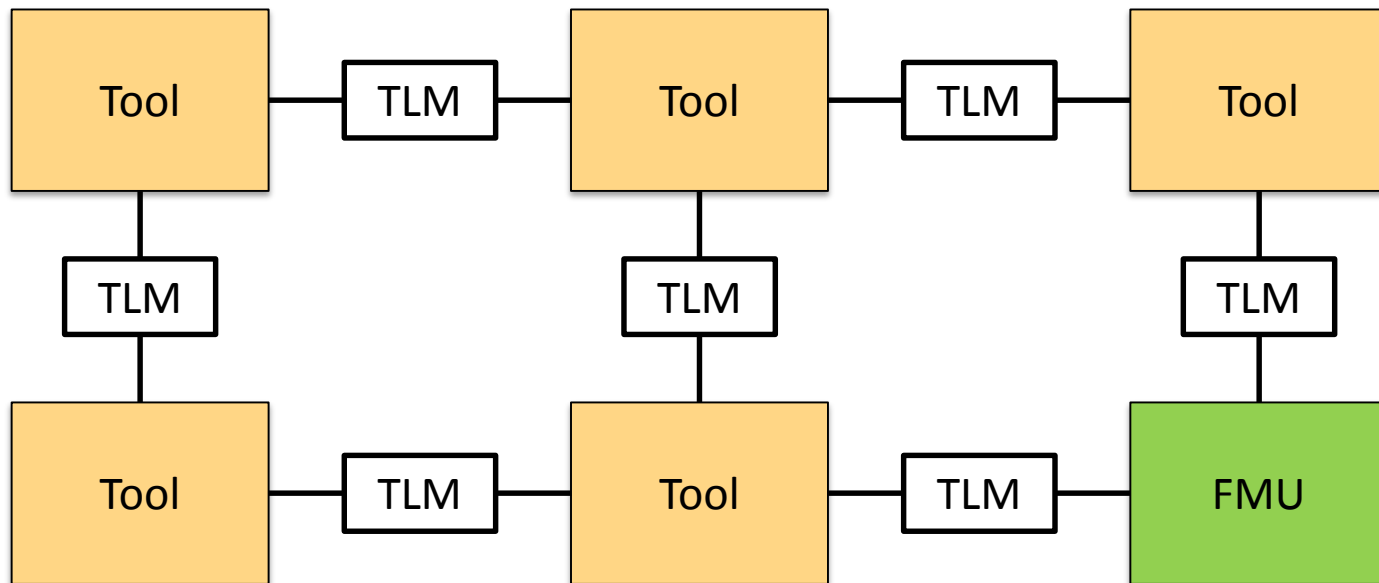# FMI for Asynchronous TLM-based Co-simulation

Robert Braun
Fluid & Mechatronic Systems
Linköping University

Introduction

What existed before:
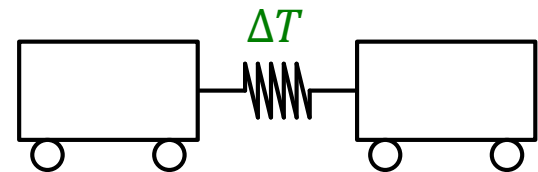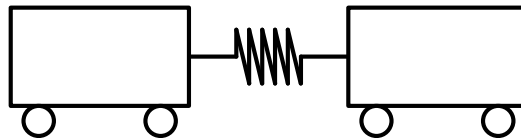

What we want!

LINKÖPING UNIVERSITY

Transmission Line Modelling (TLM)

No TLM:

TLM:

Real system:

Model:

Decoupled model:

ΔT

ΔT

ΔT

ΔT
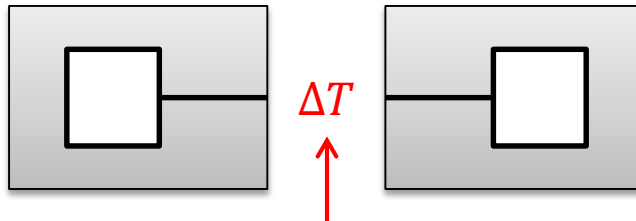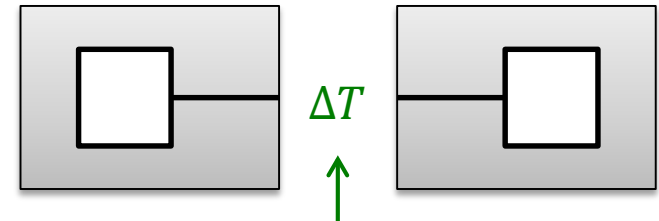
ΔT

Bad delay!

Good delay!

Tool Decoupling

State-of-the-art:

- Numerical time delay → numerical error
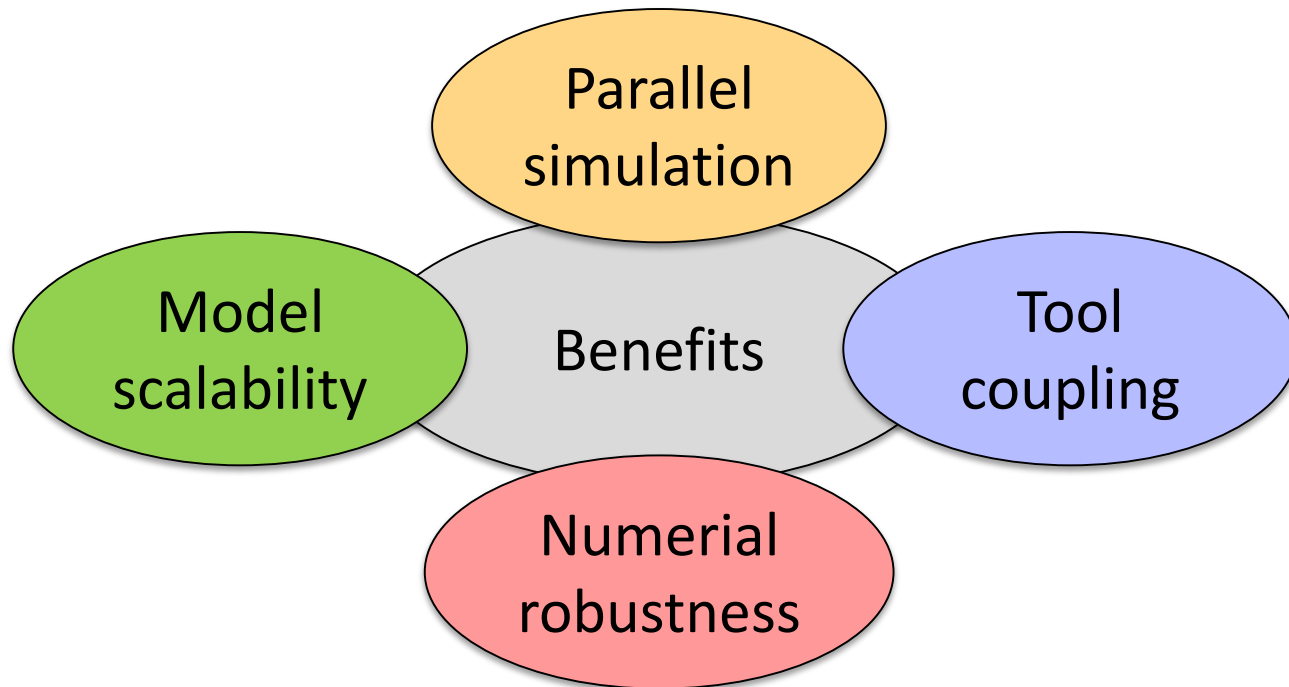
Solutions

- Variable communication step-size
- Decouple parts of model with different time scales

TLM:

- Physically motivated delay → no numerical errors!
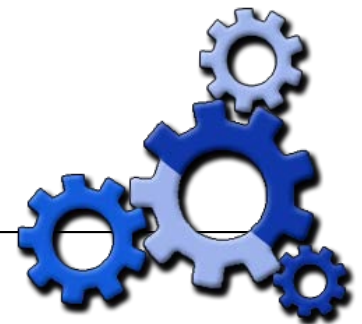
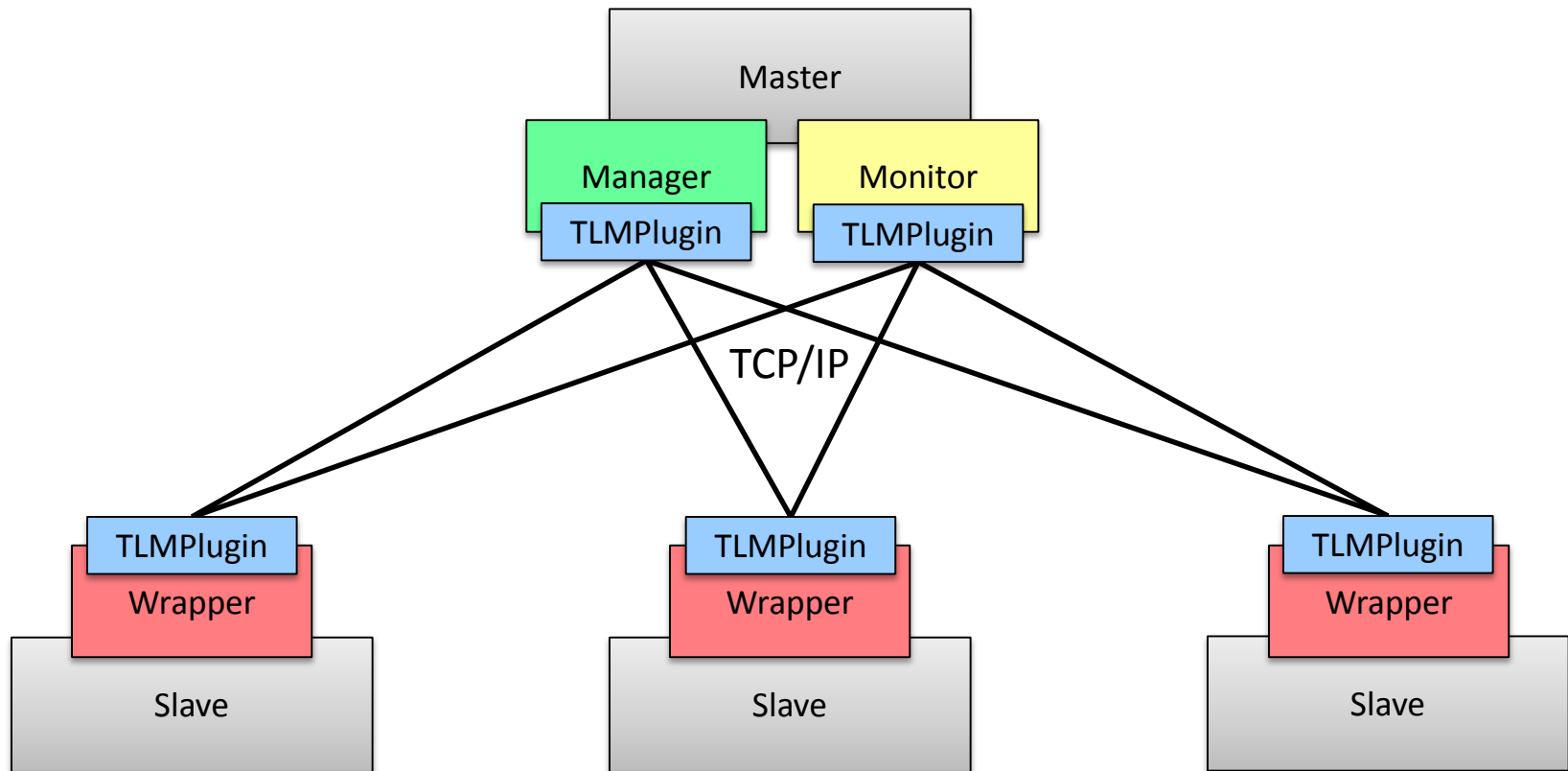# Transmission Line Modelling (TLM)

Co-simulation framework

- **Developed by SKF**
  - **Simulation of roller bearings**
- **Donated to OpenModelica Consortium**
  - **Graphical interface in OMEdit**

Co-simulation framework

XML representation of system model

```
<Model Name="MetaModel1">
  <SubModels>
    <SubModel Name="turbine" ModelFile="turbine.mo"
              StartCommand="StartTLMOpenModelica"
              Position="0,0,0" Angle321="0,0,0">
      <InterfacePoint Name="tlm1" Position="1.8,0,0" Angle321="0,0,0"/>
      <Parameter Name="x" Value="42"/>
    </SubModel>
    <SubModel Name="bearing" ModelFile="bearing.in"
              StartCommand="StartTLMBeast"
              Position="1.8,0,0" Angle321="0,0,0">
      <InterfacePoint Name="tlm1" Position="0,0,0" Angle321="0,0,0"/>
      <Parameter Name="y" Value="5.3"/>
    </SubModel>
  </SubModels>
  <Connections>
    <Connection From="bearing.tlm1" To="turbine.tlm1"
              Delay="5.7e-5" Zf="11400000" Zfr="65892000" alpha="0.3"/>
  </Connections>
  <SimulationParams StartTime="0" StopTime="5"/>
</Model>
```
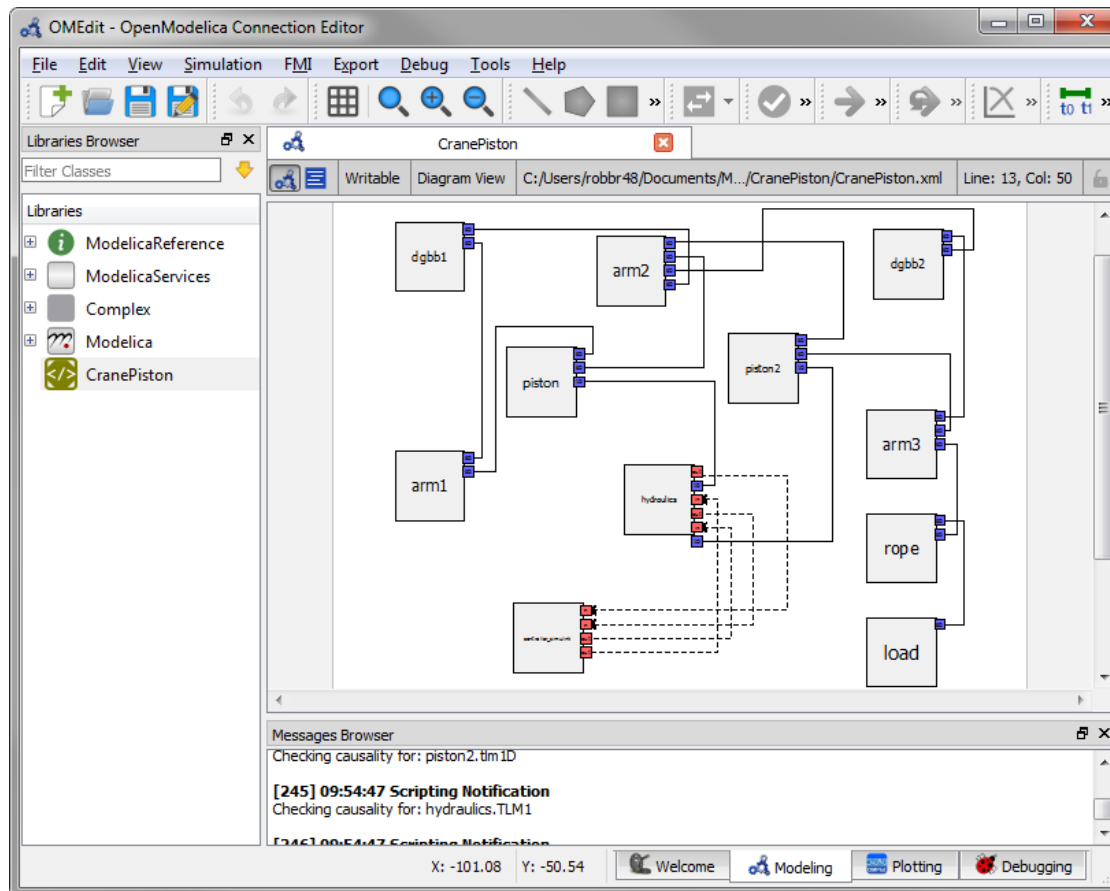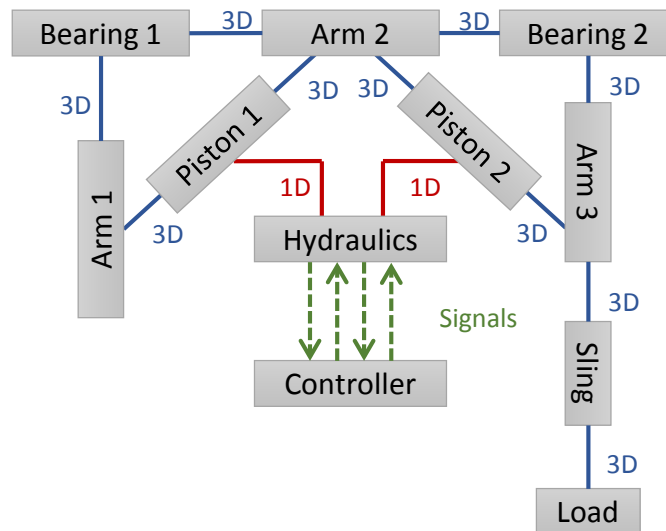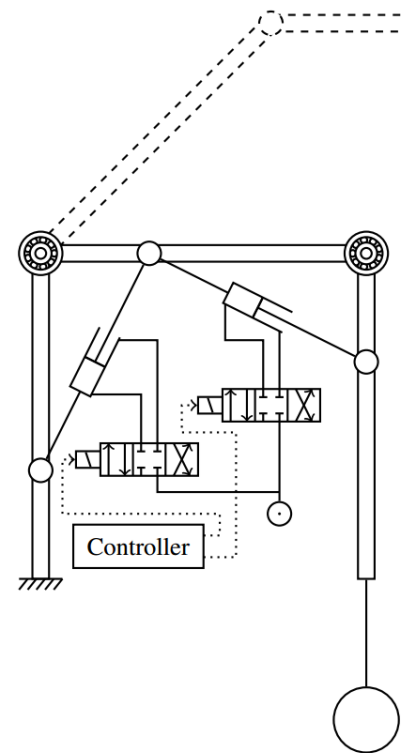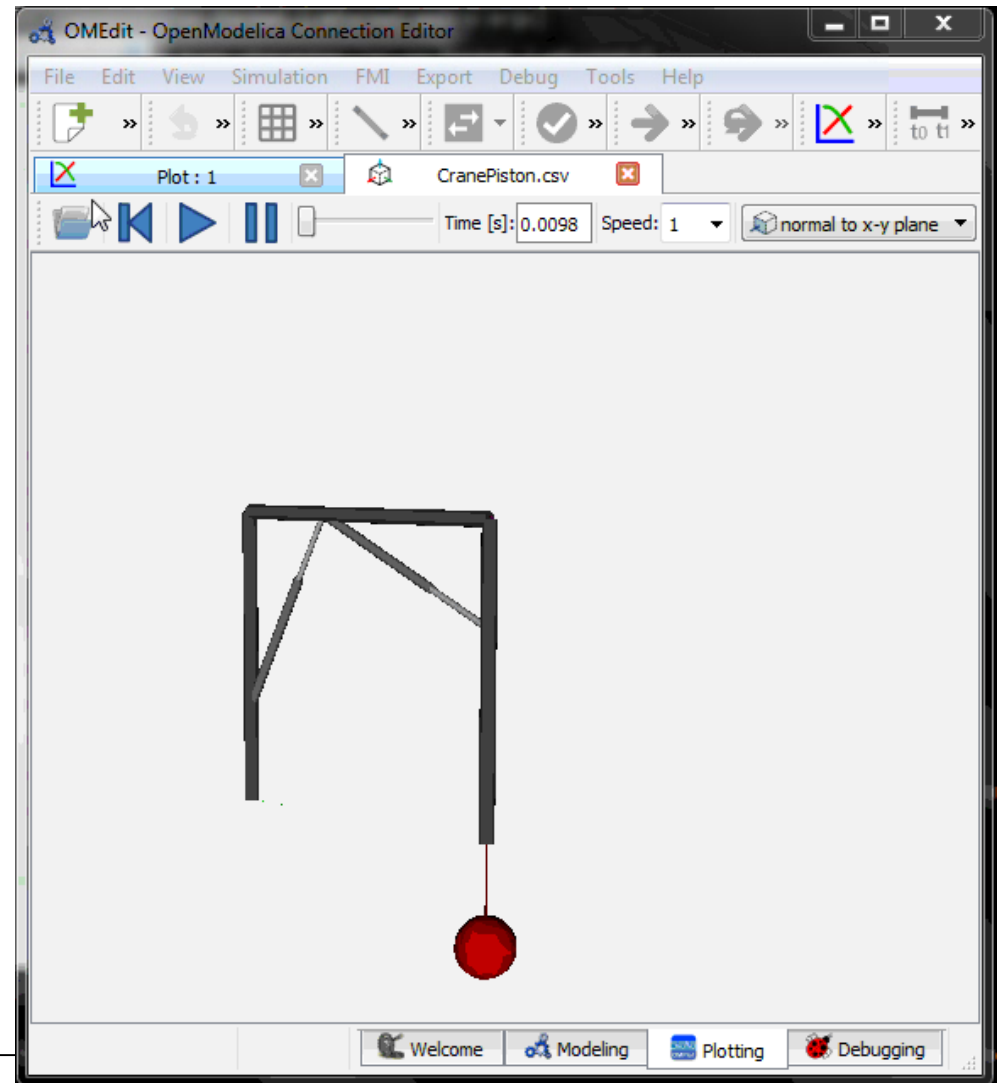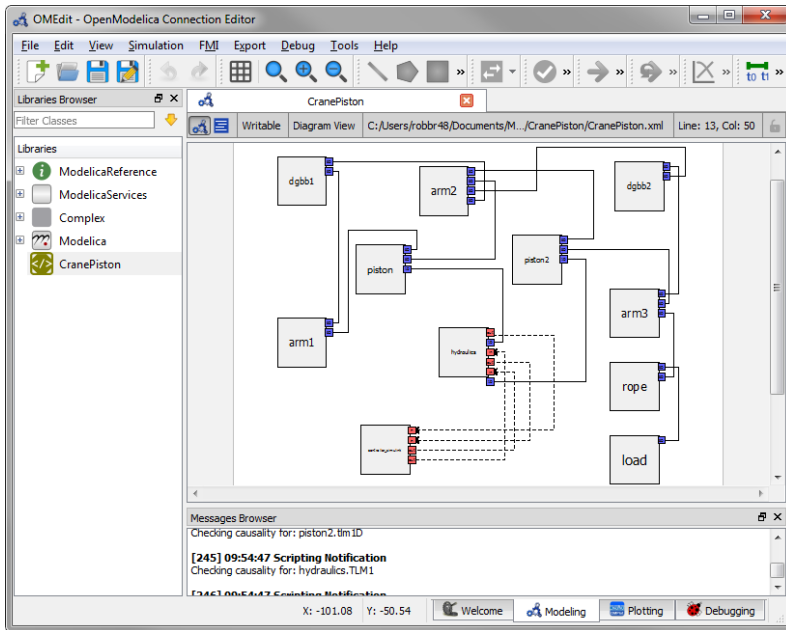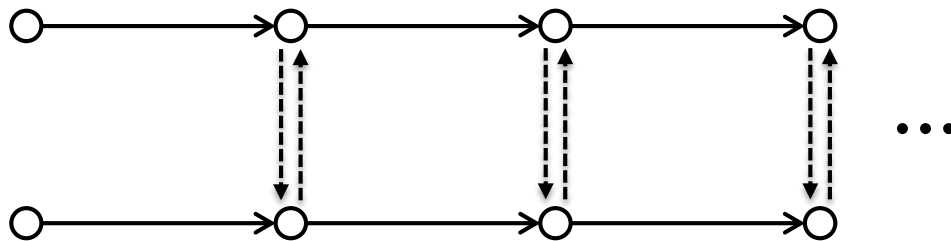
Graphical connection editor in OMEdit

Example model

Part	Setup 1	Setup 2
Arm 1	OM	Dymola
Arm 2	OM	OM
Arm 3	OM	OM
Bearing 1	BEAST	BEAST
Bearing 2	BEAST	BEAST
Piston 1	OM	OM
Piston 2	OM	OM
Sling	OM	FMI CS (OM)
Load	OM	FMI ME (OM)
Hydraulics	Hopsan	Hopsan
Controller	Simulink	FMI CS (Hopsan)

LINKÖPING UNIVERSITY
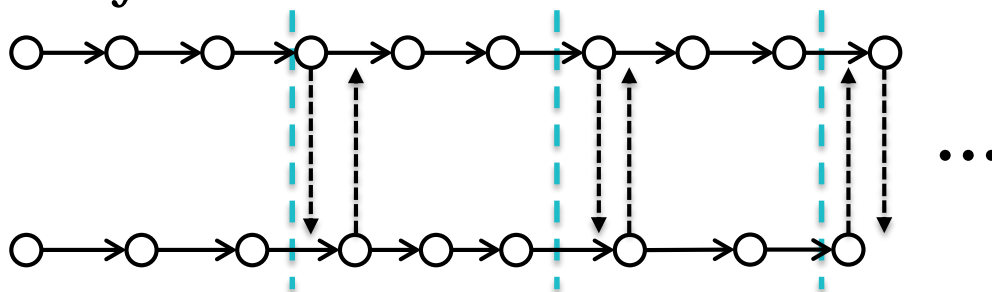
Example model

Data exchange

- **Synchronous communication**

∴ Simple to implement

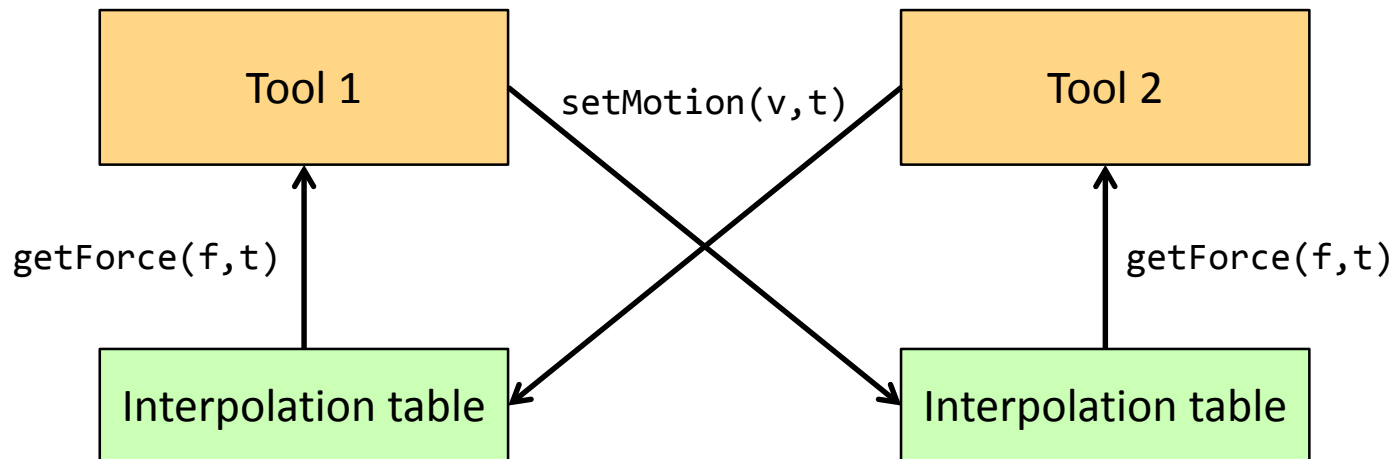∴ Requires no interpolation

- **Asynchronous communication**

∴ Variable step-size can be used!
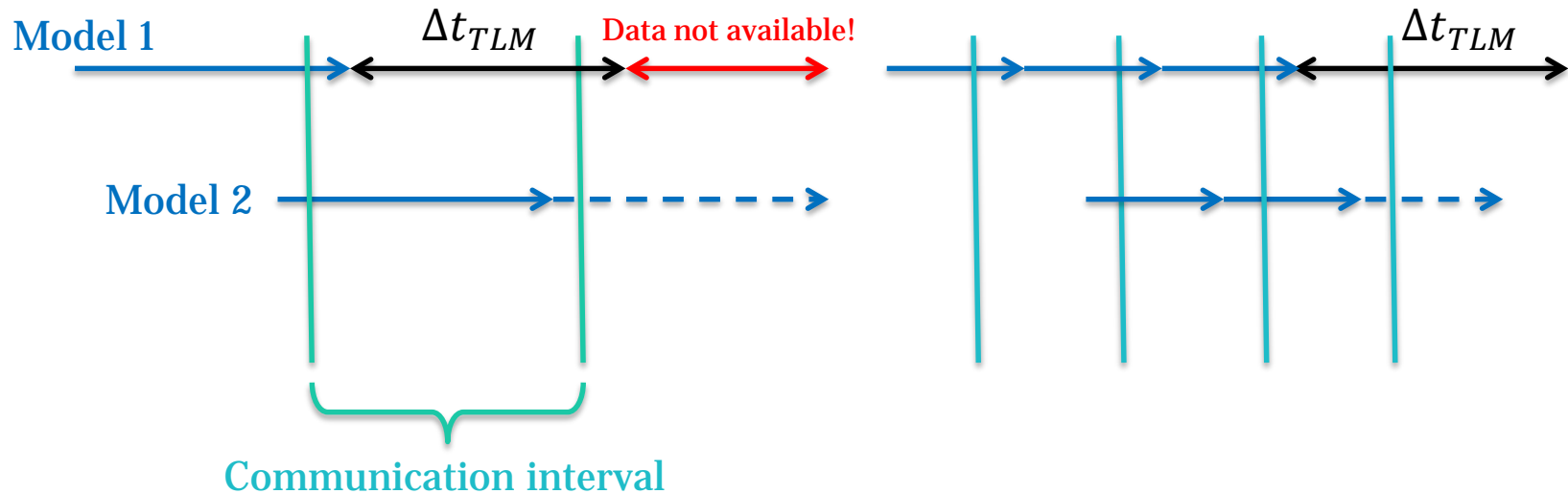
Communication points

Data exchange

- setMotion() – only at communication points
- getForce() – any time during step
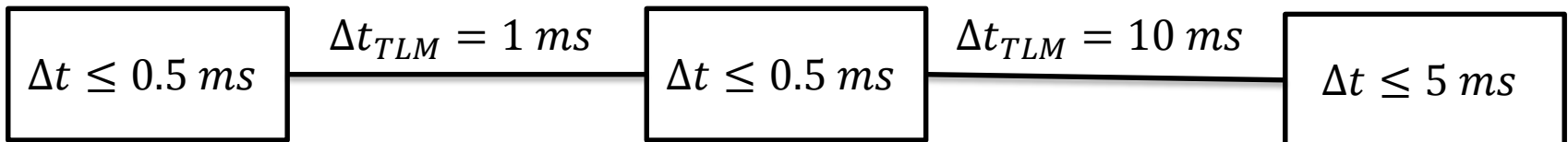
  ∴ Implicit and multi-step solvers can be used!

Data exchange

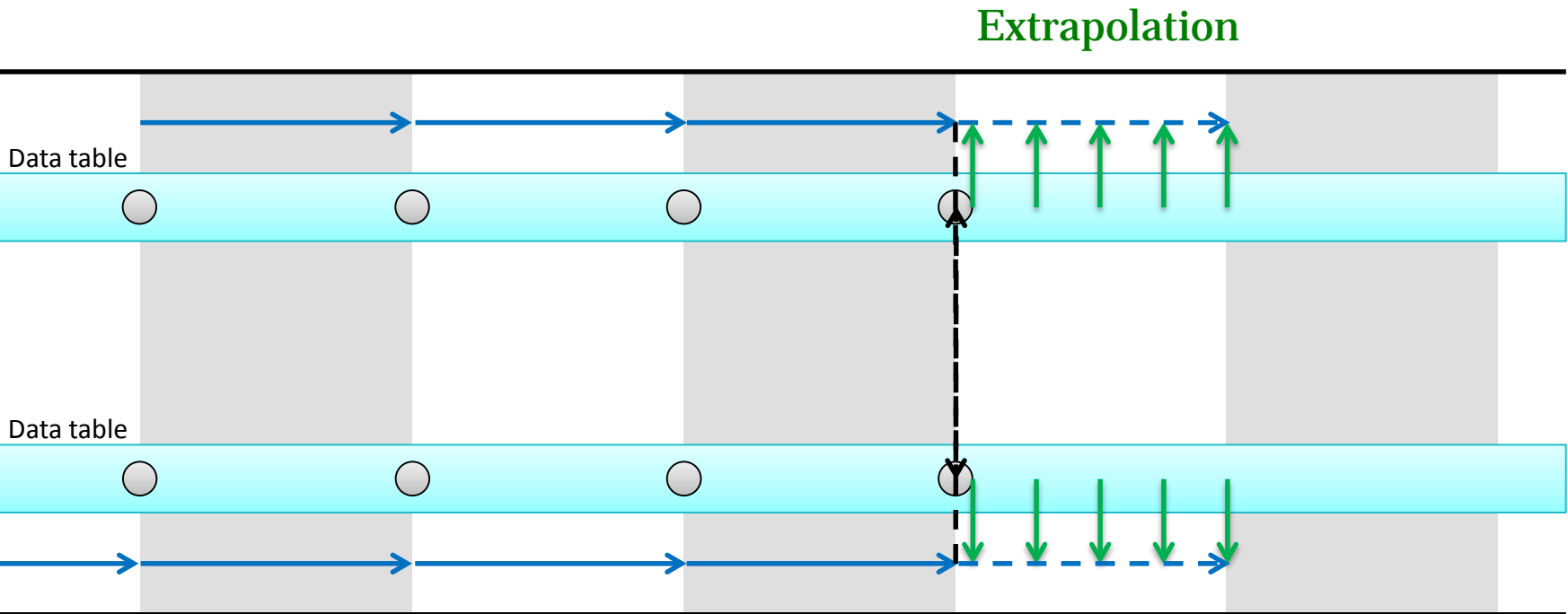- Requirement: $\Delta t_{model} \leq 0.5 \Delta t_{TLM}$

Model 1

Model 2

Δt_{TLM}

Data not available!

Δt_{TLM}

Communication interval

Data exchange

- **Requirement:** $\Delta t_{model} \leq 0.5 \Delta t_{TLM}$

$$\boxed{\Delta t \leq 0.5\ ms} \quad \underset{\Delta t_{TLM} = 1\ ms}{\rule{3cm}{0.4pt}} \quad \boxed{\Delta t \leq 0.5\ ms} \quad \underset{\Delta t_{TLM} = 10\ ms}{\rule{3cm}{0.4pt}} \quad \boxed{\Delta t \leq 5\ ms}$$

Data exchange (no TLM)

Extrapolation

Data table

Data table

Data exchange (TLM)

Data avaiable until here

Δt_{TLM}

Interpolation!

Interpolation table

Interpolation table
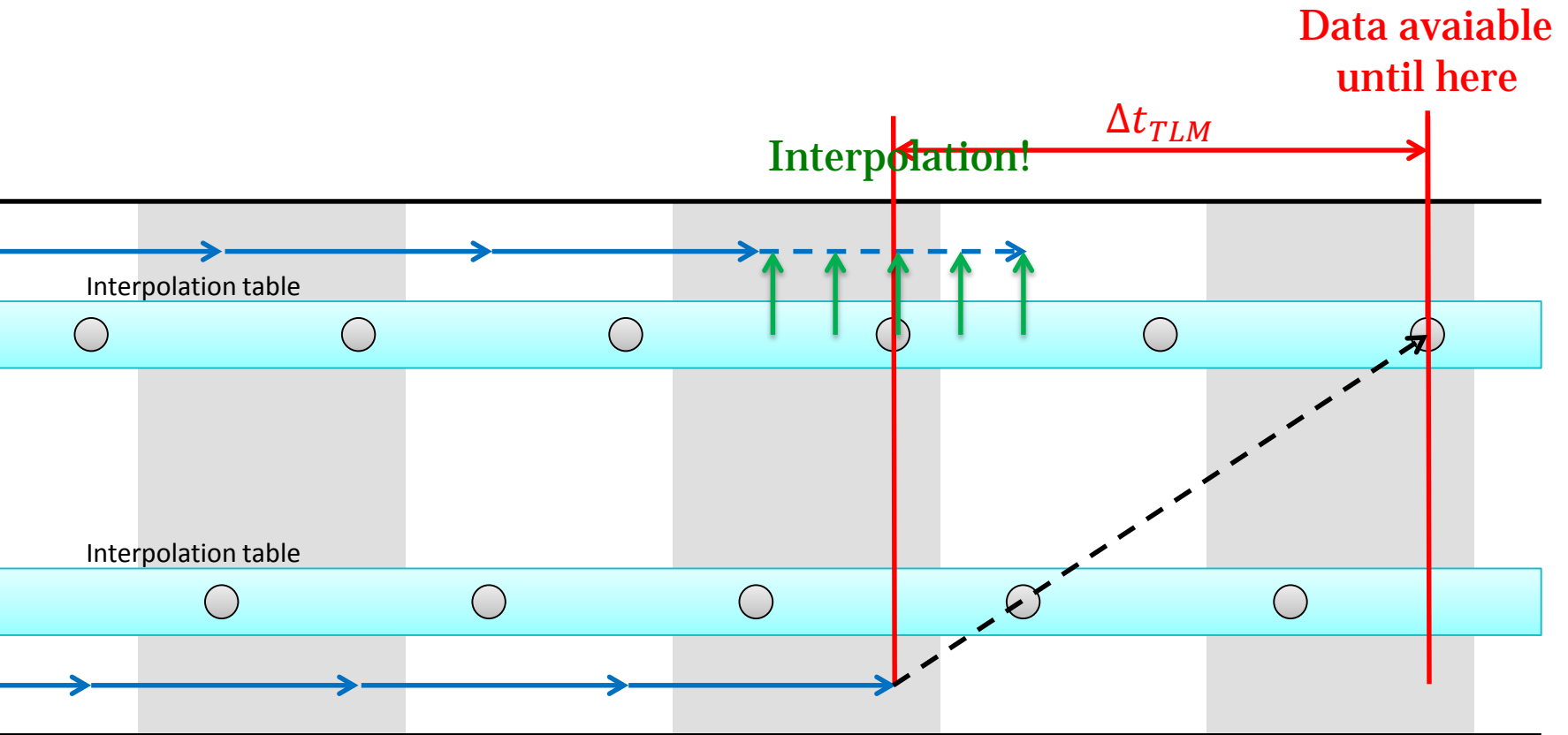
Execution models

- **Top-down:**

"Slave, here is your input!"

Master

"Slave, take a step!"

"Slave, give me your output!"

Input

Output

Slave

- **Bottom-up:**

"Master, give me my input!"

Master

Input

"I do something!"

"Master, here is my output!"

Output

Slave
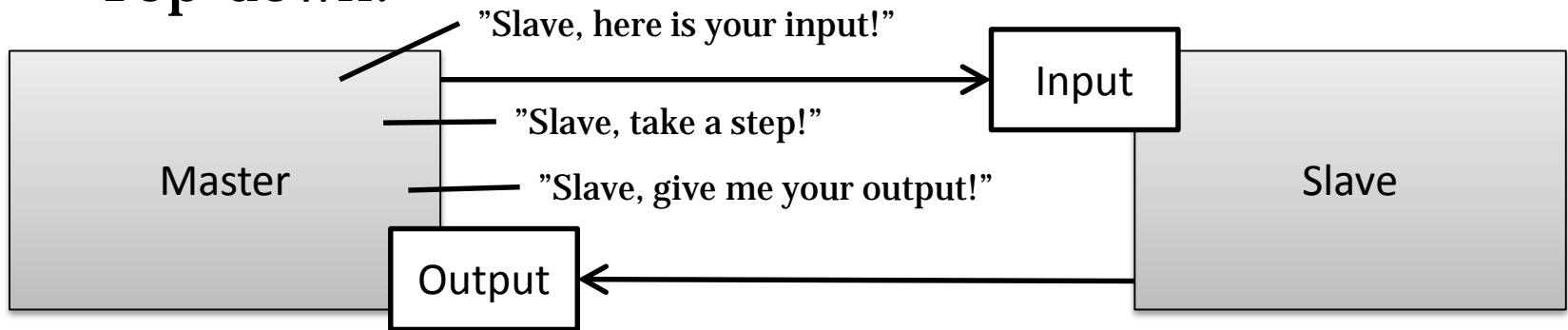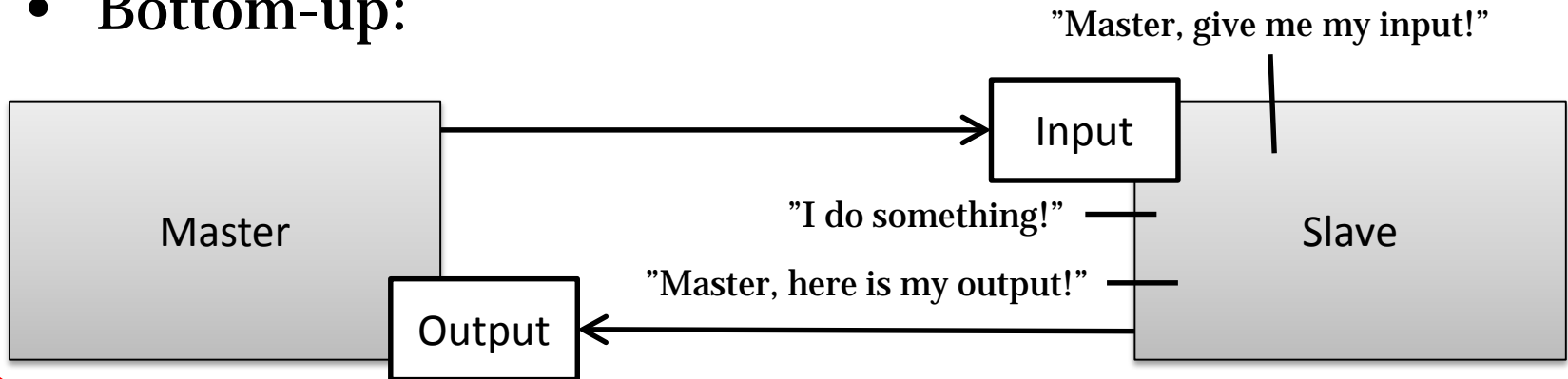
Problem description

FMI for co-simulation:

Master

setReal(u)   doStep(t)   getReal(y)

Slave

TLM-based co-simulation:

Master

getForce(f,v,t)   setMotion(v,t)

Slave
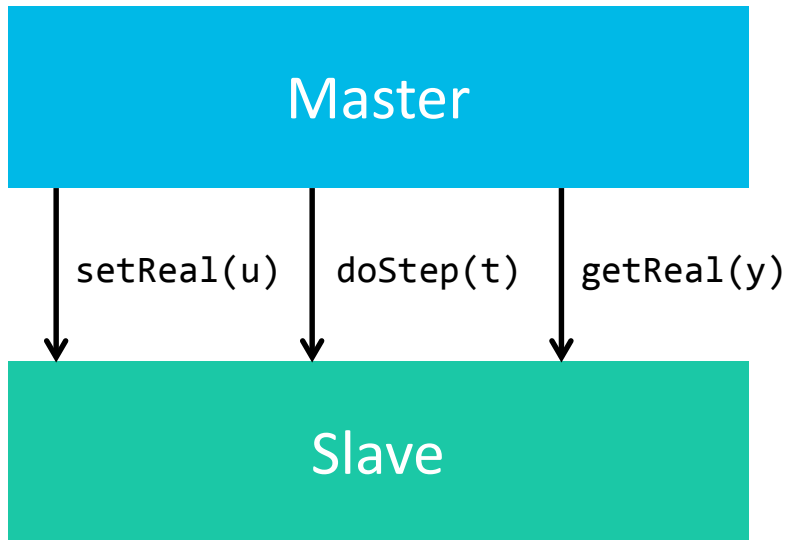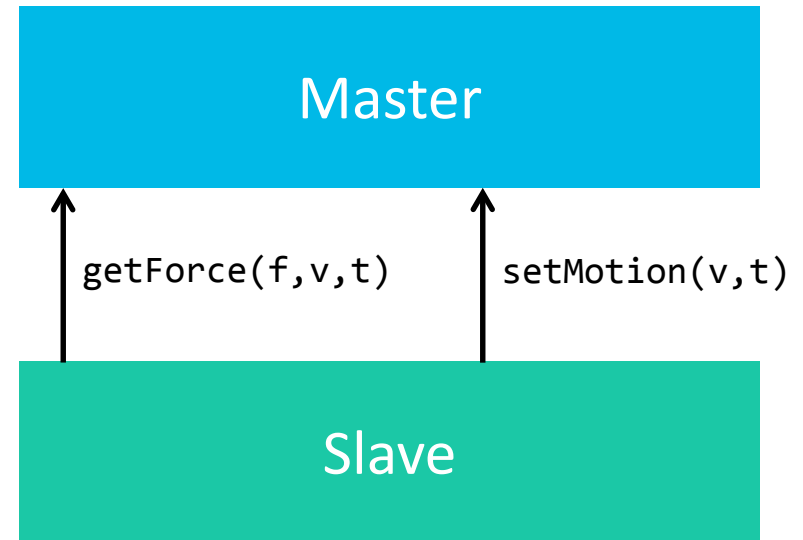
Wrong direction of the arrows!

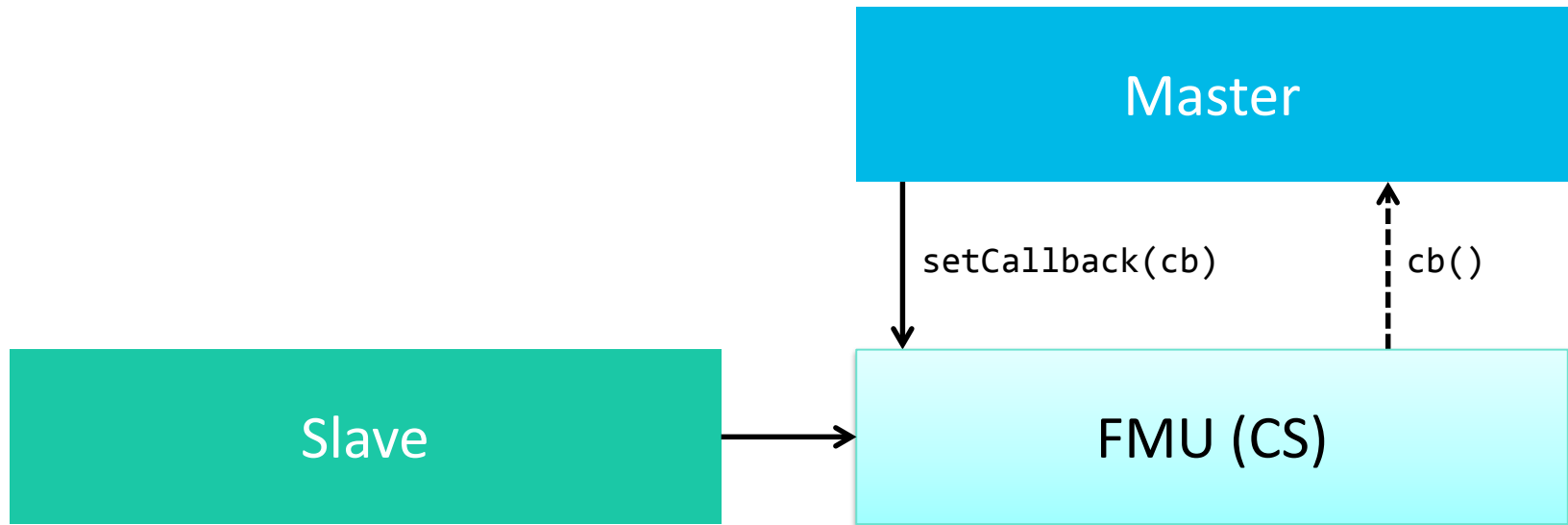LINKÖPING UNIVERSITY

Solutions

Divide steps into substeps

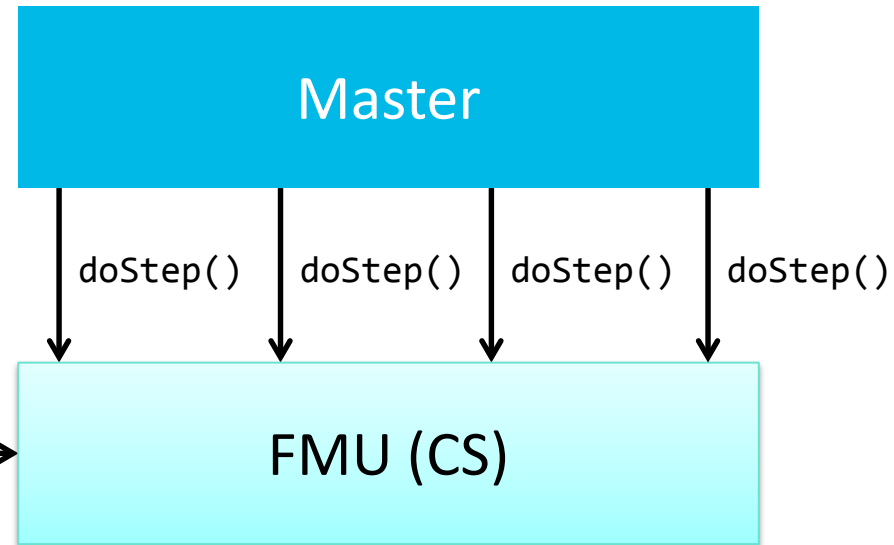✖ Contradicts FMI standard

Solutions

Divide steps into substeps

- ✗ Negative impact on performance
- ✗ Stability not guaranteed


```
while(t<tmax) {

  double f,x,v;

  size_t nSubSteps=100;

  for(size_t i=0; i<nSubSteps; ++i) {

    pFmu->setReal(0,pPlugin->getForce(t-dtTLM));

    t+=dtTLM/nSubSteps;

    pFmu->doStep(t);

    pPlugin->setMotion(t,pFmu->getReal(1));

  }

}
```
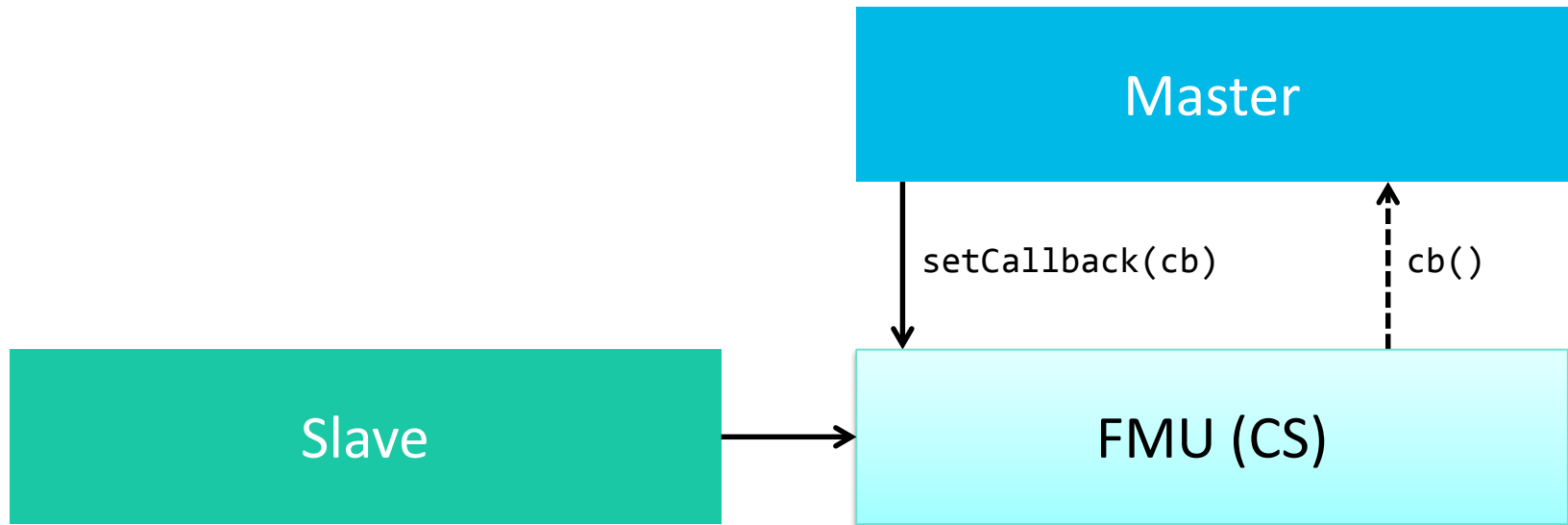
Master

doStep() doStep() doStep() doStep()

FMU (CS)

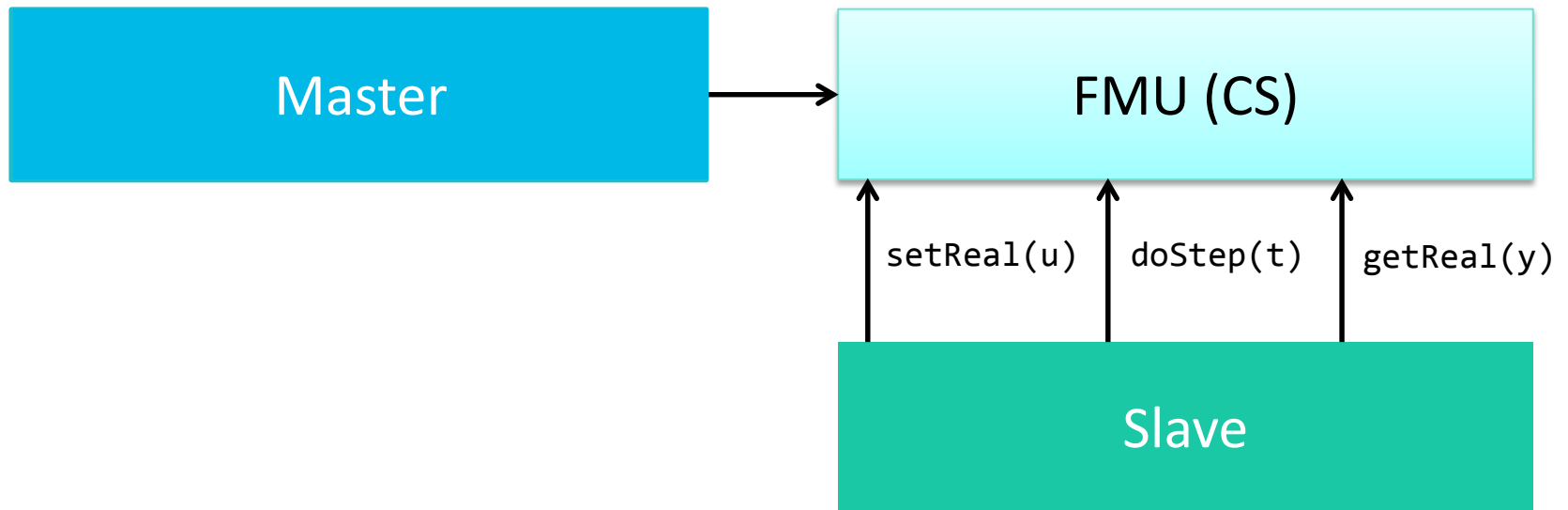Divide into 100 substeps

Solutions

Callback function

✗ Contradicts FMI standard

Solutions
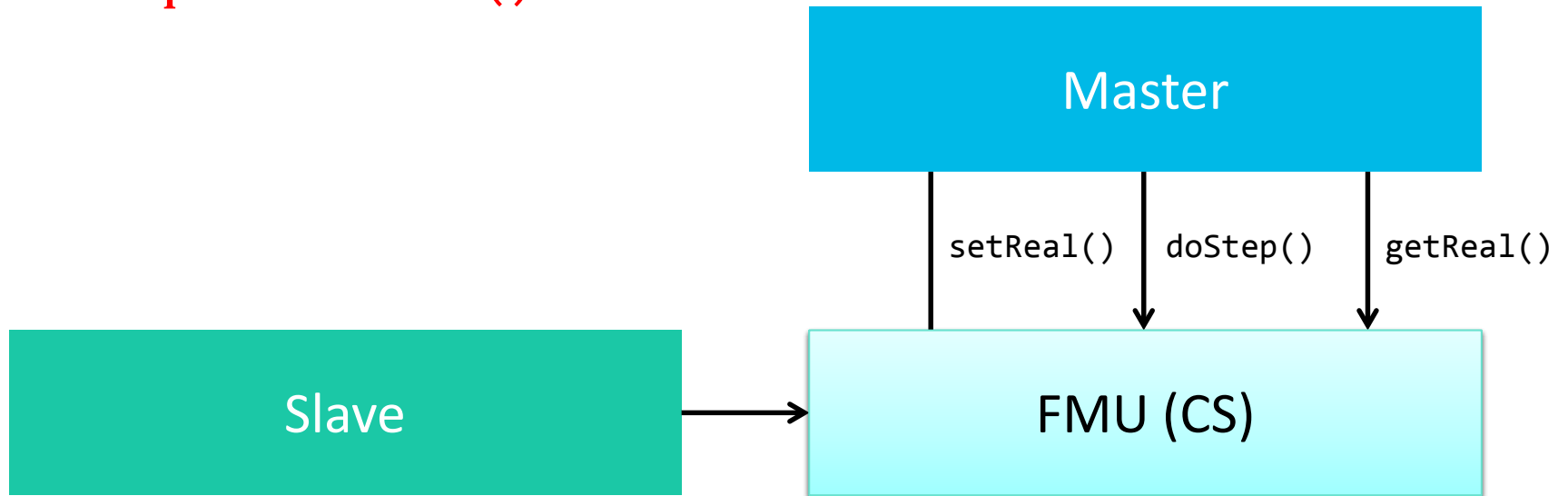
Export FMU for co-simulation from master to slave

- ✗ Not supported by general simulation tools
- ✗ Slave must control simulation time

LINKÖPING UNIVERSITY

Solutions

Interapolation in slave FMU

✖ Requires `setReal()` for time `t`

Outlook

- Prototypes of solutions
    - Suggest improvements to standard?
- Verify on industrial demonstrators
- Support for discrete-event simulation

Robert Braun

www.liu.se

Robert Braun

www.liu.se

Robert Braun

www.liu.se

LiU LINKÖPING UNIVERSITY