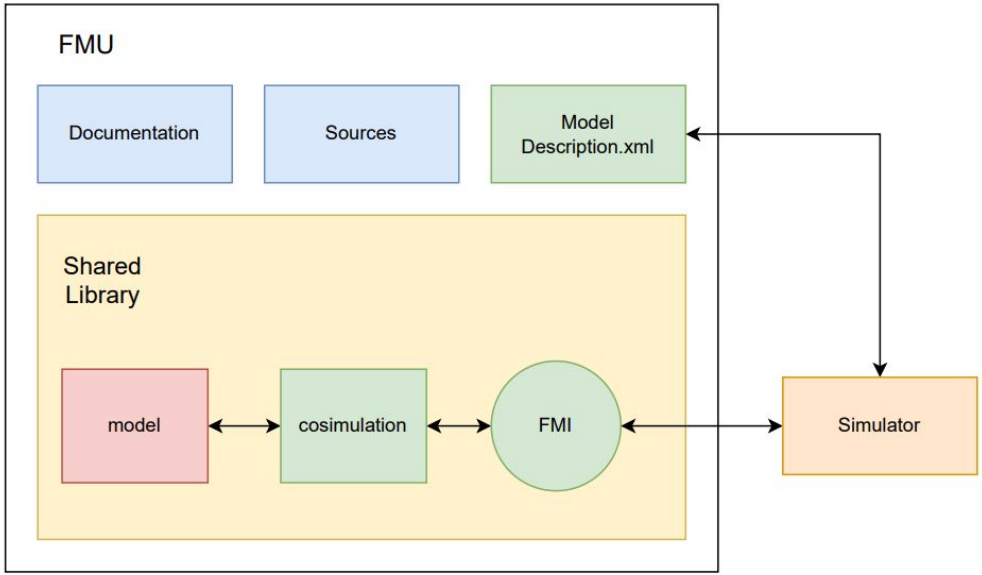


FMI Standard

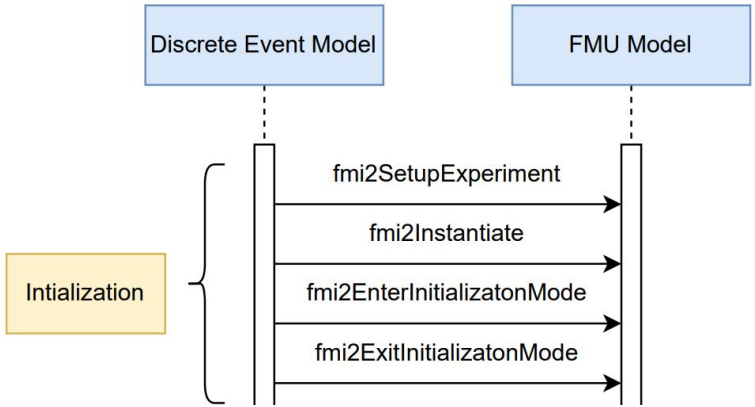
ZIP

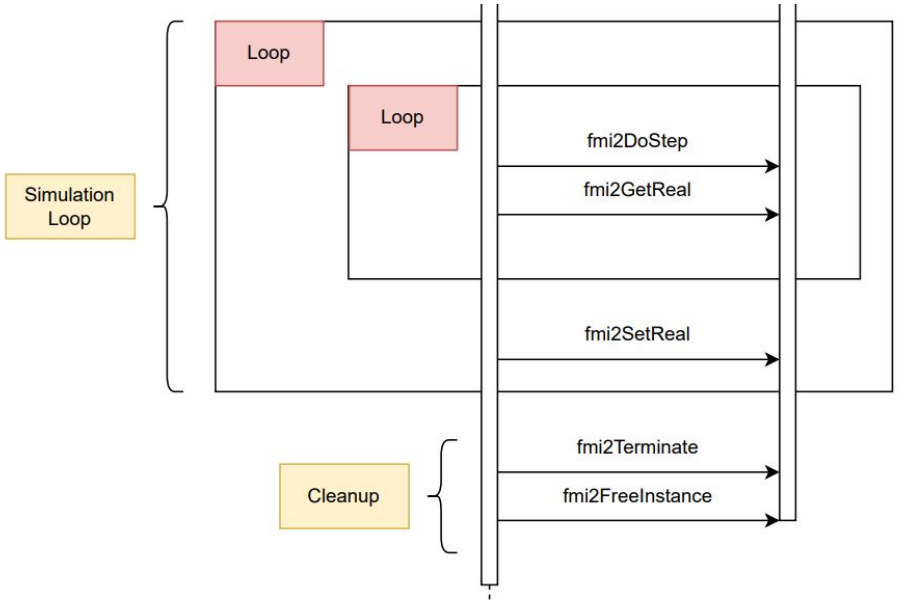
Co-Simulation

Model-Exchange

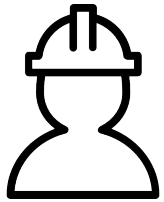


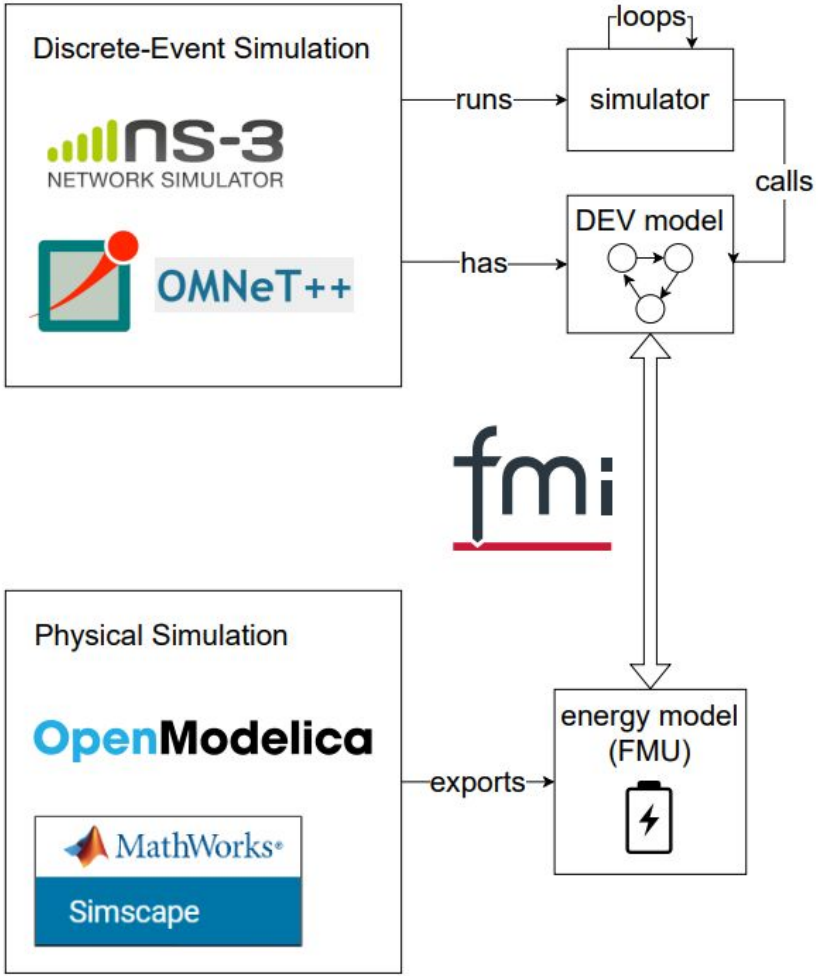
FMI Standard





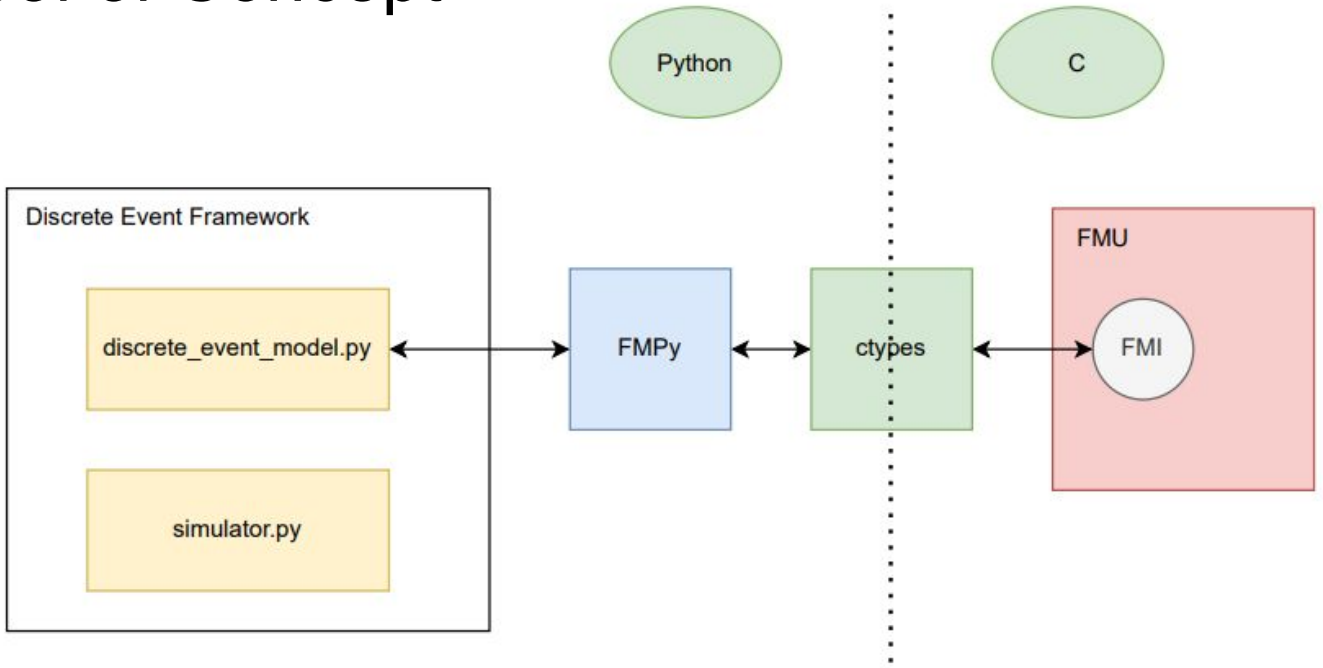
Architecture



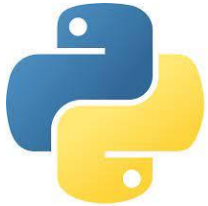


PoC Implementation

Proof of Concept



PoC Implementation



discrete_event_model.py

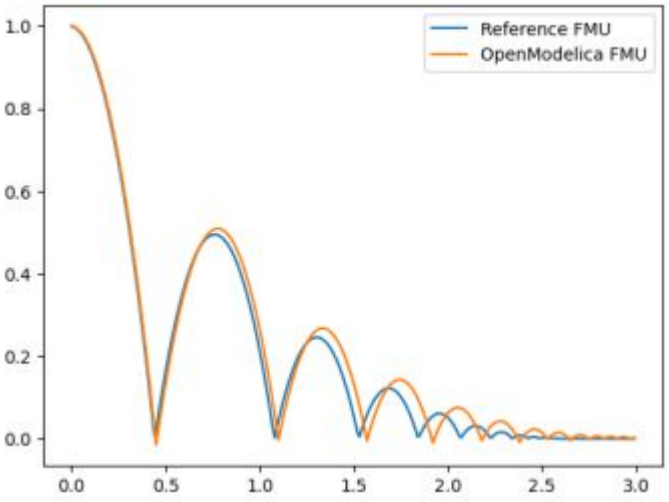
```
class BouncingBall:
    """
    https://github.com/CATIA-Systems/FMPy/t
    """
    def __init__(self, fmu_path):
        self.fmu: FMU2Slave | None = None
        self.fmu_dir = None
```

simulator.py

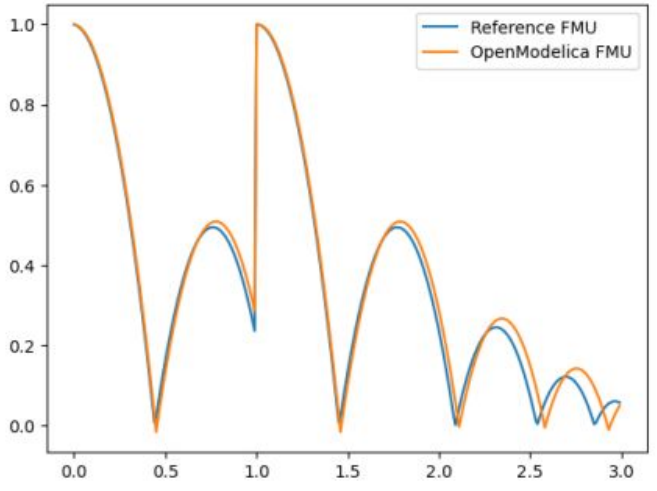
```
class Simulator:
    def __init__(self):
        self.events: list[Event] = []
        self.time = timedelta()
        self.stop_time: timedelta | None = None
```

PoC Implementation

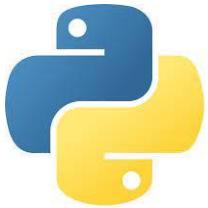
Bouncing Ball

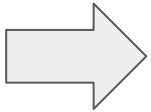


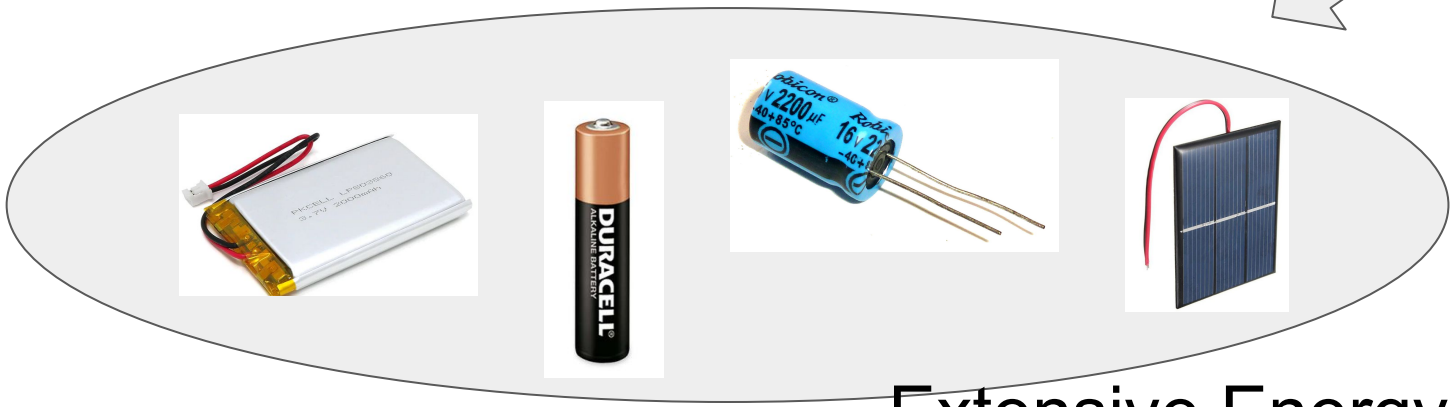
Reset Event



Future Work







Extensive Energy Framework