Towards Numerically Stable Co-simulation with FMI 3.0 using On-demand Interpolation

Robert Braun
Linköping University
Division of Fluid and Mechatronic Systems

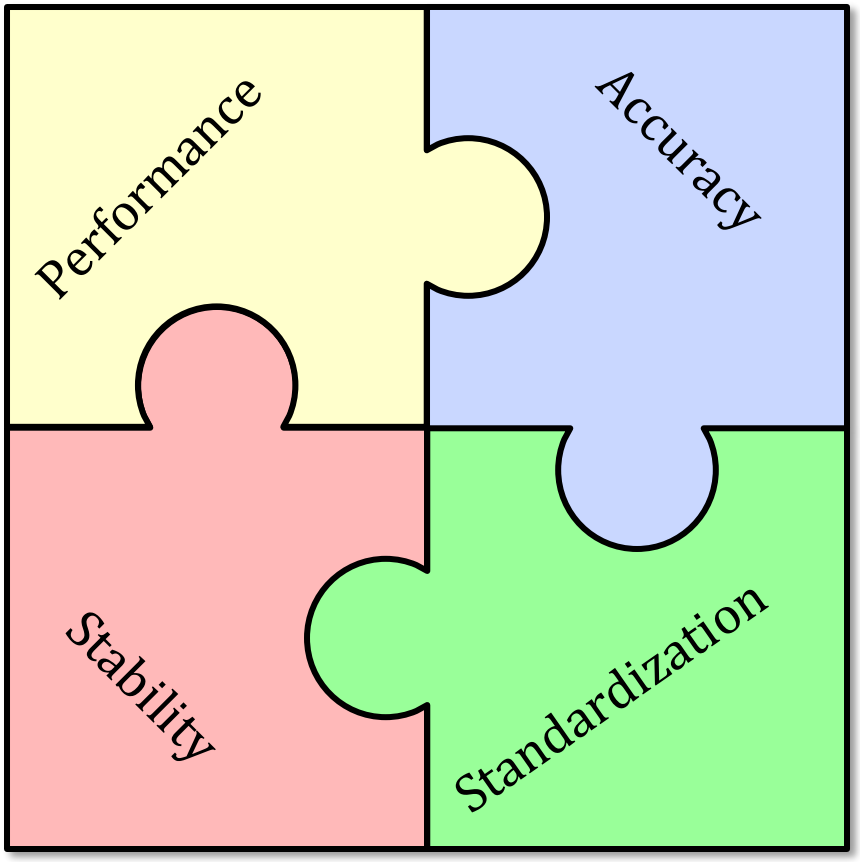
LINKÖPING UNIVERSITY

Motivation – Co-simulation

Benefits:

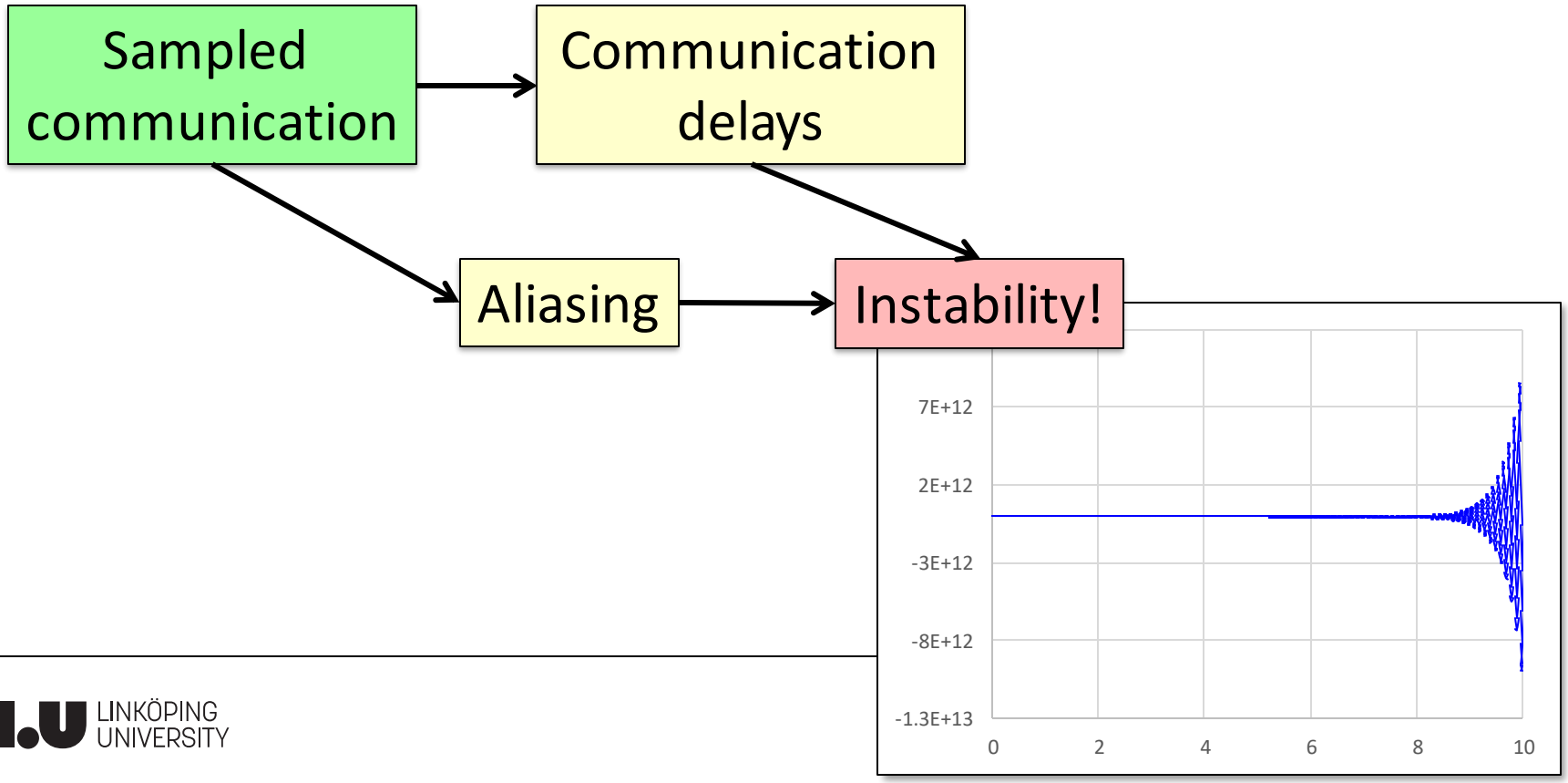
- Facilitates collaboration
- Useful for multi-domain simulation
- Each sub-model can use its best suitable solver
- Preserves investments

Motivation – Co-simulation



Motivation

Co-simulation relies on sampled communication:



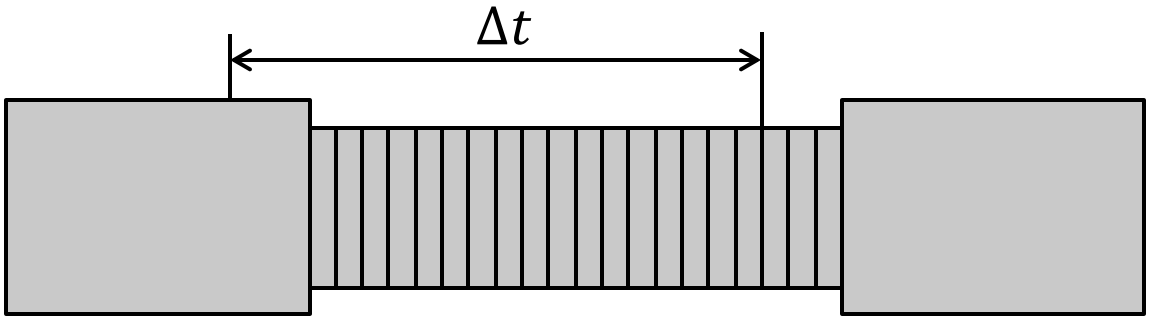
Motivation

Co-simulation execution methods:

	Requires Rollback	Ensures Stability
Fixed communication step	no	no
Extrapolation	no	no
Adaptive communication step-size	yes	yes
Waveform relaxation	yes	yes
Bi-lateral delay models (TLM)	no	yes

Transmission Line Modelling (TLM)

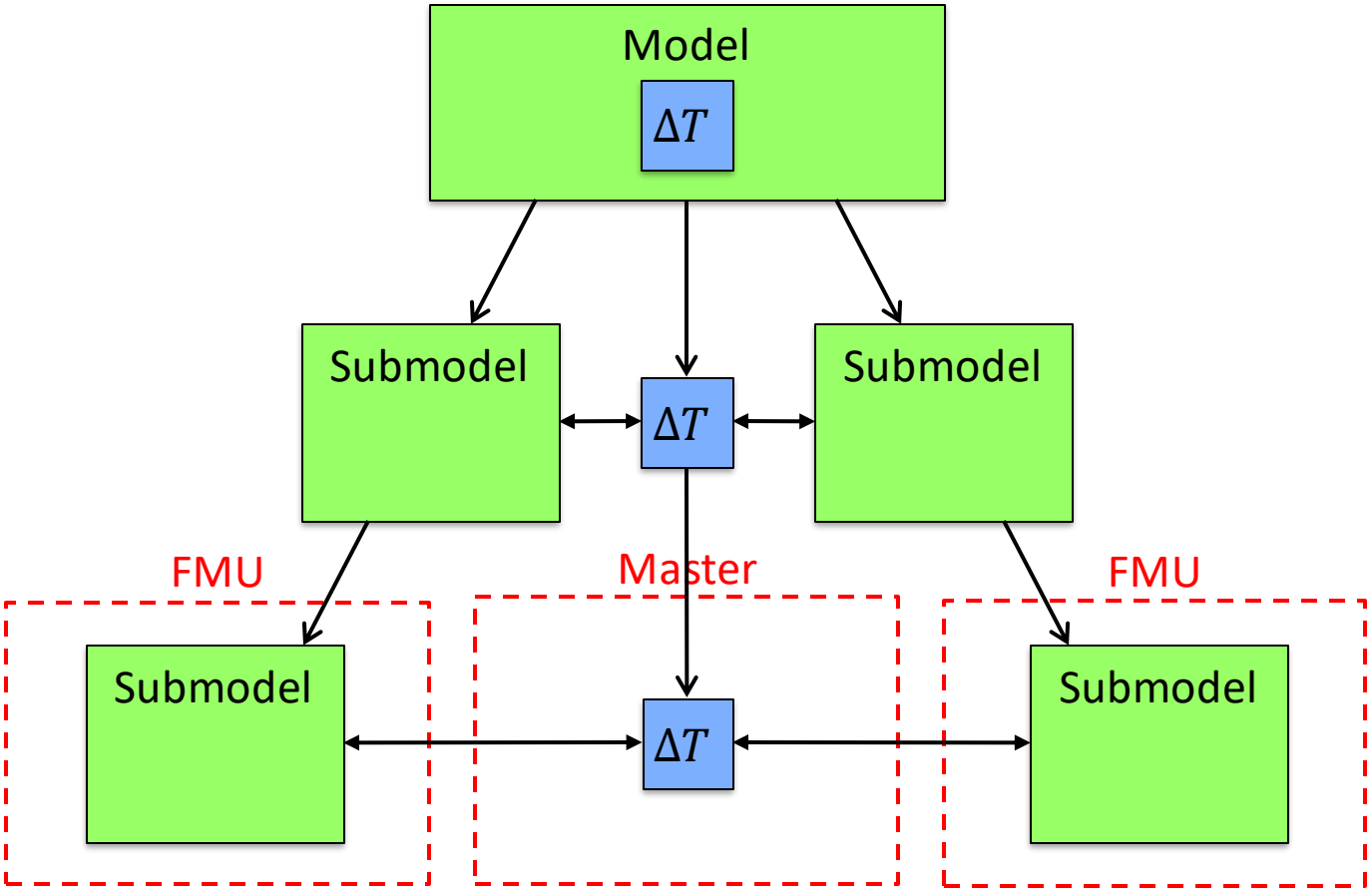
Every physical element has a natural time delay:



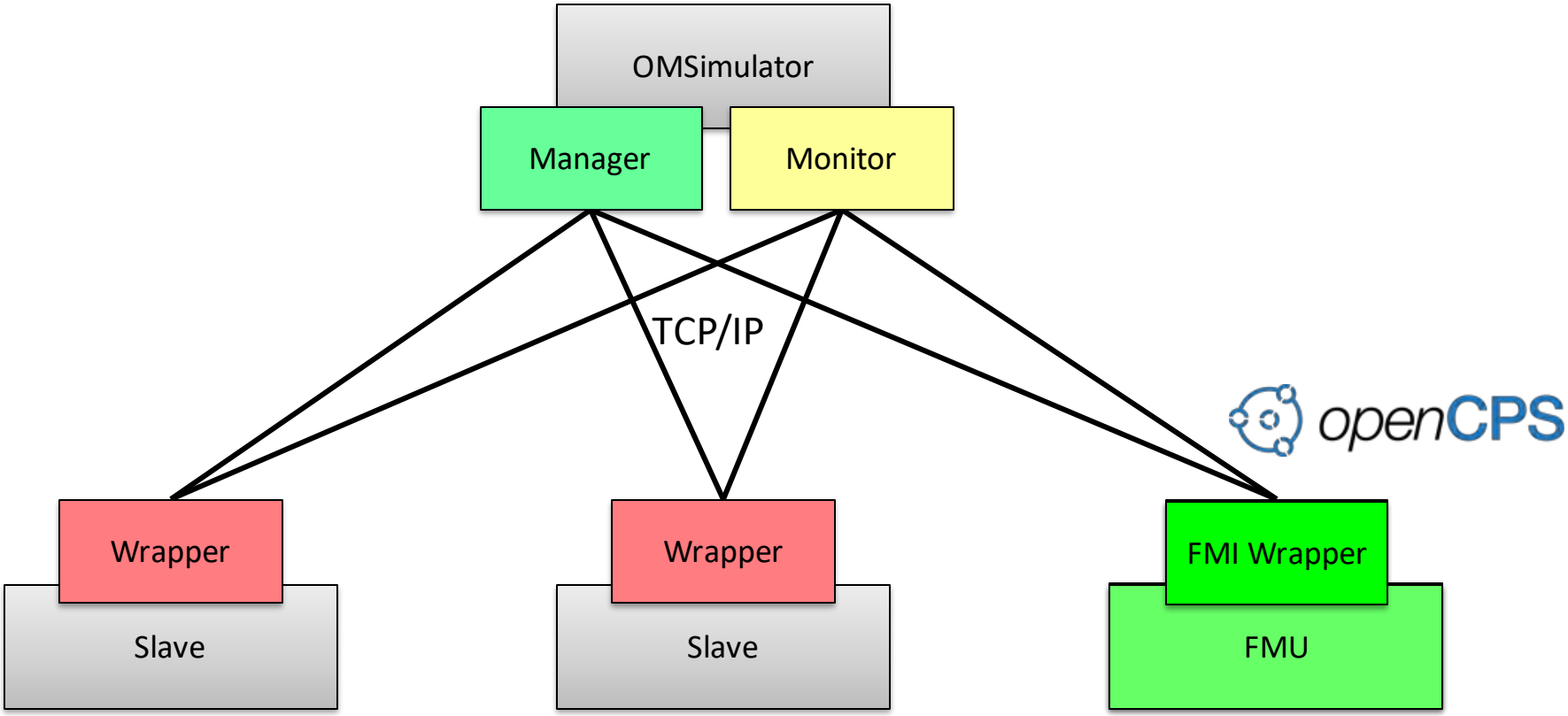
Wave equation:

$$\frac{\partial^2 u}{\partial t^2} = c^2 \frac{\partial^2 u}{\partial x^2}$$

TLM for Co-simulation

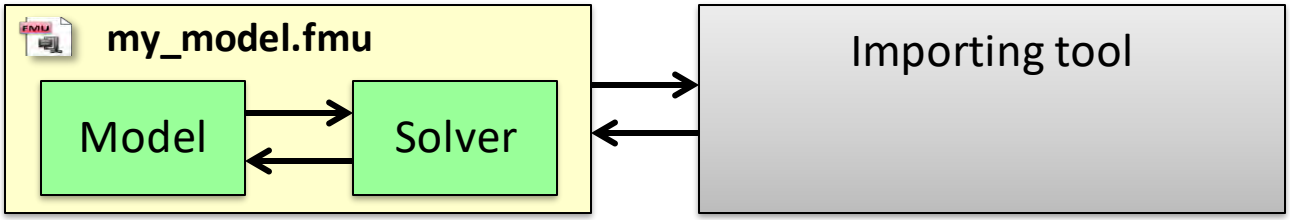


Existing TLM Solution

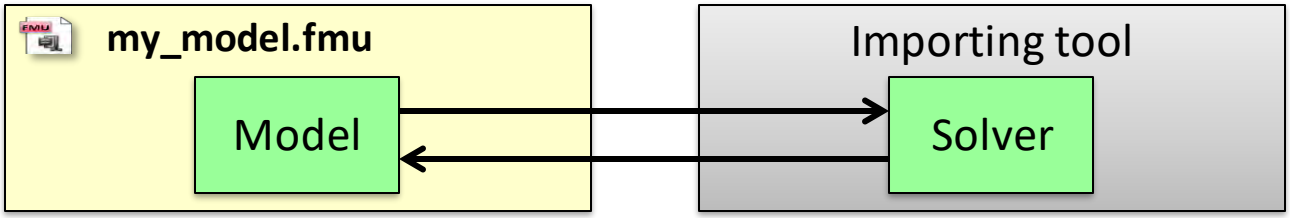


Functional Mockup Interface (FMI)

FMI for Co-simulation



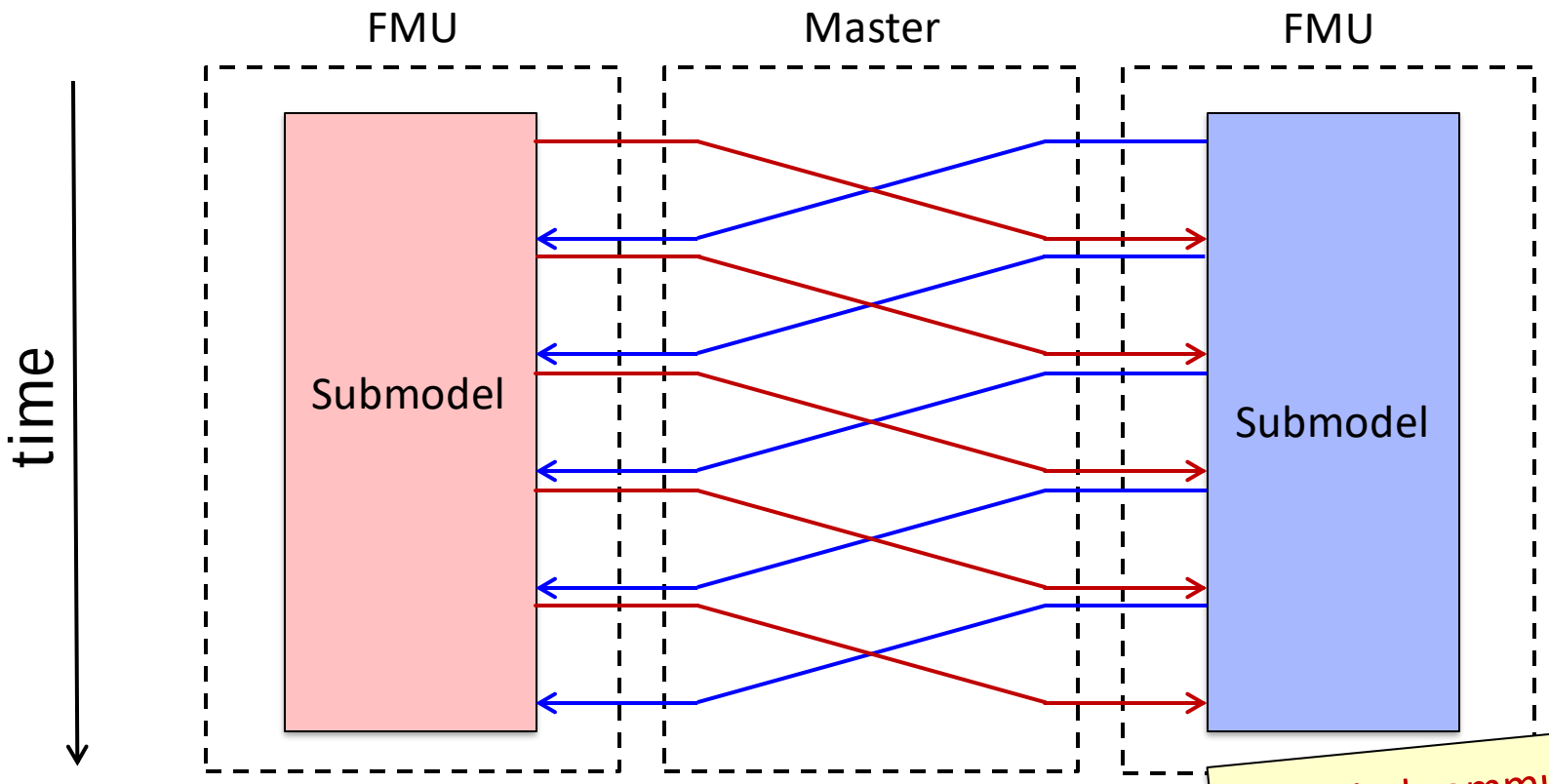
FMI for Model Exchange



Current version: 2.0.1

Functional Mock-Up Interface

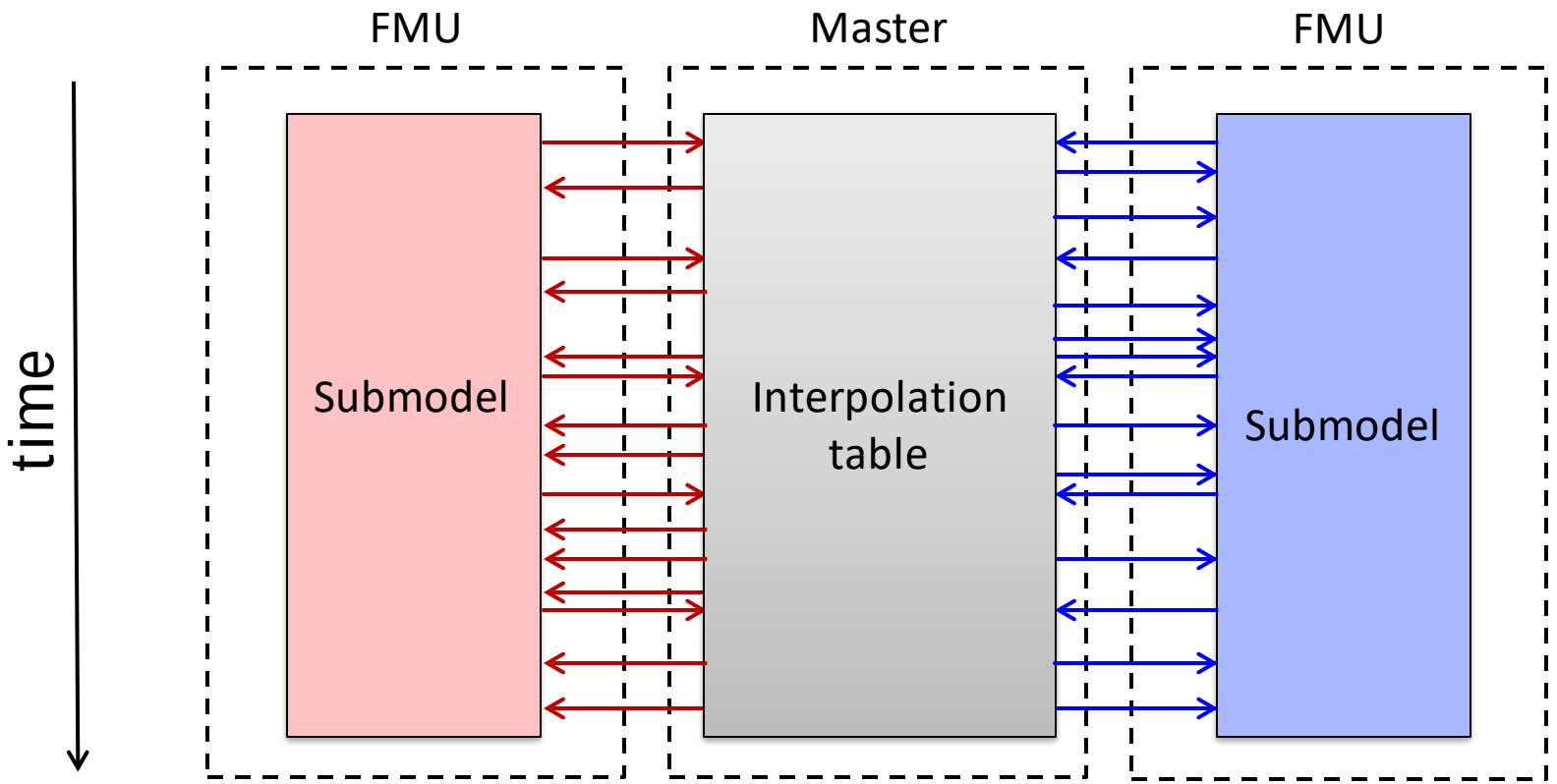
Synchronous data exchange



Sampled communication, stability not guaranteed

Asynchronous data exchange

Not available in FMI 2.0

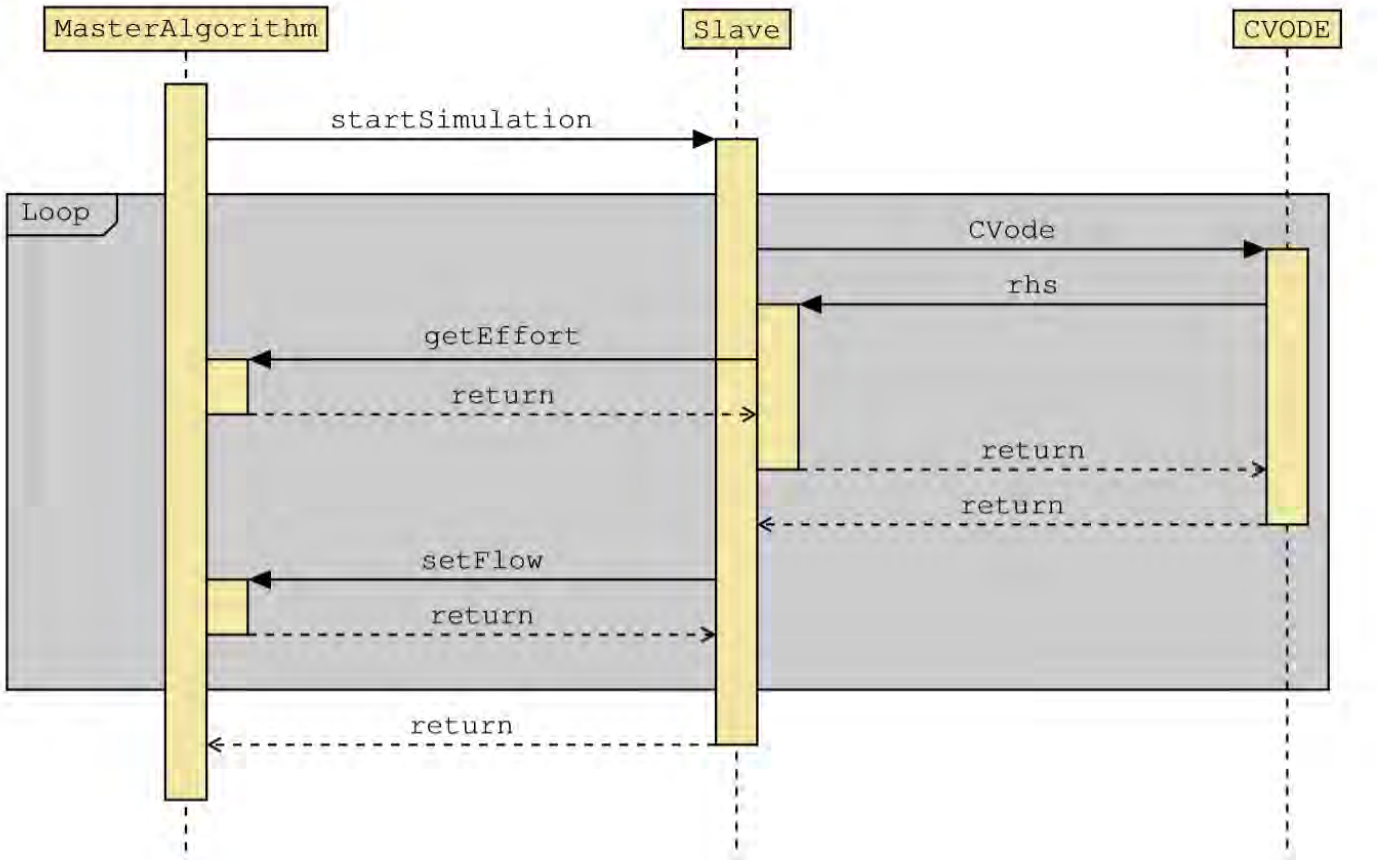


Asynchronous data exchange

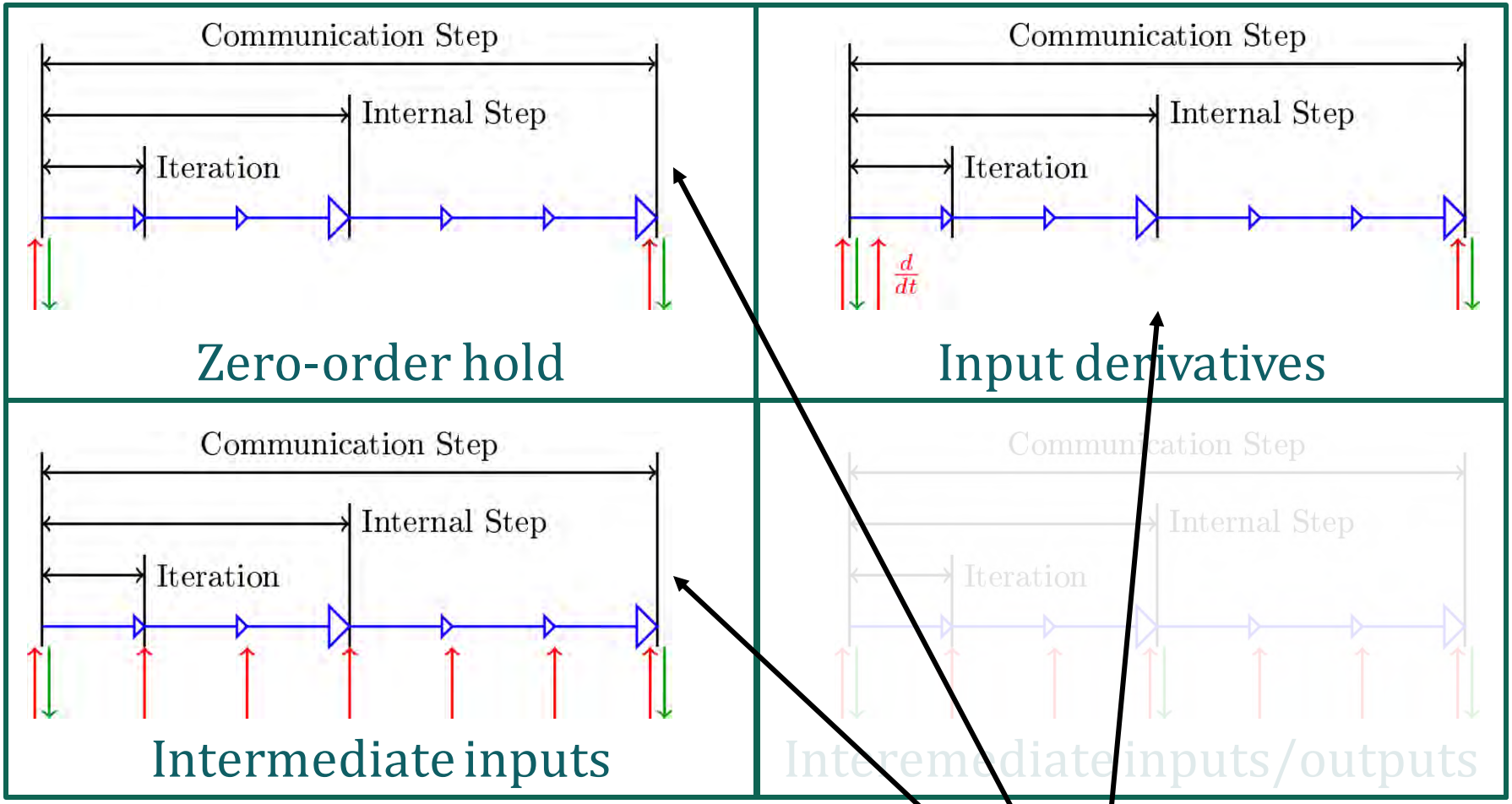
Requirements:

- Continuous data exchange
- Access to variables at intermediate solver states

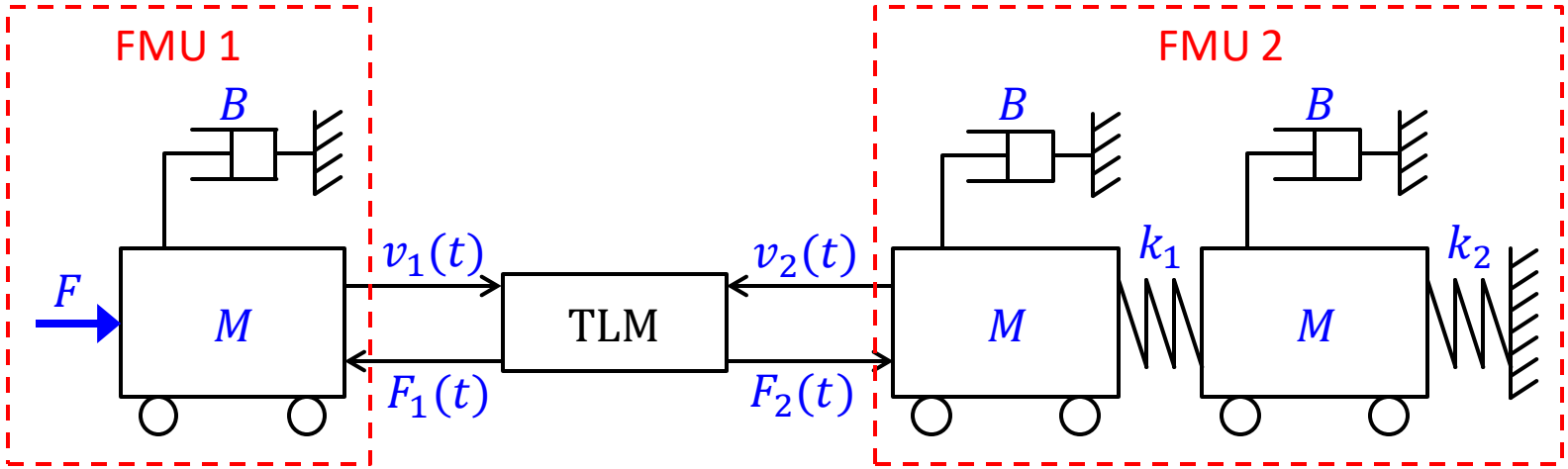
UML Sequence Diagram (CVODE)



Communication Patterns

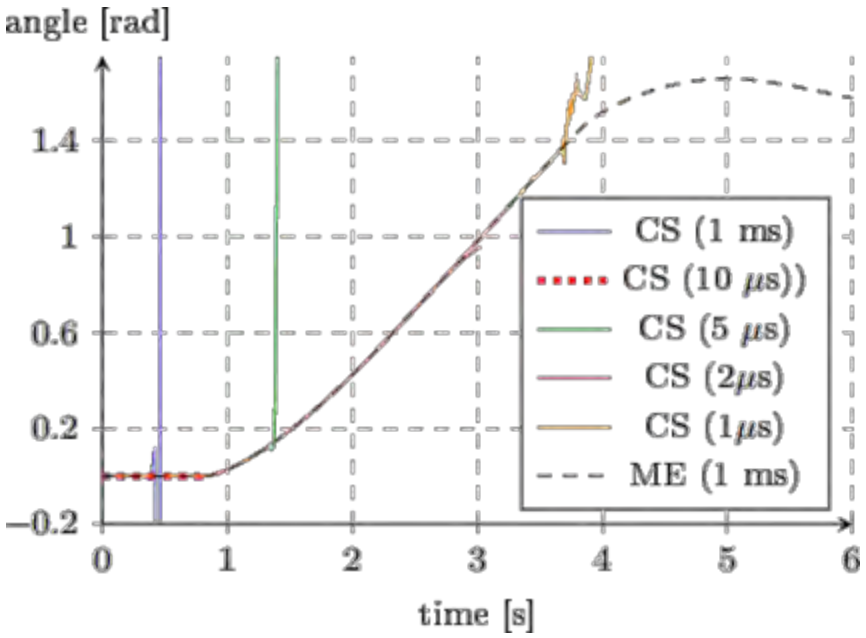


1D Test Model



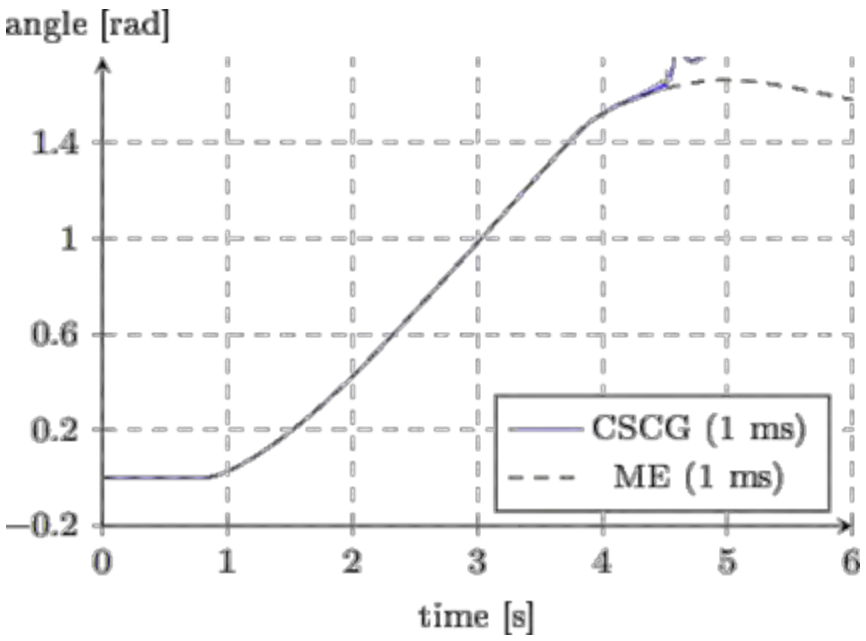
State-of-the-Art: Zero-order Hold

- Not stable after reducing step-size 1000 times
- Compatible with all FMI tools



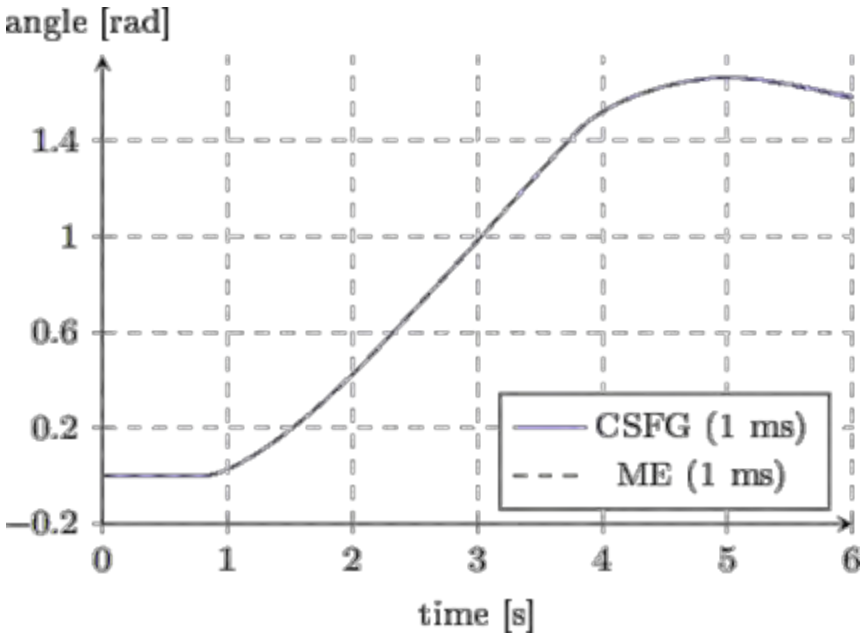
Interpolation using Input Derivatives

- Almost stable without reducing step-size
- Not supported by all FMI tools



Intermediate Input Variables

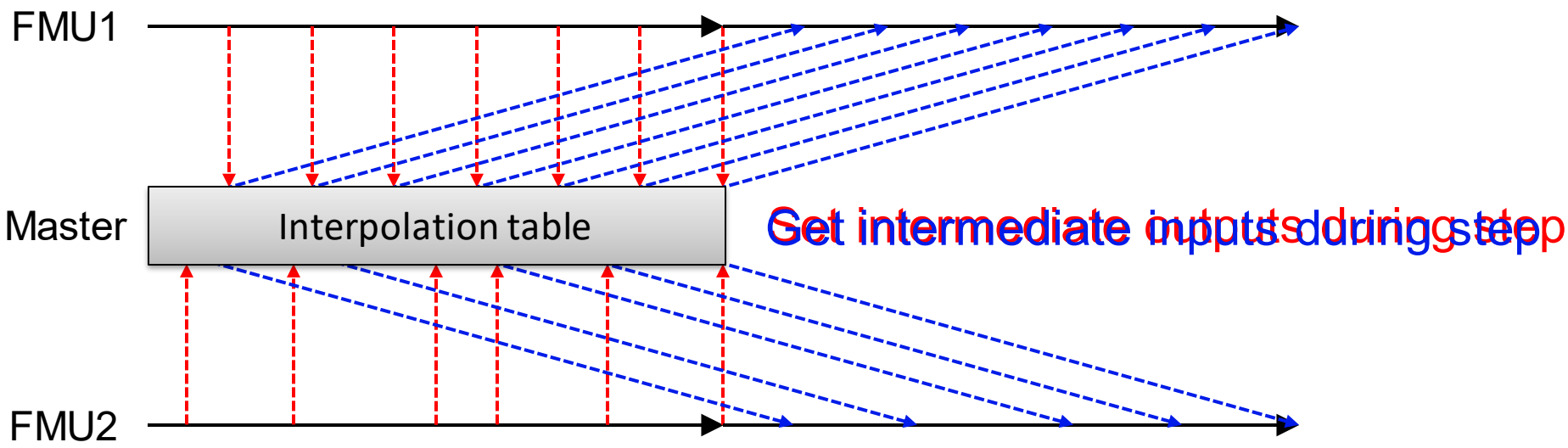
- Stable without reducing step-size
- Not supported by FMI standard



FMI Change Proposal

Intermediate Variable Access (IVA)

- Send output variables to master whenever produced
- Request input variables at any point during step



Suggested Implementation

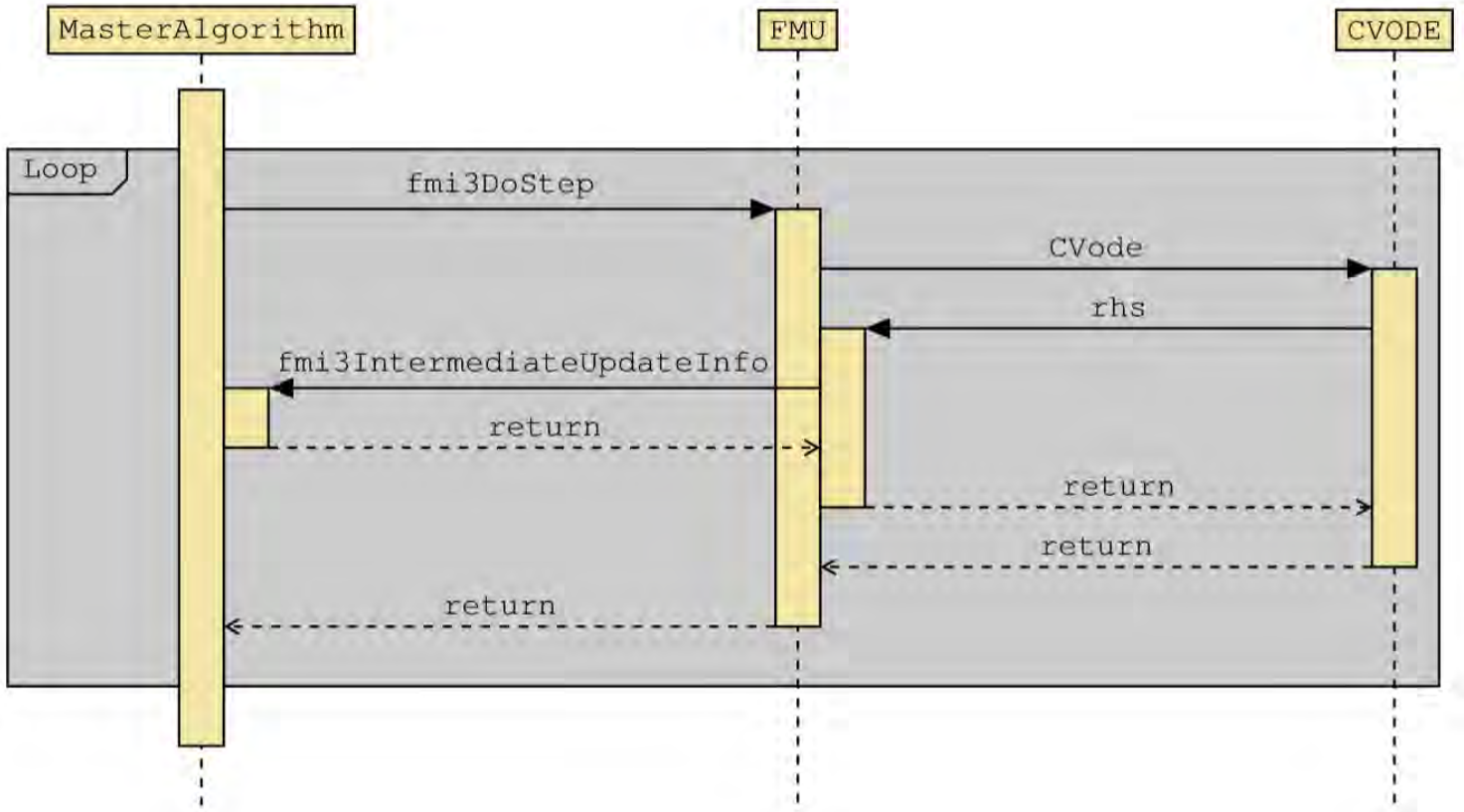
```
typedef struct {
  fmi3Float64 intermediateUpdateTime;
  fmi3Boolean eventOccurred;
  fmi3Boolean clocksTicked;
  fmi3Boolean intermediateVariableSetAllowed;
  fmi3Boolean intermediateVariableGetAllowed;
  fmi3Boolean intermediateStepFinished;
  fmi3Boolean canReturnEarly;
} fmi3IntermediateUpdateInfo;
```

Master can call setReal()

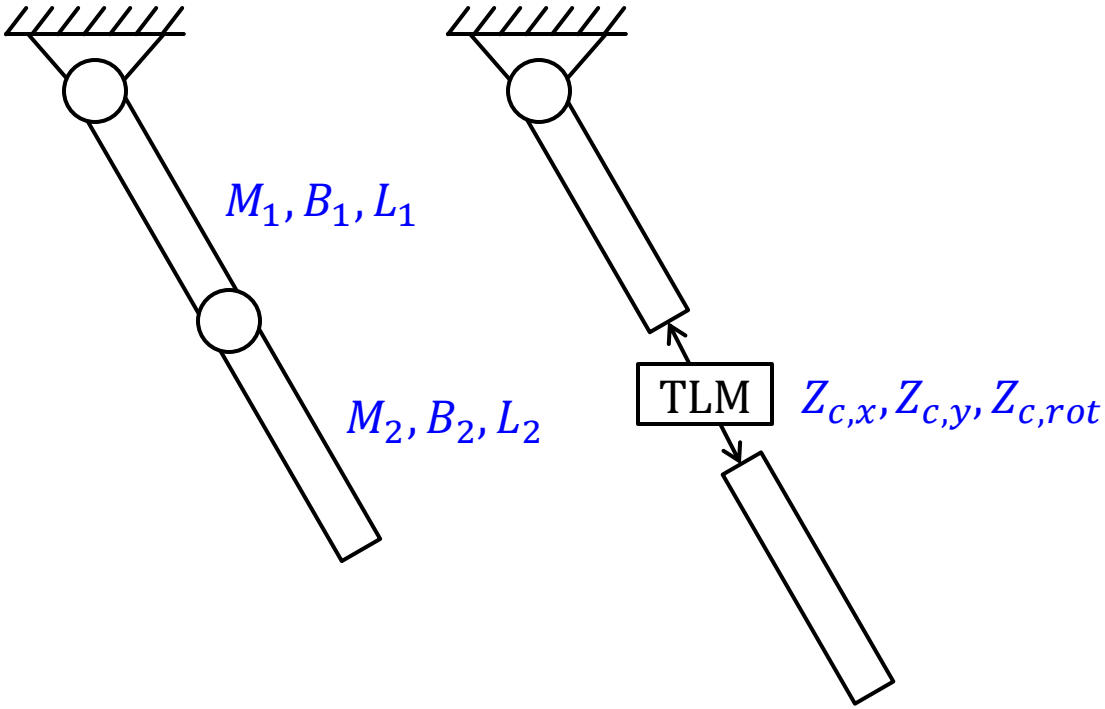
Master can call getReal()

If false, master can only use output variables to compute immediate input variables

UML Sequence Diagram (CVODE/FMI 3.0)

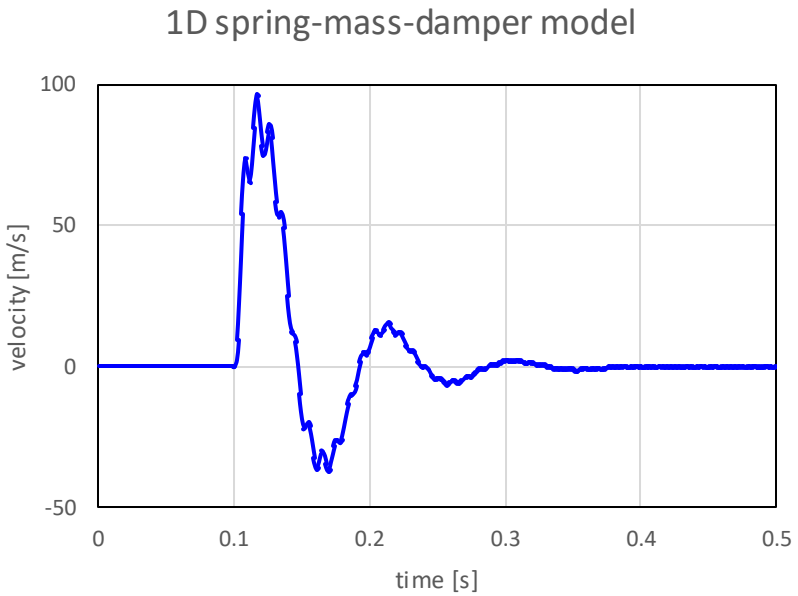


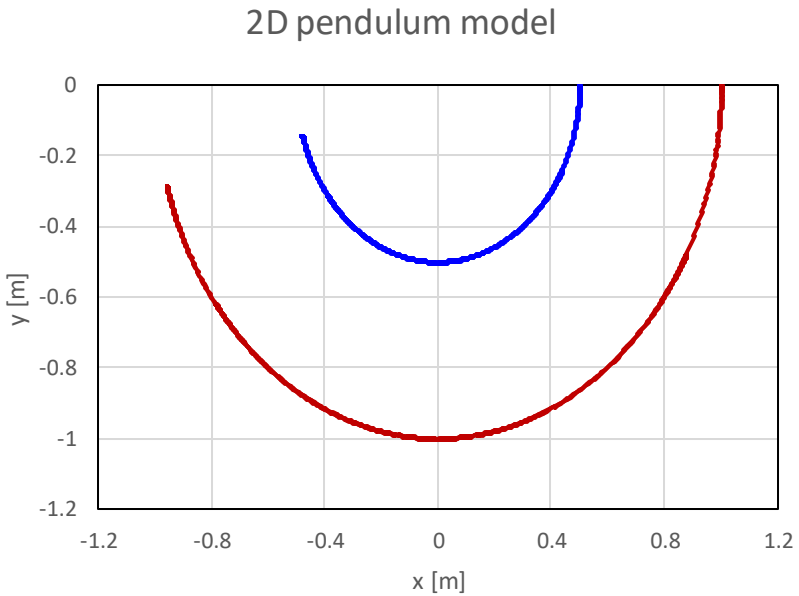
2D Test Model



Experimental Results

Custom implementation using FMI 2.0
(not according to standard)





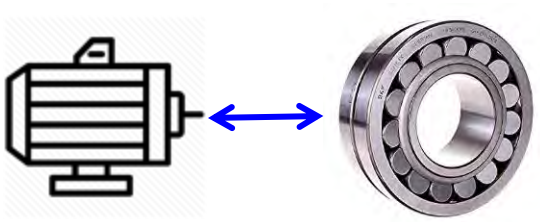
Both models stable with large communcation steps!

Conclusions

- FMI 3.0 will (most likely) support TLM
  - Intermediate variable access
  - Eliminates delays and sampling errors
- Prototype
  - Method works well with CVODE solver
  - Minimum implementation effort

Remaining Work

- Implementation in FMI **export tool(s)**
  - OpenModelica…
- Implementation in FMI **master simulation tool**
  - OMSimulator
- Industrially relevant **demonstrator**



Thank you!

www.liu.se

LINKÖPING UNIVERSITY