An Open-Source Framework for Efficient Co-simulation of Fluid Power Systems

Robert Braun[1], Adeel Asghar[2], Adrian Pop[2], and Dag Fritzson[3]

[1]Division of Fluid and Mechatronic Systems, Dept. of Management and Engineering, Linköping University, Sweden
[2]PELAB - Programming Environment Lab, Dept. of Computer Science, Linköping University, Sweden
[3]SKF Group Technology, AB SKF, Göteborg, Sweden
E-mail: robert.braun@liu.se, adeel.asghar@liu.se, adrian.pop@liu.se, dag.fritzson@skf.com

Abstract

Simulation of fluid power systems typically requires models from multiple disciplines. Achieving accurate load dynamics for a system with complex geometry, for example, may require both a 1D model of the hydraulic circuit and a 3D multi-body model. However, most simulation tools are limited to a single discipline. A solution to these kinds of problems is co-simulation, where different tools are coupled and simulated together. Co-simulation can provide increased accuracy, improved modularity and facilitated collaboration between different organizations. Unfortunately, tool coupling typically requires tedious and error-prone manual work. It may also introduce numerical problems. For these reasons, co-simulation is often avoided as long as possible. These problems have been addressed by the development of an open-source framework for asynchronous co-simulation. Simulation tools can be interconnected through a stand-alone master simulation tool. An extensive range of tools is also supported via the Functional Mockup Interface standard. A graphical user interface has been implemented in the OpenModelica Connection Editor. System models can be created and edited from both a schematic view and a 3D view. Numerical robustness is enforced by the use of transmission line modelling. A minimalistic programming interface consisting of only two functions is used. An example model consisting of a hydraulic crane with two arms, two actuators and a hanging load is used to verify the framework. The composite model consists of nine multi-body models, one hydraulic system model and a controller. It is shown that models from various simulation tools can be replaced with a minimal amount of user input.

Keywords: Co-simulation, system simulation, multi-body simulation, transmission line modelling

1 Introduction

Fluid power systems often interact with other domains, such as mechanical systems, combustion engines, electric power systems or controllers. Meanwhile, most simulation tools are specialized on a specific domain or discipline. Co-simulation makes it possible to couple different tools into a composite model. In this way the best suited tool can be used for each part of the system. It can also facilitate cooperation between or within organizations by allowing each part to keep working in their favorite tool. Such an approach preserves investments in simulation tools, existing models and trained staff. Another benefit is the modularity. If two models share a common connection interface, they can easily be replaced. Increased modularity is for example useful when working with models of different fidelity.

However, setting up a co-simulation can be time consuming, error-prone and require special knowledge. There is also a risk for lock-in effects when using a commercial master simulation tool. As a consequence, tool coupling is often avoided. This paper presents an open source framework for asynchronous co-simulation. The aims of this work include extensive compatibility with other tools, a user-friendly interface and a minimalistic coupling interface. Support for the Functional Mock-up Interface (FMI) standard has been implemented for increased tool compatibility. The framework is demonstrated with example models representing a hydraulic crane and an engine connected to a pump.

2 Related work

Several master algorithms based on the FMI standard have been proposed. A simple master was developed by [1]. More advanced algorithms based on High-Level Architecture (HLA), a standard for distributed computer simulation systems, exists [2] [3] [4]. A master with adaptive communication step-size control was proposed by [5]. An approach using relaxation techniques was presented in [6]. While the two latter methods addressed the issue of numerical stability, they

both rely on rejecting simulation steps, which is not supported by all exporting tools. This paper differs in that it focuses on decoupling at the model level using TLM. In this way, numerical stability is always guaranteed. Previously, TLM-based co-simulation using FMI has been investigated for synchronous coupling of Modelica models [7] and for connection a hydraulic models in Hopsan with multi-body mechanical models in Adams [8].

3 Transmission Line Modeling

When decoupling a model into smaller parts, introducing a time delay is inevitable. Such delays may affect numerical stability unless handled properly. This is addressed by using the transmission line modeling (TLM) method [9] [10]. In reality, every physical element has a natural time delay. Mechanical wave propagation through an element is determined by equations (1) and (2).

$$F_1(t) = F_2(t-\Delta t) + Z_c\,[v_1(t) + v_2(t-\Delta t)] \quad (1)$$
$$F_2(t) = F_1(t-\Delta t) + Z_c\,[v_2(t) + v_1(t-\Delta t)] \quad (2)$$

F is the force, v the velocity, Δt the time delay and Z_c the characteristic impedance of the element. These equations are based on the assumption that friction can be neglected. A consequence of this is that resonances can occur at high frequencies. This can be addressed by filtering the delayed variables, according to equations (3) and (4) [11]. The filter variable α represents an approximated friction model.

$$F_1(t) = v_1(t) + (1-\alpha)c_1(t) + \alpha c_1(t-\Delta t) \quad (3)$$
$$F_2(t) = v_2(t) + (1-\alpha)c_2(t) + \alpha c_2(t-\Delta t) \quad (4)$$

Where:

$$c_1(t) = Z_c v_2(t-\Delta t) + F_2(t-\Delta t)$$
$$c_2(t) = Z_c v_1(t-\Delta t) + F_1(t-\Delta t)$$
$$0 \leq \alpha < 1$$

Force and velocity can be replaced by variables from other physical domains, for example pressure and flow in hydraulic systems. As can be seen, side 1 is always independent on side 2 at the same point of time and vice versa. In this way the numerical time delay introduced by the tool coupling can be replaced by a physically motivated time delay in the model equations. Hence, numerical robustness will not be affected, which makes the method suitable for co-simulation [12] [7]. Besides, TLM can also be used for parallel simulation [13] [14] [15] and detailed modeling of wave propagation [16] [17]

The use of TLM implies limiting the maximum step size of the simulation. For this reason, it is best suited for models where the step size is generally smaller than the physical time delays. It is also desirable that the model has parts with large capacitance (or compressibility) and small inductance (or inertia). Both these requirements are usually fulfilled by fluid system models.

4 Functional Mock-up Interface

The Functional Mock-up Interface (FMI) is a standardized interface for communication between simulation tools [18]. It was launched in 2009, and is currently supported by more than 90 simulation tools. There are two versions of FMI: FMI for co-simulation (FMI CS) and FMI for model exchange (FMI ME). With co-simulation, each slave solves its own equations and exchanges data on pre-defined communication points. With model exchange, the master tool handles the numerical solver, and the slave contains only the model equations. Models are exported as Functional Mock-up Units (FMU), a zip package with the file extension FMU. This package contains an XML description file and either pre-compiled binaries or source code with the actual implementation.

5 Co-simulation Framework

The co-simulation framework was originally developed by SKF in cooperation with Linköping University. It was mainly intended for simulation of bearings coupled to models in external tools [12]. The source code has been donated to the Open Source Modelica Consortium. Models from different simulation tools, called *external models*, are connected to a *composite model*. The external models communicate through network sockets, as shown in figure 1. All messages are sent through a co-simulation *manager process*, which forwards each message from the sender to the receiver. A *monitor process* is executed in parallel, and is used for logging simulation results.

External models communicate asynchronously using two functions: `getForce()` and `setMotion()`, see listing 1. Whenever a step is complete, result data is sent to the connected models. Variables are stored in interpolation tables, and can be interpolated for a requested time instance. This makes it possible for each external model to use its own local solver with an independent time step. The only requirement is that the maximum time step is limited to half the TLM time delay, in order to avoid extrapolation [19].

Listing 1: External models communicate with the master using two function.

```
void GetForce3D(int interfaceID,
                double time,
                double position[],
                double orientation[],
                double speed[],
                double ang_speed[],
                double *force);

void SetMotion3D(int interfaceID,
                 double time,
                 double position[],
                 double orientation[],
                 double speed[],
                 double ang_speed[]);
```

Perhaps the most important property of a co-simulation framework is compatibility with as many simulation tools as possible. For this reason, support for FMI been introduced. In addition, direct tool coupling is available for several tools, including SKF BEAST, Simulink, Adams, Hopsan, OpenModelica, Dymola and Wolfram SystemModeler.

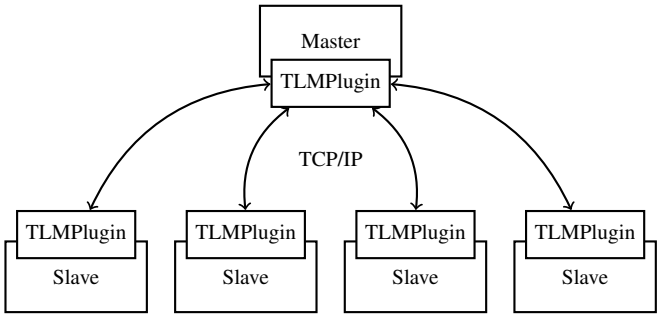

Figure 1: External models communicate with the master using network sockets.

6 Graphical User Interface

A graphical user interface [20] for the framework has been implemented as an extension to the OpenModelica Connection Editor (OMEdit) [21]. It contains an XML editor, a 2D diagram view and a 3D view. The 2D diagram view is shown in figure 2. Sub-models can be added, connected and aligned. Interface information can be fetched from each external model. A plotting tool and 3D animation can be used to analyze results after a simulation.

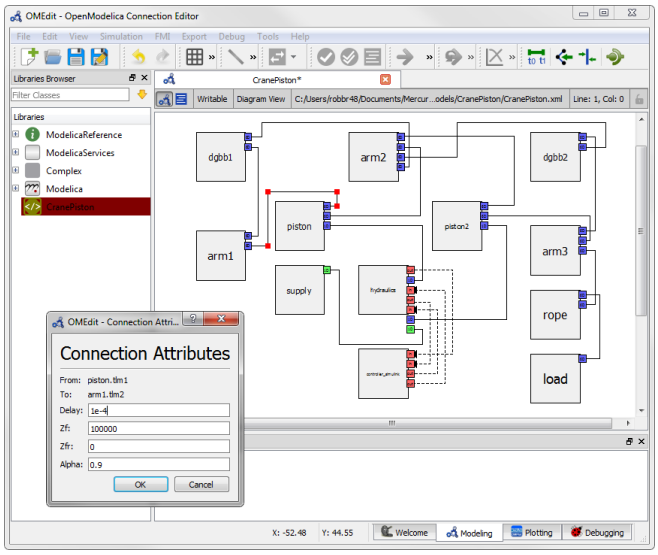

Figure 2: The graphical user interface is implemented in the OpenModelica Connection Editor.

7 Tool Description

The following tools have been used in the simulations:

OpenModelica is an open source simulation tool using the Modelica language developed at Linköping University [22].

BEAST (BEaring Simulation Tool) is a multi-body simulation tool developed by SKF [23] [24].

Hopsan is a system simulation tool mainly focused on fluid power developed at Linköping University [25] [26].

Dymola is a commercial simulation tool based on the Modelica language, developed by Dassault Systèmes [27].

Simulink is a graphical modeling and simulation tool by Mathworks, integrated with the Matlab environment [28].

Adams is a multi-body dynamics simulation tool by MSC [29].

8 Example Model 1: Hydraulic Crane

Co-simulation is often used for interfacing 1D system models with 3D multi-body mechanical models. It is also common to have the controller in a separate tool. For this reason, a model of a hydraulic crane is used to demonstrate the features of the framework, see figure 3. Two pistons are used to move two arms from a lower position (solid) to an upper position (dashed).

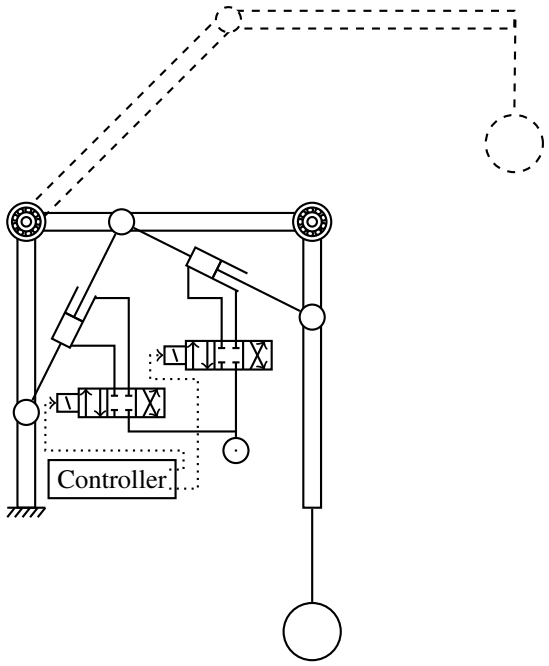

Figure 3: A composite model of a hydraulic crane controlled by two hydraulic pistons.

The crane is lifting a mass load, hanging from a stretched sling. A constant pressure hydraulic system model is driving the pistons. Two bearings are used at the joints between the arms. The motion is controlled by two position feedback controllers, one for each piston. Figure 5 shows the different parts of the composite model. Both 1D and 3D TLM connections are used, as well as directional signals. The latter are simply delayed variables, send from one external model to another.

The composite model is defined by an XML file. Two different configurations, using different external tools, are tested, see table 1. All tools mentioned above, as well as FMI CS and FMI ME, are used. This confirms the modularity of the

approach. If two tools support FMI export, models from the tools can easily be interchanged, assuming they have the same interface ports. This is performed either by modifying XML tags for model file and start script. It can also be done directly from within the graphical editor. It is not necessary to modify the TLM connection settings (time delay and characteristic impedance) when changing configurations, since these represent physical properties of the system which are independent of the surrounding sub-models. For example, the simplified controller model can be replaced with the real control code for software-in-the-loop testing. Models of individual components can also be replaced with models provided by vendors. Furthermore, different hydraulic system concepts can be analyzed and compared without changing the other parts of the complete system.

Simulation results are shown in figure 6. The two different configurations are shown as solid bright lines and dashed lines, respectively. The green lines show the positions of the center of gravity of the mass loads. As can be seen, the two configurations coincide almost perfectly with each other. This was expected, since both configurations have the same level of model fidelity. Oscillations occur due to a fast motion in combination to poor dynamic properties in the system. The model is not optimized or validated, and results are only intended to verify the feasibility of the method. Results can also be visualized in a 3D animation, as shown in figure 4.

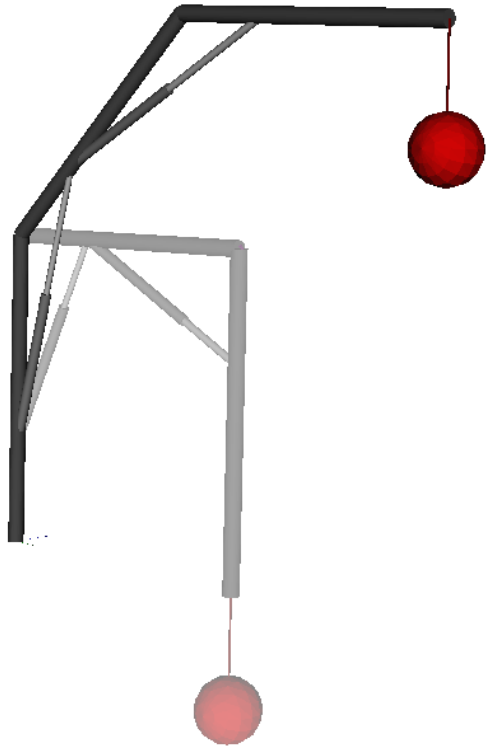

Figure 4: A 3D visualization of the upper and lower position of the crane.

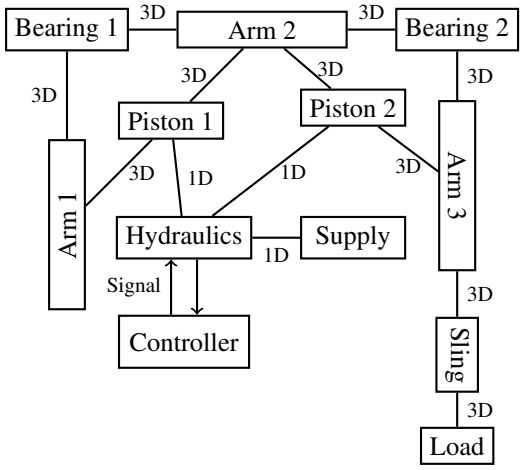

Figure 5: A schematic view of the different external models used in the hydraulic crane model.

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

Table 1: Two different configurations, using different tools, are used for the crane model.

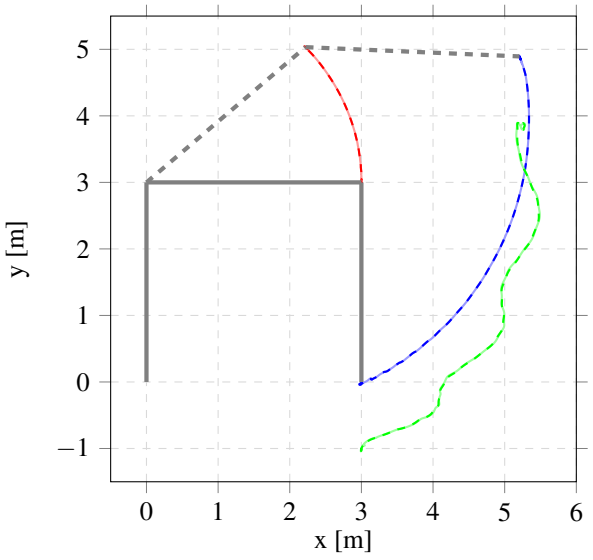

Figure 6: Simulation results for the crane model. The green line represents the position of the load.

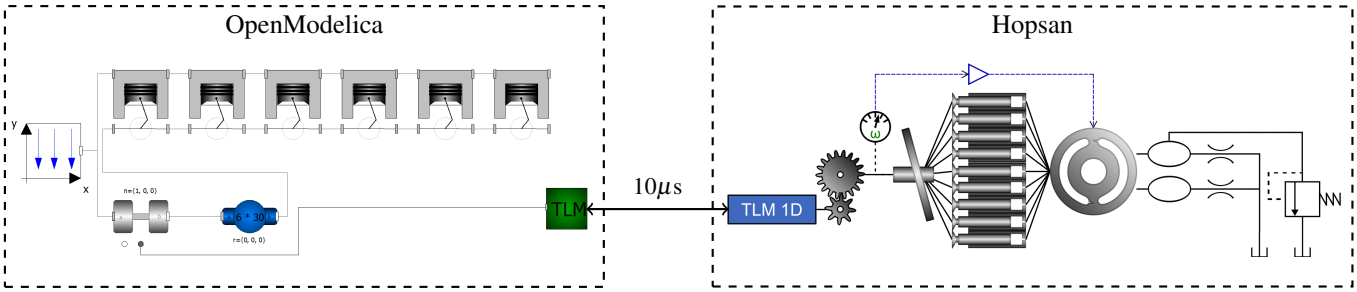

Figure 7: A combustion engine model in OpenModelica connected to a model of an in-line axial piston pump in Hopsan.

9 Example Model 2: Engine and Inline Axial-Piston Machine

As a second example, a model of a combustion engine is connected to a model of an in-line axial piston pump, see figure 7. The XML file describing the model is shown in listing 2. This makes it possible to study the interaction between the engine and the pump, without using simplified models. The engine model is an existing model from the Modelica Standard Library, and exported to FMI using OpenModelica. The pump model is a low-level system simulation model created in Hopsan. A swash plate is connected to nine pistons, which in turn are connected to a valve plate. The pressurized system is modeled using a laminar orifice and a pressure relief valve, both connected to tank. A detailed view of flow pulsations and pressure pulsations at the high-pressure side of the pump are shown in figure 8.

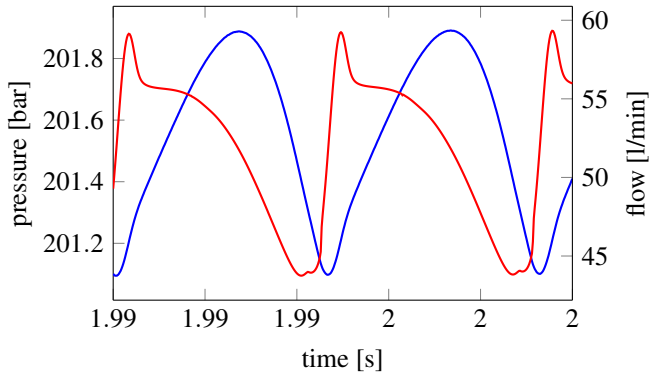

Figure 8: Pressure pulsations (blue) and flow pulsations (red) from the second example model.

Due to the modular model structure, the minimalistic interface and support for FMI, the two models can easily be replaced with more detailed ones. The pump could for example be replaced with a CFD model, and the engine with a model from a specialized engine simulation tool. Advantages of coupling 1D system models with 3D CFD models were discussed in [30]. Furthermore, the laminar orifice, representing the load, could be replaced with a more realistic model.

10 Example Model 3: Full Vehicle Model with Semi-active Suspension

In a previous experiment, a co-simulation model of semi-active hydraulic damping in a forwarder was developed in [8]. A hydraulic system in Hopsan was connected to a full-vehicle model in Adams. The main purpose was to minimize soil damage by maintaining an equal distribution of ground forces among the wheels. Tool coupling was achieved by exporting the Hopsan model as an FMU, which was imported in Adams. Adams could then read the TLM variables and convert them to forces using the TLM equations implemented as custom functions. While this experiment was successful, the method involves some drawbacks. Most importantly, the Adams model must be customized to support TLM.

By instead using the framework described in this paper, the TLM equations would be computed in the master algorithm. In this way, both the Adams and the Hopsan models could rely on general callback functions for sending velocity and receiving force. This would also make it possible for Adams to use variable step-size and implicit solvers without compromising numerical accuracy. Both tools support FMI and can be exported to the framework as FMUs. There are also direct connections available for both Hopsan and Adams, which enable more features and can reduce overhead.

11 Challenges

While the method proves to be stable and modular, there are some remaining research challenges. First, the FMI standard for co-simulation does not currently support asynchronous communication with delayed variables. Instead, input variables are assumed to be constant between communication steps (constant extrapolation). There is some support for extrapolation using time derivatives of input variables, but this is not supported by many exporting tools. Also, it does not provide guaranteed stability. An alternative solution could be adaptive step-size control with error estimation. However, this requires rollback mechanisms. Even though rollback is supported by FMI, it is optional for exporting tools to support it. Consequently, many tools are not able to do this. Hence, solving the problem completely is only possible by extending the FMI standard.

Another issue with the current framework is directional signal connections. Since there is a time delay in every connection, signals will also be delayed. Even though signals has

Listing 2: The XML file defining the second example model.

```
<Model Name="EnginePump">
  <SubModels>
    <SubModel Name="engine"
              ModelFile="engine.mo"
              StartCommand="StartTLMOpenModelica"
              Position="0,0,0"
              Angle321="0,0,0"  >
      <InterfacePoint Name="tlm"
                      Domain="Rotational"
                      Causality="Bidirectional"
                      Dimensions="1"
                      Position="0,0,0"
                      Angle321="0,0,0"  />
    </SubModel>
    <SubModel Name="pump"
              ModelFile="pump.hmf"
              StartCommand="StartTLMHopsan"
              Position="0,0,0"
              Angle321="0,0,0">
      <InterfacePoint Name="tlm"
                      Domain="Mechanical"
                      Causality="Bidirectional"
                      Dimensions="1"
                      Position="0,0,0"
                      Angle321="0,0,0" />
    </SubModel>
  </SubModels>

  <Connections>
    <Connection To="pump.tlm"
                From="engine.tlm"
                Delay="1e-5"
                Zfr="10"
                alpha="0.9"/>
  </Connections>

  <SimulationParams StopTime="2" StartTime="0"/>
</Model>
```

a natural physically motivated time delay, the magnitude is very small in for example control loops. Such small delays would impose severe limitations to simulation performance. Using larger delays, on the other hand, may affect accuracy, for example by increasing phase shift in control loops. A possible solution can be to use a wrapper with a local master algorithm for each group of signal models, which handles sorting, scheduling and stepping of these models.

Finally, the limitation on the step size can be problematic even for TLM connections. Some simulations, as for example vehicle drive cycles, requires variable step size for achieving good performance. Fixed limitations on maximum step size may then not be feasible. A possible solution could be to allow variable TLM time delays. Adaptive time-stepping in TLM models has previously been investigated [31]. Numerical stability will not be affected by a larger time delay. Instead, it will induce a modeling error in the form of a parasitic inductance. At steady-state, however, inductance will not have any effect. Hence, increasing the delays can be acceptable in certain situations.

12 Conclusions

Coupling different simulation tools is important for the fluid power domain. Hydraulic systems are typically used to transfer power from one mechanical system to another. Complex mechanics require specialized modeling and simulation tools, which are usually not suitable for simulating hydraulic circuits. Furthermore, the hydraulic system is often connected to some control software. Other simulation domains which could be of interest, although not discussed in this paper, include computational fluid dynamics and finite element analysis.

By using TLM for decoupling, numerical time delays can be avoided. Hence, there is no risk for numerical errors or instability. TLM is suitable for models which contains large time delays in comparison to the time steps required for the simulation. This includes, for example, hydraulic circuits, long electric transmission lines or mechanical models with strong dynamics. Besides, TLM can also be used for simulating wave propagation with good accuracy.

An open-source master simulation tool provides a vendor neutral co-simulation platform, without the risk for lock-in effects. Support for the FMI standard enables compatibility with a large number of simulation tools. Attaching new tools to the framework is facilitated by a minimalistic coupling interface. The example models with different configurations confirms good compatibility and modularity. Future work includes support for event handling, improved support for FMI and testing on industrial applications.

References

[1] Jens Bastian, Christoph Clauß, Susann Wolf, and Peter Schneider. Master for co-simulation using FMI. In *8th International Modelica Conference, Dresden*. Citeseer, 2011.

[2] Atiyah Elsheikh, Muhammed Usman Awais, Edmund Widl, and Peter Palensky. Modelica-enabled rapid prototyping of cyber-physical energy systems via the functional mockup interface. In *Modeling and Simulation of Cyber-Physical Energy Systems (MSCPES), 2013 Workshop on*, pages 1–6. IEEE, 2013.

[3] Muhammad Usman Awais, Peter Palensky, Wolfgang Mueller, Edmund Widl, and Atiyah Elsheikh. Distributed hybrid simulation using the HLA and the functional mock-up interface. *Industrial Electronics Society, IECON*, pages 7564–7569, 2013.

[4] Himanshu Neema, Jesse Gohl, Zsolt Lattmann, Janos Sztipanovits, Gabor Karsai, Sandeep Neema, Ted Bapty, John Batteh, Hubertus Tummescheit, and Chandraseka Sureshkumar. Model-based integration platform for fmi co-simulation and heterogeneous simulations of cyber-physical systems. In *Proceedings of the 10 th International Modelica Conference; March 10-12; 2014; Lund; Sweden*, number 096, pages 235–245. Linköping University Electronic Press, 2014.

[5] Tom Schierz, Martin Arnold, and Christoph Clauß. Co-simulation with communication step size control in an FMI compatible master algorithmnak. In *9th Int. Modelica Conf., Munich, Germany*, pages 205–214, 2012.

[6] Bernhard Schweizer, Daixing Lu, and Pu Li. Co-simulation method for solver coupling with algebraic constraints incorporating relaxation techniques. *Multibody System Dynamics*, 36(1):1–36, 2016.

[7] Robert Braun and Petter Krus. Tool-independent distributed simulations using transmission line elements and the Functional Mock-up Interface. October 2013.

[8] Robert Braun, Liselott Ericsson, and Petter Krus. Full vehicle simulation of forwarder with semi active suspension using co-simulation. In *ASME/BATH 2015 Symposium on Fluid Power and Motion Control*, October 2015.

[9] Donard De Cogan, William J O'Connor, and Susan Pulko. *Transmission line matrix (TLM) in computational mechanics*. CRC press, 2005.

[10] Petter Krus. Robust modelling using bi-lateral delay lines for real time and faster than real time system simulation. In *ASME 2009 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, pages 131–138. American Society of Mechanical Engineers, 2009.

[11] P. Krus, A. Jansson, J-O. Palmberg, and K. Weddfelt. Distributed simulation of hydromechanical systems. In *The Third Bath International Fluid Power Workshop*, Bath, England, 1990.

[12] Alexander Siemers, Dag Fritzson, and Iakov Nakhimovski. General meta-model based co-simulations applied to mechanical systems. *Simulation Modelling Practice And Theory*, 17(4):612–624, 2009.

[13] Kaj Nyström and Peter Fritzson. Parallel simulation with transmission lines in Modelica. In *5th International Modelica Conference*, Vienna, Austria, September 2006.

[14] Martin Sjölund, Mahder Gebremedhin, and Peter Fritzson. Parallelizing equation-based models for simulation on multi-core platforms by utilizing model structure. In Alain Darte, editor, *Proceedings of the 17th Workshop on Compilers for Parallel Computing*, Lyon, France, July 2013.

[15] Robert Braun and Petter Krus. Multi-threaded distributed system simulations using the transmission line element method. *SIMULATION*, 92(10):921–930, October 2016.

[16] P. Krus, K. Weddfelt, and J-O. Palmberg. Fast pipeline models for simulation of hydraulic systems. *Journal Of Dynamic Systems Measurement And Control*, 116:132–136, 1994.

[17] Nigel Johnston. The transmission line method for modelling laminar flow of liquid in pipelines. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 226(5):586–597, 2012.

[18] T. Blochwitz, M. Otter, M. Arnold, C. Bausch, C. Clauß, H. Elmqvist, A. Junghanns, J. Mauss, M. Monteiro, T. Neidhold, D. Neumerkel, H. Olsson, J.-V. Peetz, and S. Wolf. The Functional Mockup Interface for tool independent exchange of simulation models. In *8th International Modelica Conference 2011*, Como, Italy, September 2009.

[19] Iakov Nakhimovski. *Contributions to the Modeling and Simulation of Mechanical Systems with Detailed Contact Analyses*. PhD thesis, Linköping University, PELAB - Programming Environment Laboratory, The Institute of Technology, 2006.

[20] Alachew Mengist, Adeel Asghar, Adrian Pop, Peter Fritzson, Willi Braun, Alexander Siemers, and Dag Fritzson. An open-source graphical composite modeling editor and simulation tool based on FMi and TLM co-simulation. In Peter Fritzson and Hilding Elmqvist, editors, *Proceedings of the 11th International Modelica Conference*. Modelica Association and Linköping University Electronic Press, September 2015.

[21] Syed Adeel Asghar, Sonia Tariq, Mohsen Torabzadeh-Tari, Peter Fritzson, Adrian Pop, Martin Sjölund, Parham Vasaiely, and Wladimir Schamai. An open source Modelica graphic editor integrated with electronic notebooks and interactive simulation. In Christoph Clauß, editor, *Proceedings of the 8th International Modelica Conference*. Linköping University Electronic Press, March 2011.

[22] Peter Fritzson, Peter Aronsson, Håkan Lundvall, Kaj Nyström, Adrian Pop, Levon Saldamli, and David Broman. The openmodelica modeling, simulation, and software development environment. *Simulation News Europe*, 44:8–16, 2005.

[23] Lars Erik Stacke, Dag Fritzson, and Patrik Nordling. Beast—a rolling bearing simulation tool. *Proceedings of the Institution of Mechanical Engineers, Part K: Journal of Multi-body Dynamics*, 213(2):63–71, 1999.

[24] Dag Fritzson, Lars-Erik Stacke, and Jens Anders. Dynamic simulation–building knowledge in product development. *Evolution*, 1, 2014.

[25] Mikael Axin, Robert Braun, Alessandro Dell'Amico, Björn Eriksson, Peter Nordin, Karl Pettersson, Ingo Staack, and Petter Krus. Next generation simulation software using transmission line elements. In *Fluid Power and Motion Control*, Bath, England, September 2010.

[26] B. Eriksson, P. Nordin, and P. Krus. Hopsan NG, a C++ implementation using the TLM simulation technique.

In *The 51st Conference On Simulation And Modelling*, Oulu, Finland, 2010.

[27] Modelon. Dymola. http://www.modelon.com/products/dymola/. Accessed 2015-10-07.

[28] The Mathworks, Inc. Simulink - Simulation and Model-Based Design. https://se.mathworks.com/products/simulink/. Accessed 2016-11-22.

[29] RR Ryan. ADAMS - multibody system analysis software. In *Multibody Systems Handbook*, pages 361–402. Springer, 1990.

[30] J. Galindo, A. Tiseira, P. Fajardo, and R. Navarro. Coupling methodology of 1d finite difference and 3d finite volume {CFD} codes based on the method of characteristics. *Mathematical and Computer Modelling*, 54(7-8):1738 – 1746, 2011.

[31] S. H. Pulko, A. Mallik, R. Allen, and P. B. Johns. Automatic timestepping in TLM routines for the modelling of thermal diffusion processes. *International Journal of Numerical Modelling: Electronic Networks, Devices and Fields*, 3(2):127–136, 1990.