Modeling, Identification and Control, Vol. 39, No. 3, 2018, pp. 179–190, ISSN 1890–1328

Composite modelling in 3-D mechanics utilizing Transmission Line Modelling (TLM) and Functional Mock-up Interface (FMI)

Dag Fritzson[1] Robert Braun[2] Jan Hartford[1]

[1] *AB SKF, SE-415 50 Göteborg, Sweden. E-mail:* dag.fritzson@skf.com

[2] *Linköping University, SE-581 83 Linköping, Sweden. E-mail:* robert.braun@liu.se

Abstract

Composite modelling and simulation is a solution to utilize investments in models and tools, use the right tool for the right task, increase the accuracy by means of more accurate modelled boundary conditions, switch between levels in model complexity for a specific sub-system, and facilitate co-operation in organizations. With the new Functional Mock-up Interface (FMI) standardization, efforts are increasing to make this happen. SKF BEAST is an advanced dynamic simulation tool for rolling bearings and other mechanical systems with contacts. The tool incorporates a framework for composite modelling and co-simulation, i.e., a Master Simulation Tool (MST). It uses Transmission Line Modelling (TLM) to ensure robust numerical behaviour of the complete composite system model and supports the Functional Mock-up Interface (FMI) for model import, including both model exchange and co-simulation. In this paper, the tools and the techniques for composite modelling are discussed in further detail and application examples are given.

Keywords: Master Simulation Tool (MST), Functional Mockup Interface (FMI), Transmission Line Modelling (TLM), co-simulation, composite modelling, BEAST

1 Introduction

The need for composite modelling/simulation or co-simulation arises from several different aspects.

First of all, different simulation tools are popular within different organizations. Some tools may also be commonly used within certain technical disciplines or engineering sectors. It is not uncommon that a specific simulation tool becomes a de facto standard within a specific field, and there is a large investment in existing models and trained staff. Connecting different tools with each other can facilitate cooperation between organizations and departments.

Another reason for connecting multiple simulation tools is that different tools may be suitable for different parts of the model. Co-simulation makes it possible to simulate larger systems, and to analyze how different subsystems interact with each other.

It can be desirable to switch between models of different complexity, and thus computation cost, for a specific sub-system.

Finally, it can also improve accuracy of the results by allowing more realistic boundary conditions by means of simulating a larger system. This is especially suitable for simulation of rolling bearings, where realistic loading and motion are of great importance.

Tool coupling is often time consuming and requires highly specialized knowledge. For this reason, it is often avoided and used only when absolutely necessary. Hence, it is important to minimize the workload and knowledge requirements in order to maximize the benefits. This requires a minimalistic coupling interface and an intuitive user interface.

Continuous-time simulation puts high demands on numerical robustness. If numerical stability cannot be guaranteed, simulation results cannot be trusted. The co-simulation framework must be able to simulate numerically stiff models without stability problems, or manual tuning of model parameters or solver settings.

There is an evolving standard, Functional Mock-up Interface (FMI) (Functional Mock-up Interface (FMI)), for encapsulation of models/tools. It specifies low-level interfaces for connecting simulation tools. It does not cover the system specific issues, i.e., master simulation tool, or composite model building/editing. In order to provide maximum connectivity it is important to support FMI, while still providing easy-to-use composite modelling and also ensure simulation accuracy.

Thus the main challenges in composite modelling and simulation are:

- robust and numerical stable simulation,
- parallel simulation of the complete composite model,
- support of FMI,
- modelling on appropriate abstraction level, e.g., encapsulating low level details of FMI, and
- easy-to-use tool for building composite models.

In the following sections the composite modelling tool and the underlying technology is discussed.

Examples of industrial use-cases and demonstrators are given. They motivate the research and prove that it works in real applications.

2 Related work – FMI/TLM

Several researchers have investigated master algorithms for co-simulation using FMI.

A basic master algorithm was proposed by (Bastian et al., 2011). Master algorithms based on High-Level Architecture (HLA) have also been investigated (Elsheikh et al., 2013; Awais et al., 2013; Neema et al., 2014). All these implementations provides successful results for their respective test models. However, neither of them attempts to address the issues of numerical stability.

In (Schierz et al., 2012), a master algorithm for FMI using communication step size control was presented. Error estimation was used to adapt the communication step, which provided significant improvements to stability and performance. Unlike the work presented in this paper, this approach relies on rollback mechanisms. FMUs must be able to save and restore states to previous communication points. This is often not possible, due to limitations in the tools from where the FMUs are exported.

A method for solver coupling using a predictor-corrector method and relaxation techniques was presented in (Schweizer et al., 2016). In this way, stable co-simulation could be achieved for higher-order extrapolation of coupling variables. However, the corrector step requires the FMUs to support solver rollback to previous states.

The Transmission Line Modelling (TLM) method enables numerically stable couplings using fixed-size communication delays (Johns and O'Brien, 1980; Auslander, 1968). Co-simulation using FMI and TLM with synchronous communication has been investigated for Modelica (Modelica and the Modelica Association) models (Braun and Krus, 2013) and for connections between Hopsan (Hopsan project) and ADAMS (MSC Adams; Braun et al., 2015). These experiments confirmed the feasibility of the concept, but does not provide a general platform for FMI co-simulation.

2.1 Discussion of related work

Roughly, there are two subgroups of master simulation tools. One that focuses mostly on the modelling, computation, and communication aspects. The issues of numerical stability are not adressed, and left for the user to solve.

The other group tries to handle tightly coupled models as well. They typically rely on the FMI functionality for rejecting steps and supply of Jacobians. A drawback is that the external simulation tools must support rejecting steps, which is not always the case. The same can be said for Jacobians, which also can be costly to compute.

Unlike these methods, the work in this paper is focused on decoupling at the modeling level using TLM. Hence, time delays are inserted directly into the model equations and no numerical decoupling algorithms are required.

The asynchronous transmission line modelling technique is employed to simulate the composite model in a completely robust and parallel way.

FMI is supported by treating it just as an additional generic simulation tool type. A wrapper is used to encapsulate and translate to the low level details of the FMI model interface.

To provide easy to use tool support to build composite models, a proven 3-D model editor is extended to support FMI external models and mixed models.

3 Composite model editor and master simulation tool

A co-simulation model is here denoted *composite model*, which consists of several interconnected *external models*, simulated independently in *external tools*.

To study and predict dynamic phenomena, one needs to use a dynamic simulation tool. One such tool is BEAST (Fritzson et al., 2014), which is a 3-D multi-body simulation tool specialized in detailed contact calculations, making it a very efficient tool for rolling bearings and other machine elements where contacts are important.

The tool simultaneously solves the dynamics of the multi-body system, the structural deformations, the thermal balance, and the local lubricated contact conditions. The tool is used for optimizing the product design, for evaluating performance under various application conditions, and for advanced damage and failure analysis.

The composite model editor utilized here is the 3-D model editor in the tool set, which can handle external models as well, i.e., it is a master simulation tool editor. It enables the user to build from scratch, import, connect, and combine models from a pre-defined library of basic models and components, including general CAD bodies, and external models in various formats including FMI. Existing components can be copied or exported for later re-use.

To represent and store the model, a special purpose language representation with a strict notation and grammar is used. It includes basic classes for:

- *coordinate systems*, fixed or describing motion as function of time,
- *bodies*, having *surfaces* and *coordinate systems*,
- *connections* between bodies, having *contacts* between surfaces, and *ties* (e.g., stiffness/damping matrices) between coordinate systems,
- *models*, including coordinate systems, bodies, and other models (enabling a hierarchy of models),
- *external models*, i.e., imported encapsulated models from other tools with *coordinate system* interfaces. This includes direct tool connection support for Matlab/Simulink (MathWorks Simulink), Adams (MSC Adams), BEAST, and Modelica (Modelica and the Modelica Association) tools; Wolfram SystemModeler (Wolfram SystemModeler), Dymola (Dassault Systémes Dymola), and OpenModelica (Open Source Modelica Consortium (OSMC)).

  For external models in FMI 2.0 (Functional Mock-up Interface (FMI)) format, both variants of FMI, i.e., model exchange and co-simulation, are supported since December 2016.

Alternatives based on XML or Modelica have been investigated, but a small dedicated language was preferred in the end (Siemers, 2010).

Among the *ties*, there is a 3-D mechanical bi-directional TLM connector type. Thus, a single high level connector type encapsulates 24 scalar variables describing the motion and the loading in the tie.

A model with only external models is defined as a “pure” composite model, whereas a model with both internal components and external models is defined as a “mixed” model.

User defined parameters can be defined at different levels in the system model, and also as expressions of other parameters. The parameter look-up follows the model hierarchy. Typically user defined parameters, or expressions of those, are used for all inputs. This simplifies model building, makes re-use of model components easy, facilitates use of standardized design parameters, and prepares for parametric studies.

If the model created contains any *external model*, then a master simulation tool is started from the simulation environment. Simulation of all the external models are executed in parallel on computer system resources specified a priori for each tool.

The tool has composite modelling capacity for a long time (Siemers et al., 2009; Siemers, 2010; Nakhimovski, 2006). The recent additions are the capability for mixed modelling, the support for FMI, and improved direct couplings to Modelica tools including a high precision one-step simulation mode for OpenModelica.

4 Co-simulation framework and transmission line modelling

The co-simulation framework is based on Transmission Line Modelling (TLM) (Johns and O’Brien, 1980), also known as bi-lateral delay line modelling (Auslander, 1968). All physical systems have finite energy wave propagation speed. This introduces a natural time delay in every element in the system. In simulation models, such delays often have small effects on the results and can be neglected. However, this can also be used as a physically motivated decoupling of different parts of the model. If two parts of the model does not depend on each other at the same point in time, they can use independent numerical solvers. This enables co-simulation with different tools (Siemers et al., 2009), but also parallel and distributed simulation (Krus, 2007). Most importantly, it eliminates the need for numerical decoupling methods and will thereby not affect the numerical stability. The fundamental equations for a TLM element are shown in (14) and (15). For mechanical systems, they can be derived from Newton's second law of motion and Hooke's law, see below. Similar equations can be derived for other physical domains, such as electrical circuits (Hui and Christopoulos, 1990) and hydraulic systems (Viersma, 1980).

The TLM equations can be derived from the elementary laws of physics. A wave propagating through a physical element is described by the wave equation and the telegrapher's equations, see (1), (2) and (3). Z_c is the characteristic impedance of the element. For a mechanical connection F is force, v velocity, x position and a speed of sound. These are similar for most physical domains. For mechanics, they can be derived from Netwon's second law of motion and Hooke's law, describing the inertia and flexibility of the element, respectively.

$$\frac{\partial^2 v(t,x)}{\partial t^2} = a^2 \frac{\partial^2 v(t,x)}{\partial x^2}, \tag{1}$$

$$\frac{\partial F(t,x)}{\partial t} = a Z_c \frac{\partial v(t,x)}{\partial x}, \tag{2}$$

$$\frac{\partial F(t,x)}{\partial x} = \frac{Z_c}{a}\frac{\partial v(t,x)}{\partial t} \tag{3}$$

The general solution to (1) in the frequency domain is given by (4):

$$V(s,x) = C_1 e^{\frac{sx}{a}} + C_2 e^{-\frac{sx}{a}}. \tag{4}$$

Applying the solution in (4) on (2) yields (5):

$$F(s,x) = -Z_c\Big[C_1 e^{\frac{sx}{a}} + C_2 e^{-\frac{sx}{a}}\Big]. \tag{5}$$

The unknown variables C_1 and C_2 can be computed using boundary values, where T is the time delay of the element:

$$\begin{cases} F(s,x=0) = F_1(s), \\ V(s,x=0) = V_1(s), \\ F(s,x=aT) = F_2(s), \\ V(s,x=aT) = V_2(s). \end{cases}$$

Inserting the boundary values and rearranging the equations yields (6) and (7).

$$F_2(s) = \frac{e^{Ts}+e^{-Ts}}{2}F_1(s) - Z_c\frac{e^{Ts}-e^{-Ts}}{2}V_1(s), \tag{6}$$

$$V_2(s) = \frac{e^{Ts}-e^{-Ts}}{2}F_1(s) - Z_c\frac{e^{Ts}+e^{-Ts}}{2}V_1(s) \tag{7}$$

Subtracting (6) from (7) and transforming back to time domains yields (8), and due to symmetry also (9).

$$F_2(t) - Z_c v_2(T) = F_1(t-T) + Z_c v_1(t-T), \tag{8}$$

$$F_1(t) - Z_c v_1(T) = F_2(t-T) + Z_c v_2(t-T) \tag{9}$$

These equations can be further simplified by defining the delayed wave information as *wave variables* c_1 and c_2:

$$c_1(t) = F_2(t-T) + Z_c v_2(t-T), \tag{10}$$

$$c_2(t) = F_1(t-T) + Z_c v_1(t-T). \tag{11}$$

Inserting (10) and (11) into (8) and (9) finally gives the TLM boundary equations:

$$F_1(t) = c_1(t) + Z_c v_1(t), \tag{12}$$

$$F_2(t) = c_2(t) + Z_c v_2(t) \tag{13}$$

which can also be written as

$$F_1(t) = F_2(t-T) + Z_c\left[v_1(t) + v_2(t-T)\right], \tag{14}$$

$$F_2(t) = F_1(t-T) + Z_c\left[v_2(t) + v_1(t-T)\right]. \tag{15}$$

The co-simulation framework uses socket communication between the participating tools, as shown in Figure 1. A master process is responsible for setting up connections and forwarding messages from senders to receivers. All instances, including the master, use a plug-in for socket communication called `TLMPlugin`.

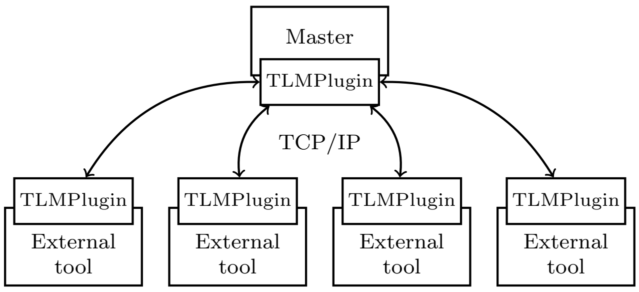


Figure 1: The co-simulation framework consist of interconnected external tools, communicating through a master using TCP/IP sockets.

A minimalistic interface (API) using only two functions is used (Listings 1 and 2) and exposed to the external tools. Communication of data and its protocol is internal to the master simulation tool and is not exposed to the external tools. After each complete time step, a function for sending the motion to the other side of the connection is called, see Listing 1. The motion data is sent to the connected tool, where it is stored in interpolation tables. At any point during the step, the force for a specific time and motion can be interpolated from these tables by calling the second function, see Listing 2.

The mechanical coupling (motion, force) is active simultanously in both directions of the connection.

The interpolation makes it possible to use multi-step and implicit solvers, where input variables need to be

Listing 1: A function for sending motion data to the interconnected external tool.

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

Listing 2: A function for obtaining an interpolated force from the interconnected external tool for given time and motion data.

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

evaluated multiple times during the same step. Moreover, it also makes it possible to use different step sizes in two interconnected tools. Each tool can thus use any desired numerical solvers and step sizes, which can be either fixed or variable. The only limitation is that the step size (h_{model}) must never exceed half the communication delay (h_{TLM}), according to (16) (Nakhimovski, 2006). This ensures that the interpolation tables are always populated past the end time for the next step, so that extrapolation can be avoided.

$$h_{\text{model}} \leq \frac{h_{\text{TLM}}}{2} \tag{16}$$

This enables full decoupling and parallel simulation of the external models/tools, while preserving the numerical stability even for numerical stiff systems.

5 Supporting FMI in the co-simulation framework

Functional Mock-up Interface (FMI) is an open interface for tool-independent exchange of simulation models (Blochwitz et al., 2011), supported by a wide range of tools (Functional Mock-up Interface (FMI)). Models are provided as Functional Mock-up Units (FMU): ZIP archives with the file extension FMU. These contain an XML description schema along with pre-compiled binary files or source code, which can be used by a host environment. A generic wrapper executable is used for importing FMUs to the framework. This wrapper contains the TLMPlugin, and works like a translator between the socket communication and the FMI interface.

There are two kinds of FMUs: *co-simulation* and *model exchange*. With FMI for co-simulation, each FMU contains its own numerical solver and interchanges data with the environment only at pre-defined communication points. With FMI for model exchange, on the other hand, the master provides a numerical solver. Each FMU then only needs to evaluate its derivatives for a given state. This makes it possible to use the same numerical solver for multiple FMUs and avoid numerical instability.

FMI for model exchange works well with the co-simulation framework at hand. Since the solver is located in the wrapper master tool, it can interpolate forces for any states necessary. A drawback is that the external tools must be able to export FMUs for model exchange. Furthermore, it requires a generic solver, while many tools require special solver algorithms. It is for example not possible to use solvers optimized for a certain model structure.

Supporting FMI for co-simulation is more problematic due to potential numerical stability issues. The forces can only be provided to the FMU at the beginning of each step, as shown in Figure 2. Hence, it only works well with explicit one-step solvers. Implicit and multi-step solvers can still be used, but then the forces must be extrapolated (constant or linearly). This has a negative effect on numerical stability, and reduces the benefits of the robust TLM connections.

The current solution chosen for the wrapper master tool is to divide each step into multiple sub-steps, as shown in Figure 3. Forces are kept constant during each sub-step, and updated between them. In this way, the resolution is increased and only shorter extrapolation is required. Pseudo-code is shown in Listing 3. Choosing a sufficiently small length for each sub-step requires simulation experiments for each type of application. Even though numerical properties are improved, numerical stability cannot be guaranteed with this method. A solution could be to monitor the errors and use variable sub-step length. Another approach could be to supply the FMU with a callback function, so that the solver can obtain force variables also during the step. This would, however, require a modification of the FMI standard. These solutions are adressed by ongoing research (Braun et al., 2017).

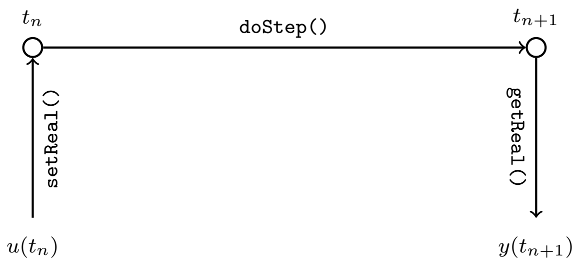


Figure 2: With FMI for co-simulation, input variables can only be provided at the beginning of each step.

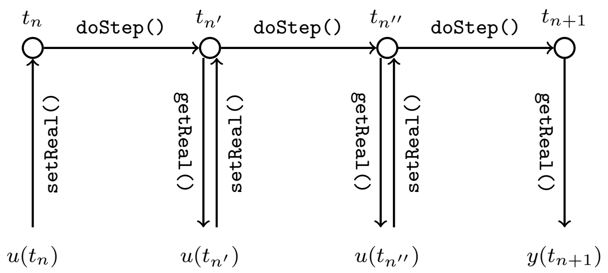


Figure 3: By dividing each step into multiple sub-steps, extrapolation errors can be reduced.

Listing 3: Pseudo-code for dividing the step for an FMU into 100 sub-steps.

```
while(t<tmax) {
  size_t nSubSteps=100;
  for(size_t i=0; i<nSubSteps; ++i) {
    pFmu->setReal(0,pPlugin->getForce(t-dt));
    t+=dt/nSubSteps;
    pFmu->doStep(t);
    pPlugin->setMotion(t,pFmu->getReal(1));
  }
}
```

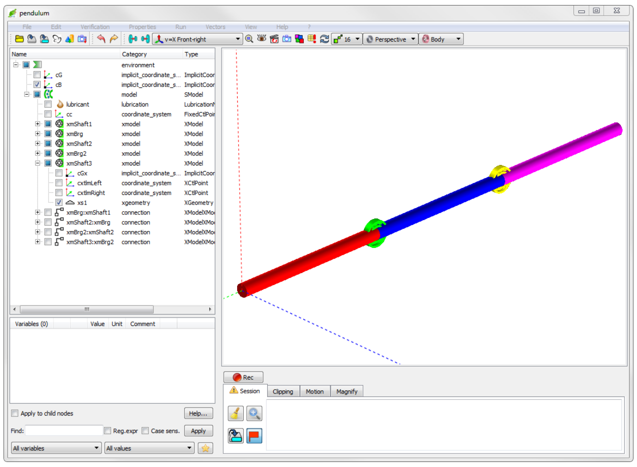
6 Application examples

6.1 Triple pendulum with rolling bearings

A triple pendulum model with two rolling bearings is used to illustrate the multi-tool capability of the composite modelling and co-simulation framework. Here the compatibility is demonstrated by the following connections:

- import of FMI for model exchange,
- import of FMI for co-simulation,
- direct coupling with a Modelica tool,
- direct coupling with BEAST.

The *direct coupling* means interfacing directly to the tool in question without using FMI. Thus the interfacing will vary between tools, but they all involve motion of a point or coordinate system, and loading the same point.



Name	Category	Type
	environment	
cG	implicit_coordinate_s...	ImplicitCoor
cB	implicit_coordinate_s...	ImplicitCoor
	model	SModel
lubricant	lubrication	Lubrication
cc	coordinate_system	FixedCtlPoi
xmShaft1	xmodel	XModel
xmBrg	xmodel	XModel
xmShaft2	xmodel	XModel
xmBrg2	xmodel	XModel
xmShaft3	xmodel	XModel
cGx	implicit_coordinate_s...	ImplicitCoor
cxtlmLeft	coordinate_system	XCtlPoint
cxtlmRight	coordinate_system	XCtlPoint
xs1	xgeometry	XGeometry
xmBrg:xmShaft1	connection	XModelXMo
xmShaft2:xmBrg	connection	XModelXMo
xmBrg2:xmShaft2	connection	XModelXMo
xmShaft3:xmBrg2	connection	XModelXMo

Variables (0)	Value	Unit	Comment

Figure 4: The BEAST composite model editor showing the model of the triple pendulum with rolling bearings.

The pendulum is a classical example in mechanics, and can be difficult to simulate with good numerical accuracy. In this example, three arms are connected through two bearings. At each bearing, the outer ring is attached to the arm above and the inner ring to the arm below. See Figure 4, and Figure 5.

The bearings are modelled in BEAST, and imported through direct tool coupling. The arms can be modelled either in BEAST or in Modelica. Modelica models can be connected to the co-simulation framework through direct tool coupling with OpenModelica, Wolfram SystemModeler, or Dymola. It is also possible to import models from FMI for co-simulation or FMI for model exchange. With FMI for model exchange, the framework supports the CVODE and IDA solvers from the Sundials suite (Hindmarsh et al., 2005). Figure 6 shows one possible configuration of the demonstrator model, where all specified tool couplings are used.

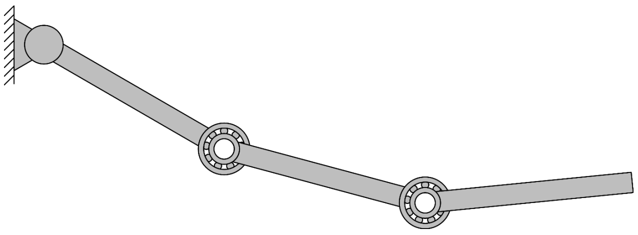


Figure 5: The triple pendulum model with two rolling bearings.

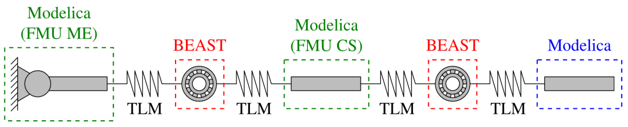


Figure 6: The modelling tools used for the different external models and the couplings between them (FMU ME = FMU for model exchange, FMU CS = FMU for co-simulation, "no notation" indicate direct coupling).

Comparing simulation results for this model without co-simulation, i.e., monolithic, and with co-simulation ($h_{\text{TLM}} = 1 \cdot 10^{-5}$s) shows no practical difference, see Figure 7.

6.2 Wheel end bearing unit in a vehicle model

An early demonstrator for composite modelling was a wheel end bearing unit connected to a complete car model (Nakhimovski, 2006). The purpose was to use the car model, with its driving cycle, as boundary condition to the detailed rolling bearing model, i.e., to get realistic loading and motion.

The car and wheel end bearing unit is modelled in MSC.Adams and BEAST, respectively. With the composite modelling tool the bearing unit was placed and connected to the car model in the right place, see Figure 8 and Figure 9.

The modelling and simulation gave proof of concept and showed the feasibility of the process and tools. For details on the simulation result, see (Nakhimovski, 2006).

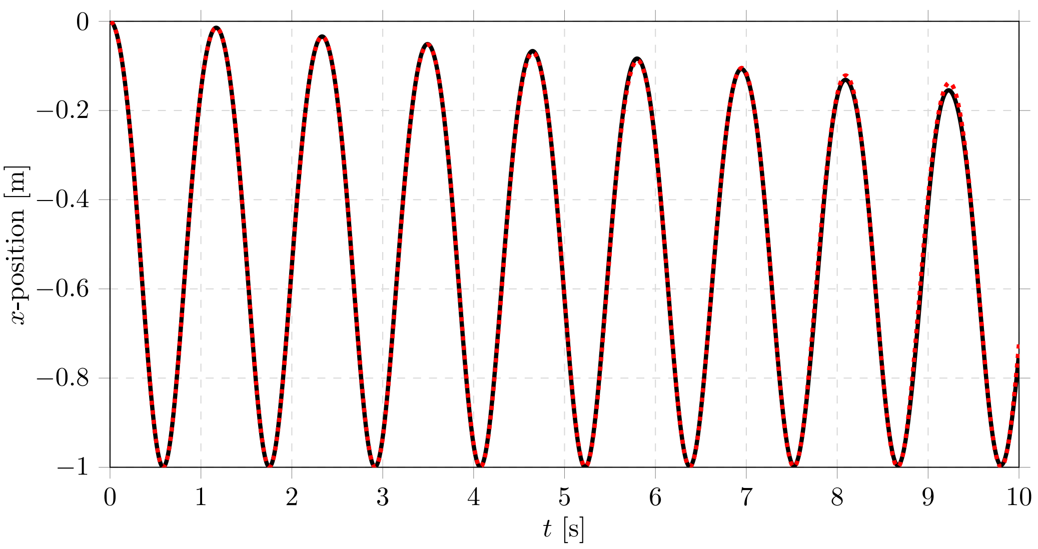


Figure 7: Simulation with complete model in BEAST (black curve) compared with co-simulation using BEAST for each part (red curve) and $h_{\mathrm{TLM}} = 1 \cdot 10^{-5}$s. The curves shows the x-position of the lower arm bearing connection point.

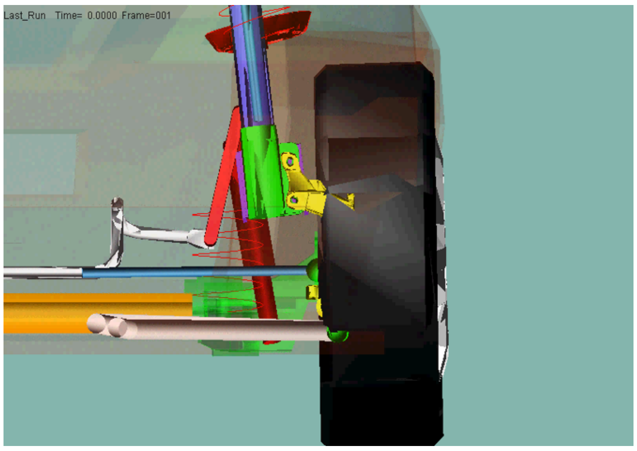


Figure 8: The car modelled in MSC.Adams. Details of the front left wheel suspension.

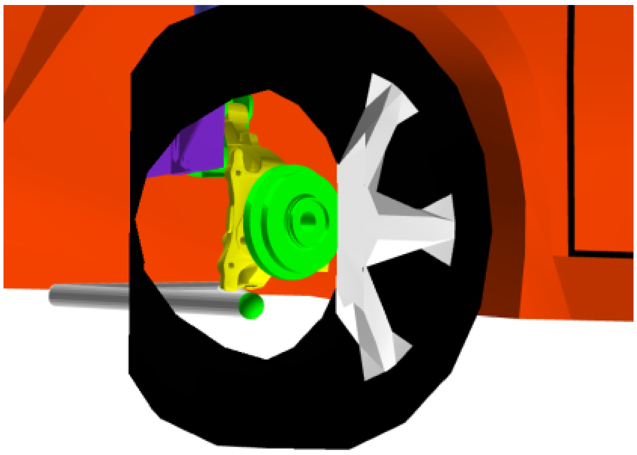


Figure 9: The SKF wheel end bearing unit placed and connected to the wheel and the front wheel suspension.

6.3 Magnetic spindles

Magnetic spindles are typically rotating shaft systems where the shaft is supported on magnetic bearings. Under normal operation the spindles are levitated by the magnetic force, and have no physical contact with the supporting structure. See Figure 10.

There are sensors that measure the motion of the spindle which is input to a control system which then controls the magnetic force.

The sensitive magnetic parts are protected by touchdown bearings, typically a set of angular contact ball bearings, or a deep grove ball bearing. These bearings normally have a clearance between the inner rings and the shaft that is smaller than the clearance between the magnetic parts, see Figure 11.

When the spindle drops on to the touchdown bearings under operation, they experience high impact loads and fast accelerations.

Touchdown bearings only operate for very limited periods of time, which means that traditional rolling contact fatigue failures does not occur. Instead the

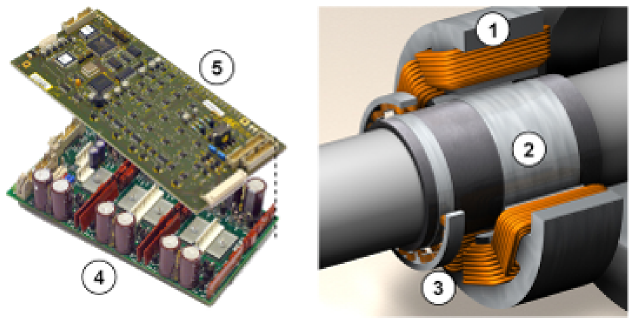


Figure 10: A magnetic bearing. 1 - electro-magnets, 2 - rotor, 3 -position sensors, 4 - power amplifiers, 5 - control.

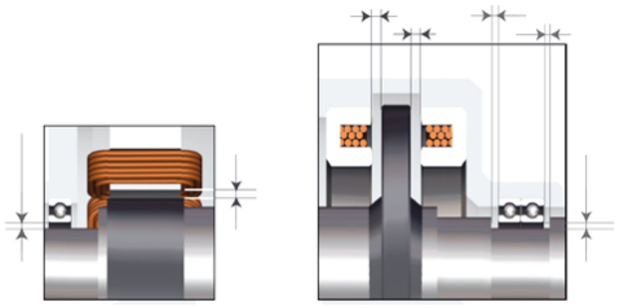


Figure 11: Touchdown bearings which protect the magnetic bearings. Clearances indicated.

failure modes are more related to high power dissipation in the contacts, due to high slip speeds, high loads, combined with poor lubrication conditions.

The main failure modes are therefore:

- Load at impact may lead to indentation marks in the raceways.
- Shaft whirl may give very high bearing loads, especially in the case of backward whirl.
- High acceleration may lead to high sliding power, leading to smearing damage.

A number of magnetic spindle configurations have been modelled and simulated as well as experimentally investigated (Anders et al., 2013).

One of them, a turbomolecular pump, is a small system. The length of the shaft is 192 mm and the complete rotor weighs 4.4 kg. Figure 12 shows a drawing of the rotor. The model system contains two sets of radial magnetic and landing bearings. One of the landing bearings is modelled in BEAST. The remaing system including the rotor, magnetic actuators, position sensors and controller are modelled in Matlab/Simulink. For simplicity, the axial motion and therefore all axial bearings, are excluded. By means of co-simulation, the complete system is integrated into one simulation model for magnetic spindles. This allows for a wide range of studies of the function of the complete magnetic spindle system, and for further optimization.

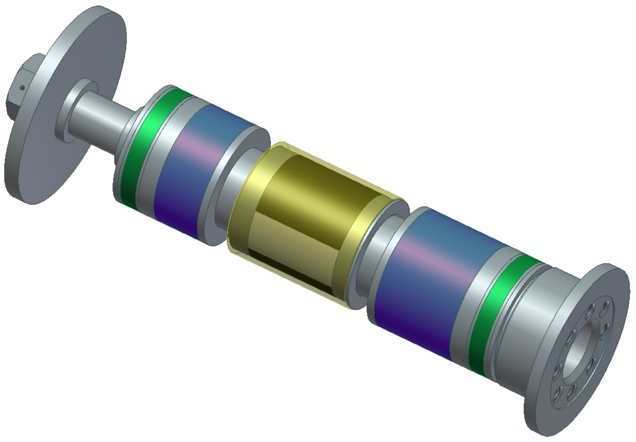


Figure 12: The modelled turbomolecular pump spindle with a simplified fan wheel to the left. The yellow part in the centre is the electric rotor. The magnetic bearings are indicated as blue segments and their corresponding position sensors in green. The landing bearing modelled in BEAST (not shown in the drawing) is located close to the left position sensor.

In particular, landing bearings can become active for large variation in load and/or external vibrations. Such situations require detailed description of controller, actuators and internal mechanics to determine the system response. As an example we present results for translational disturbance of the pump causing displacement of the rotor relative the stator (housing). Figure 13 shows the displacement in x and y of the shaft at the location of the bearing relative to the stator. The controller cannot fully compensate the disturbance and the shaft moves in x until it hits the landing bearing. The contact sets the bearing in motion. The resulting slip power densities in the contacts between balls and raceways are shown in Figure 14. After some time the control system is able to restore the position of the shaft. The bearing continues to roll after the contact, but without slip.

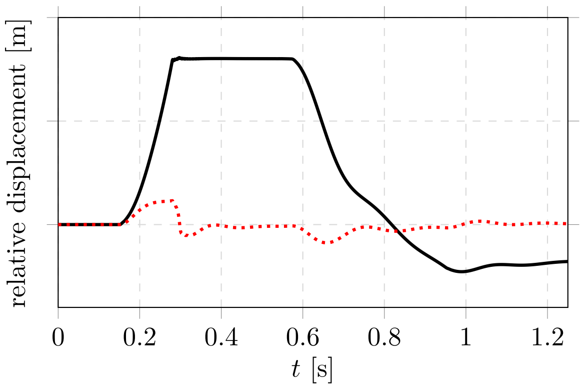


Figure 13: Displacement (x black curve, y red curve) of rotor shaft relative to the stator at the location of the landing bearing modelled in BEAST.

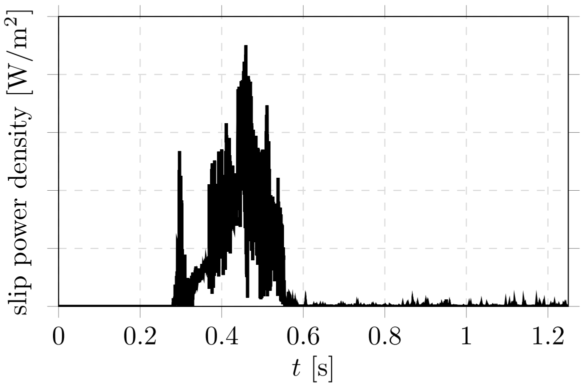


Figure 14: Maximum slip power loss density for all ball/raceway contacts in the landing bearing.

7 Discussion

A TLM connection is a physical model. It can be created for all kinds of physics that include wave propagation.

If it is practical to use TLM in a MST depends on the typical TLM delay time compare with typical simulation time step lengths of the connected external models, e.g., TLM for electricity in a short conductor would yield a very short delay time that might dominate the simulation behaviour. However, in many situations it is possible to modify the TLM input, e.g., assume a longer electrical conductor, to give longer delay times while still obtaining useful results of the system simulation.

The MST framework currently supports bi-directional TLM connectors for 3-D mechanics, 1-D mechanics (rotational and translatory), and hydraulics. It has also one-directional delayed signals of any type. More connector types will be added as need arise.

Refering to the five main challenges listed in the Introduction, the MST:

- has robust and numerical stable simulation due to the TLM technique,
- has inherent parallel execution of the external models,
- does support FMI 2.0,
- has high level connector types that hides low level details,
- can be integrated with different model editors, model languages, and postprocessing tools.

  Which editor to use depends on the main modelling domain, e.g., for 3-D mechanics BEAST model editor discussed here is suitable, while for other domains a 2-D editor can work fine.

The MST framework has been donated to OSMC (Open Source Modelica Consortium (OSMC)) by SKF, and since then further developed in cooperation. It is available from OSMC as the OMSimulator 1.0 with OMEdit as the model editor.

8 Conclusion

BEAST is a dynamic simulation software specialized in the analysis of mechanical systems with contacts.

It has also a framework for composite modelling and co-simulation, i.e., it is a Master Simulation Tool (MST) including composite model editor. It uses asynchronous TLM-based co-simulation to decouple the external models/tools and ensure robust numerical behaviour of the complete composite model. The excecution is parallel, i.e., each external model/tool runs in parallel, preferably on its own hardware.

Attaching new tools to the framework is facilitated by a minimalistic coupling interface. The framework has direct or native support for several simulation tools since long. New extensions are the capability for mixed modelling, the support for FMI, and improved direct couplings to Modelica tools including a high precision one-step simulation mode for OpenModelica.

The support for FMI adds compatibility with a large number of simulation tools, currently about 100 tools with partial or full conformance (Functional Mock-up Interface (FMI)).

Composite modelling and simulation can be a solution to utilize investments in models and tools, use the right tool for the right task, increase the accuracy by means of better boundary conditions, and facilitate cooperation in organizations. With the new FMI standardization, efforts are increasing to make this happen.

References

Anders, J., Stacke, L.-E., and Leslie, P. Rotor drop simulations and validation with focus on internal contact mechanisms of hybrid ball bearings. In *Proc. of ASME Turbo Expo*. San Antonio, Texas, USA, 2013. doi:10.1115/GT2013-95816.

Auslander, D. Distributed system simulation with bilateral delay-line models. *Journal of Basic Engineering*, 1968. pages 195–200. doi:10.1115/1.3605079.

Awais, M. U., Palensky, P., Mueller, W., Widl, E., and Elsheikh, A. Distributed hybrid simulation using the hla and the functional mock-up interface. *Industrial Electronics Society, IECON*, 2013. pages 7564–7569. doi:10.1109/IECON.2013.6700393.

Bastian, J., Clauß, C., Wolf, S., and Schneider, P. Master for co-simulation using FMi. In *8th International Modelica Conference, Dresden*. Citeseer, 2011. URL `Permalink:http://publica.fraunhofer.de/documents/N-162331.html`.

Blochwitz, T., Otter, M., Arnold, M., Bausch, C., Clauß, C., Elmqvist, H., Junghanns, A., Mauss, J., Monteiro, M., Neidhold, T., Neumerkel, D., Olsson, H., Peetz, J.-V., and Wolf, S. The Functional Mockup Interface for tool independent exchange of simulation models. In *8th International Modelica Conference*. Como, Italy, 2011. doi:10.3384/ecp12076173.

Braun, R., Ericsson, L., and Krus, P. Full vehicle simulation of forwarder with semi active suspension using co-simulation. In *ASME/BATH 2015 Symposium on Fluid Power and Motion Control*. 2015. doi:10.1115/FPMC2015-9588.

Braun, R., Hällquist, R., and Fritzson, D. TLM-based Asynchronous Co-simulation with the Functional Mockup Interface. In *Proceedings of IUTAM Symposium on Co-simulation and Solver coupling*. Darmstadt, Germany, 2017.

Braun, R. and Krus, P. Tool-independent distributed simulations using transmission line elements and the functional mock-up interface. In *SIMS 54th Conference*. 2013. doi:10.1145/2666202.2666212.

Dassault Systémes Dymola. 2018.
On-line: http://www.3ds.com.

Elsheikh, A., Awais, M. U., Widl, E., and Palensky, P. 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*. IEEE, pages 1–6, 2013. doi:10.1109/MSCPES.2013.6623315.

Fritzson, D., Stacke, L.-E., and Anders, J. Dynamic simulation — Building knowledge in product development. *SKF Evolution*, 2014. 1(1):21–26.

Functional Mock-up Interface (FMI). 2018.
On-line: http://www.fmi-standard.org.

Hindmarsh, A. C., Brown, P. N., Grant, K. E., Lee, S. L., Serban, R., Shumaker, D. E., and Woodward, C. S. SUNDIALS: Suite of nonlinear and differential/algebraic equation solvers. *ACM Transactions on Mathematical Software (TOMS)*, 2005. 31(3):363–396. doi:10.1145/1089014.1089020.

Hopsan project. 2018. On-line:
http://www.iei.liu.se/flumes/system-simulation/hopsan.

Hui, S. and Christopoulos, C. Numerical simulation of power circuits using transmission-line modelling. *IEE Proceedings A (Physical Science, Measurement and Instrumentation, Management and Education)*, 1990. 137:379–384(5). doi:10.1049/ip-a-2.1990.0060.

Johns, P. B. and O'Brien, M. A. Use of the transmission-line modelling (t.l.m.) method to solve non-linear lumped networks. *The Radio and Electronic Engineer*, 1980. 50(1/2):59–70.

Krus, P. Distributed modelling techniques for system simulation. Technical report, Linköping University, The Institute of Technology, 2007.

MathWorks Simulink. 2018.
On-line: http://www.mathworks.com.

Modelica and the Modelica Association. 2018.
On-line: http://www.modelica.org.

MSC Adams. 2018.
On-line: http://www.mscsoftware.com.

Nakhimovski, I. *Contributions to the Modeling and Simulation of Mechanical Systems with Detailed Contact Analysis*. Ph.D. thesis, Linköpings universitet, Sweden, 2006. URL `oai:DiVA.org:liu-6342`. Linköping Studies in Science and Technology, Dissertation No. 1009, ISBN: 91-85497-43-X, ISSN: 0345-7524.

Neema, H., Gohl, J., Lattmann, Z., Sztipanovits, J., Karsai, G., Neema, S., Bapty, T., Batteh, J., Tummescheit, H., and Sureshkumar, C. 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*, 096. Linköping University Electronic Press, pages 235–245, 2014. doi:10.3384/ECP14096235.

Open Source Modelica Consortium (OSMC). 2018. On-line: http://www.openmodelica.org.

Schierz, T., Arnold, M., and Clauß, C. Co-simulation with communication step size control in an FMi compatible master algorithm. In *9th Int. Modelica Conference, Munich, Germany*. pages 205–214, 2012. doi:10.3384/ecp12076205.

Schweizer, B., Lu, D., and Li, P. Co-simulation method for solver coupling with algebraic constraints incorporating relaxation techniques. *Multibody System Dynamics*, 2016. 36(1):1–36. doi:10.1007/s11044-015-9464-9.

Siemers, A. *Contributions to the Modeling and Visualisation of Multibody Systems Simulations with Detailed Contact Analysis*. Ph.D. thesis, Linköpings universitet, Sweden, 2010. URL oai:DiVA.org:liu-60303. Linköping Studies in Science and Technology, Dissertation No. 1337, ISBN: 978-91-7393-317-9, ISSN: 0345-7524.

Siemers, A., Fritzson, D., and Nakhimovski, I. General meta-model based co-simulations applied to mechanical systems. *Simulation Modelling Practice and Theory*, 2009. 17(4):612–624. doi:10.1016/j.simpat.2008.10.006.

Viersma, T. J. *Analysis, synthesis, and design of hydraulic servosystems and pipelines*. Elsevier Scientific Pub. Co., 1980.

Wolfram SystemModeler. 2018. On-line: http://www.wolfram.com.