FMI Change Proposal FCP-XXXX:
Callback functions for numerical stability

Abstract

Numerical stability is a key aspect in co-simulation of physical systems. Decoupling a system into independent sub-models will introduce time delays on interface variables. Utilizing physical time delays for decoupling reduces the effect on numerical stability and helps to conserve signal energy content. This requires interpolation, to allow solvers to request input variables for the time slot where they are needed. The FMI for co-simulation standard does not support callbacks between FMU and Master which would enable each FMU to access high resolution data interpolated from interpolation tables located in the orchestrating master steering the simulation. Mechanical and thermodynamic models of industrial relevance have been used to demonstrate the need for interpolation and it is shown that callback functions are the most suitable enabler. This FMI change proposal provides a reference implementation of an extension to the callback functions struct, a new API function, and an added attribute to the FMI specification xml schema.

Revision

March 12, 2018	Initial draft by Robert Hällqvist and Robert Braun
April 20, 2018	Revises draft by Robert Hällqvist Robert Braun, and Magnus Eek

Table of Contents

1. Rational

Sketch of the proposal (especially with examples) and an explanation of the business case: Why should this feature be included? What problems can be solved (better) that cannot be solved (as easily) now? Sketch industrial (real world) use cases where the proposal is essential, and if possible the companies/organizations that require it.

Numerical robustness is a key factor in simulation solver coupling. Using different solvers for different parts of a simulation model can be of great benefit in terms of increasing simulation speed and robustness. However, all variables shared by more than one solver will need to be delayed in time. This may result in numerical errors and instability. One solution is to decouple the model where significant physically motivated time delays are present. In this way, all time delays are a natural part of the model and no non-physical time delays will thus need to be inserted.

Kommenterad [EM1]: Vet inte hur det brukar vara i FCP men detta känns långt och mer som ett paper. Tveksamt om det är bra att namnge olika verktyg i ett CFP, det ska nog vara så neutralt som möjligt (men refa är så klart bra).

Kommenterad [EM2]: Det är väl så att time delay plockas bort från modell och läggs i master?

Transmission Line Modelling (TLM) is a well-known technique for decoupling of simulation models and successfully applied in several M&S tools, see e.g. Ref. [1], [2], [3]. In addition, the TLM technique is implemented in several Modelica libraries. The basic idea of TLM is to utilize that the information propagation speed always is limited by the physical properties of the propagation medium. This includes, for example, stress waves in materials or pressure waves in fluids. By including physical time delays in model variables, equations can be separated without affecting numerical stability. A 1D mechanical TLM element and its equations are shown in Figure 1. F is the force, v velocity, Δt the time delay and Zc characteristic impedance. The delayed information traversing the element is denoted the wave variable. Force and velocity can be replaced by effort and flow variables in other physical domains.

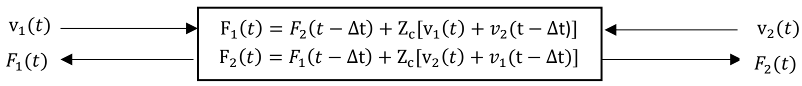


Figure 1. A TLM element including its equations.

The TLM technique is fully applicable to FMI for model exchange. However, FMI for co-simulation induces several challenges concerning numerical stability. Since input variables can only be updated at the beginning of each communication step, the stability benefits of TLM are lost. The TLM technique requires each individual solver to have access to interpolated input variables in between the global communication points.

Several solutions enabling use of fine-grained interpolation of FMI inputs, both with and without modifications to the FMI standard, have been implemented and analyzed [4]. An Open-source master simulation tool under development in the ITEA3 research project OpenCPS [5] has been used as reference implementation platform. Industrially relevant use cases provided by the OpenCPS partners have been used to evaluate the solutions ensuring relevance to the industry and academia. The stability benefits of TLM enabled by fine-grained interpolation are quantified by the use-cases presented in Section 3.2. Fine-grained interpolation is best realized via the two suggested callback functions *getComputedEffort* and *setFlow,* see Section 2. In the general case, the function *getComputedEffort* enable FMUs to request more accurate input data at specific times instances different from the pre-defined communication times. The FMU providing the output can then communicate updated output information for whatever internal time point where it is available via the function *setFlow.*

The callback functions *getComputedEffort* and *setFlow* are essential to the OpenCPS partners: Ericsson, The Royal Institute of Technology [6],Linköping University [7], SKF [8], EQUA Simulation Technology

Group [9], RISE SICS East AB [10], Saab Aeronautics [11], SIEMENS Turbine [12], VTT Technical Research Centre of Finland [13], ELTE-Soft [14], IncQueryLabs [15], CEA [16], EDF [17], ESI [18], RTE [19], Inria [20], SHERPA Engineering [21], Sirehna Naval Group [22].

2. Proposed Changes in the FMI Specification

This part must include the proposed text to be included (1:1) in to the FMI specification, as well as changes of the current specification. In the specification document that will be attached to the FCP, one can remove unrelated sections as long as section headings are kept so that section numbers are not disturbed.

The proposed changes in the FMI specification [23] are described in detail in this section. The specification changes are provided in, Listings 1, Listings 2, and Listings 3. The modifications are marked as red. Listings 1 quantifies the proposed additions to the callback struct presented in section 2.1.5 of the FMI specification. Two new callback functions are added to the struct:

- *getComputedEffort*

  The function obtains an array containing *effort* variable(s) from an interpolation table located inside the master. This array is valid for the specific FMU local *time* of which it is needed, provided the current value(s) on *flow* values.

- *setFlow*

  The function enables population of interpolation tables located inside the Master. *Flow* variables(s) at time *time* are passed to the interpolation table.

The proposed extension to the FMI API is provided in Listings 2 and the proposed addition of XML schema attribute is provided in Listings 3.

```
typedef struct {
    void (*logger)(fmi2ComponentEnvironment componentEnvironment,
                   fmi2String instanceName,
                   fmi2Status status,
                   fmi2String category,
                   fmi2String message, ...);
    void* (*allocateMemory)(size_t nobj, size_t size);
    void (*freeMemory) (void* obj);
    void (*stepFinished) (fmi2ComponentEnvironment componentEnvironment,
                          fmi2Status status);
    fmi2ComponentEnvironment componentEnvironment;
    void (*getComputedEffort) (fmi2ComponentEnvironment componentEnvironment,
                               fmi2ValueReference vr,
                               size_t dimensions,       //1, 2 or 3
                               const fmi2Real time,
                               const fmi2Real flow [],  //arrays of size 1, 3 or 6
                               fmi2Real effort []);     //arrays of size 1, 3 or 6
    void (*setFlow) (fmi2ComponentEnvironment componentEnvironment,
                     fmi2ValueReference vr,
                     size_t dimensions,                 //1 ,2 or 3
                     const fmi2Real time,
                     const fmi2Real flow []);           //arrays of size 1, 3 or 6
} fmi2CallbackFunctions;
```

Listings 1. Proposed extension to callback functions struct. The extension is marked as red in the listing.

```
fmi2Status fmi2EnableFlowEffortCallbacks ();
```

Listings 2. Proposed extension to FMI API. The extension is marked as red in the listing.

attributes

modelIdentifier	
type	xs:normal§izedString

needsExecutionTool	
type	xs:boolean
default	false

canHandleVariableCommunicationStepSize	
type	xs:boolean
default	false

canInterpolateInputs	
type	xs:boolean
default	false

maxOutputDerivativeOrder	
type	xs:unsignedInt
default	0

canRunAsynchronously	
type	xs:boolean
default	false

canBeInstantiatedOnlyOncePerProcess	
type	xs:boolean
default	false

canGetAndSetFMUstate	
type	xs:boolean
default	false

canSerializeFMUstate	
type	xs:boolean
default	false

providesDerictionalDerivative	
type	xs:boolean
default	false

canUseFlowEffortCallbacks	
type	xs:boolean
default	false

Listings 3.Proposed addition of XML schema attribute. The extension is denoted as red in the listing

3. Implementation

3.1 Backwards compatibility

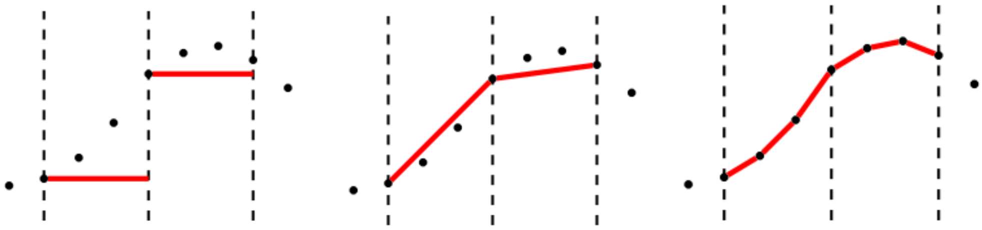
It has to be analyzed whether the proposal is backwards compatible. If any possible, this should be the case. Even if it is backwards compatible, issues should be listed and analyzed that may cause problems.

Backwards compatibility is ensured by the *canUseFlowEffortCallbacks* flag and the *enableFlowEffortCallbacks()* function. In this way, the master will never provide a struct with wrong size to the FMU, and the FMU will never attempt to call a non-existing callback function.

3.2 Experience with prototype

The prototype implementation has to be sketched and the experience gained with the prototype, especially the implementation effort.

Several different prototype implementations were investigated by Braun et. al. in [ref IUTAM]. Use-cases from both industry and academia were used to converge on the most suitable prototype enabling numerically robust co-simulation of FMUs. Four different methods for communication between de-coupled models are investigated: constant input extrapolation, coarse grained input interpolation, fine grained input interpolation inside each FMUs, and fine grained input interpolation inside the master, see Figure 2.



(a) Constant extrapolation (b) Coarse-grained interpolation (c) Fine-grained interpolation

Figure 2. Investigated methods.

- *Constant extrapolation*. Input variables are updated at the beginning of each step and remain constant during the step, according to the zero-order hold model.
- *Coarse-grained Interpolation*. It is possible to provide an FMU with the approximated values of input variable time derivatives. These can be used by the FMU for interpolation or extrapolation
- *Fine-grained Interpolation Inside the FMU.* An interpolation table is sent to each FMU, the FMU can then interpolate information on effort variables using the TLM equations, see Figure 1, for whatever local time that it is necessary.
- *Fine-grained Interpolation Inside the Master.* An interpolation table located in the master is accessed by interfacing FMUs via the callback functions presented in Listings 1 and Listings 2.

All four different methods are investigated in the first two use-cases presented below. Fine-grained interpolation inside the Master is enabled as these use cases are developed using a hand coded prototype of the presented callback functions. The first three methods are investigated in the third use-case. The third use-case is developed in a commercially available Modelica tool supporting the current version of the FMI standard. Fine grained interpolation inside the master is therefore not investigated; however, the stability benefits of fine grained interpolation are demonstrated by means of modifying the models such that they can handle internal interpolation.

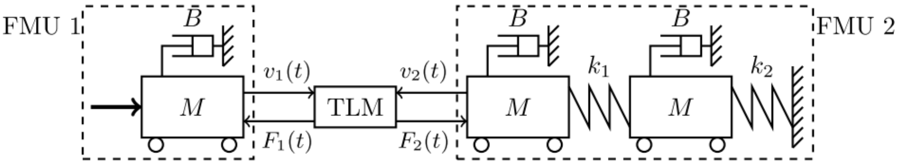
The use-cases demonstrate that fine-grained interpolation is required to achieve stable connections for all the presented composite models. Variables can either be interpolated locally inside the FMU, or by the master simulation tool. Fine-grained Interpolation Inside the FMU requires the FMU to contain the TLM boundary equations. Furthermore, each FMU must be provided with its complete interpolation table. This leads to a large amount of data exchange, which may reduce simulation performance. The resolution of the data that is passed to each FMU needs to be with fixed resolution; this is introduces a severe limitation in terms of modelling flexibility, simulation robustness and performance.

The investigation shows that interpolation inside the master is the best solution in terms of complexity, modeling flexibility, and simulation robustness. However, the callback functions presented in Listings 1and Listings 2 are required for this solution. Such callback functions would not only improve on master algorithms implementing TLM. Numerical stability and simulation robustness can be greatly improved in non-inherently parallel algorithms as well. The callback functions could enable FMUs to pass output information to the master whenever it is available internally, and FMUs could ask the master for updated input information when locally needed. However, this option has not been further investigated here.

Three-mass System

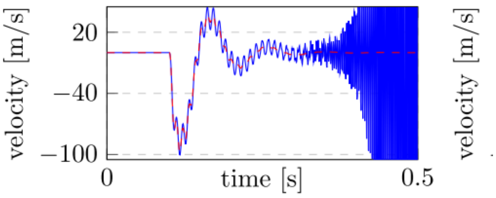
An example composite model consisting of a three-mass system separated into two FMUs is presented in Figure 3. Simulation results verify the proposed method with second order dynamics in one dimension. One of the FMUs has internal dynamics with two different resonance frequencies. The composite model is simulated with a communication step size (i.e. TLM time delay) of 0.4 ms. After 0.1 s, a step force of 100 N is applied on the first mass. Parameters are intentionally chosen to make the simulation unstable when using constant extrapolation. Simulation results using constant extrapolation, coarse-grained interpolation, interpolation and callback functions are shown in
Figure 4
~~Figure 4~~. Constant extrapolation and coarse-grained interpolation methods are unstable. Fine-grained interpolation is able to stabilize the coupling both when computed in MST and locally inside the FMU.

Formaterat: Normal, Vänster

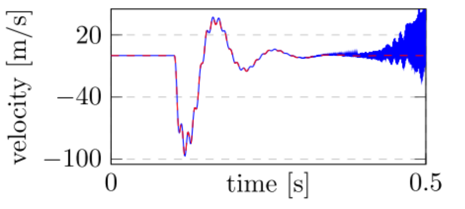


$M = 0.005$ kg $k_1 = 90$ N/m
$B = 0.2$ Ns/m $k_2 = 140$ N/m

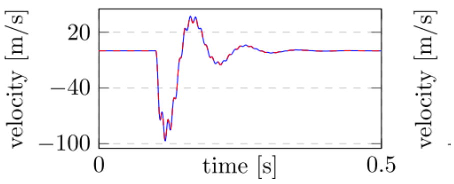
Figure 3. A test model consisting of a three-mass system is divided into two FMUs



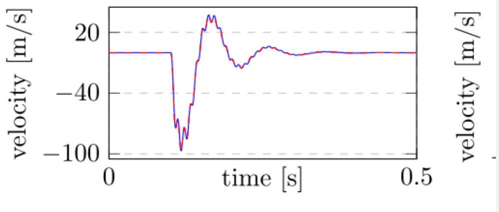
(a) Constant extrapolation



(b) Coarse-grained interpolation



(c) Fine-grained interpolation inside FMU



(d) Fine-grained interpolation in MST

Figure 4. Velocity against time at the left side of the TLM connection. Red dashed line is the exact reference solution

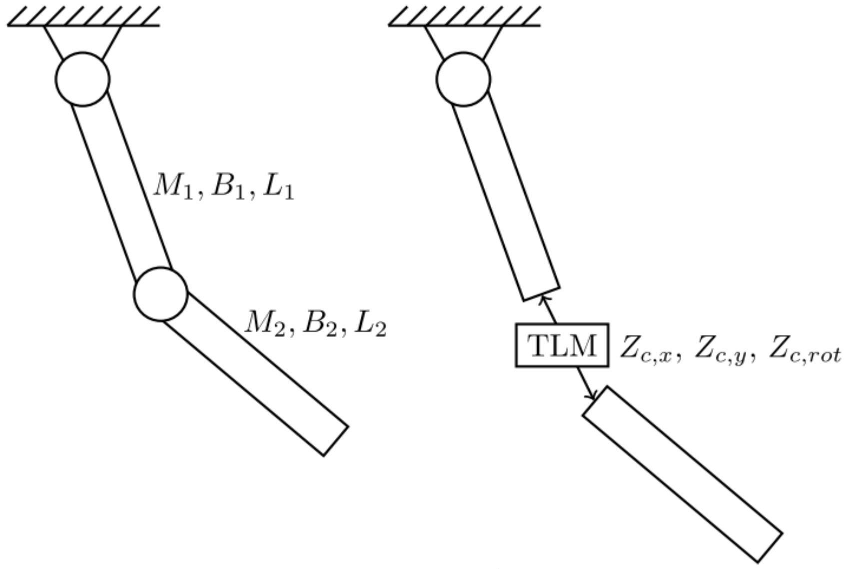
Two-dimensional double pendulum

The second example composite model consists of a two-dimensional double pendulum, see Figure 5. Double pendulums exhibit chaotic motion and are sensitive to initial conditions. This makes it an interesting example for verifying simulation techniques. The two arms are connected through a TLM element with a time delay of Δt = 1e−3 s and a characteristic impedance of Zc = 1e5 for X and Y directions. This corresponds to a spring of stiffness Ks = 1e8 N/m. Rotational impedance is set to zero. Hence, the TLM element represents a revolute joint with some flexibility. Simulation is initiated with both arms pointing horizontally. Results from simulations implementing the four different methods are shown in

Figure 6

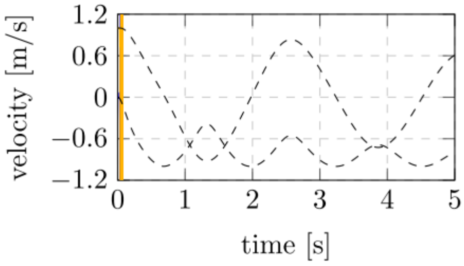
~~Figure 6~~. Blue and orange curves represent vertical and horizontal position, respectively. Black dashed curves are the exact reference solutions. Constant extrapolation and coarse-grained interpolation are both unstable. Fine-grained interpolation results in stable simulation.

Formaterat: Normal

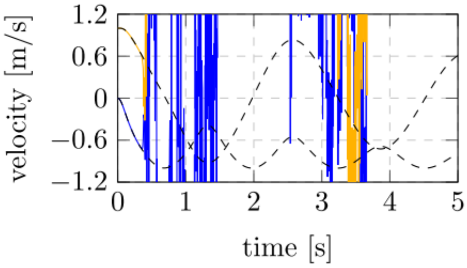


$$M_1 = 100 \text{ kg} \quad B_1 = 20 \text{ Ns/m} \quad L_1 = 0.5 \text{ m}$$
$$M_2 = 100 \text{ kg} \quad B_2 = 10 \text{ Ns/m} \quad L_2 = 0.5 \text{ m}$$
$$Z_{c,x} = 1e5 \text{ Ns/m} \; Z_{c,y} = 1e5 \text{ Ns/m} \; Z_{c,rot} = 0 \text{ Nms}$$

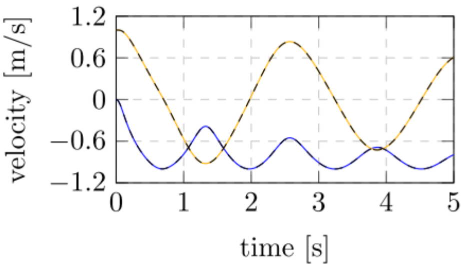
Figure 5. A double pendulum is modelled as two arms connected by a TLM element. Impedance in the rotational dimension is set zero to allow free rotation.



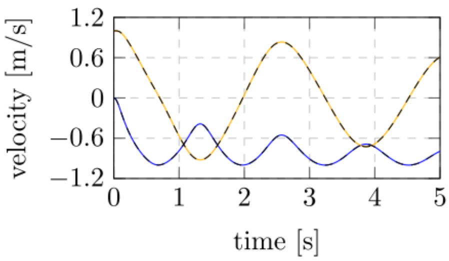
(a) Constant extrapolation



(b) Coarse-grained interpolation



(c) Fine-grained interpolation inside FMU



(d) Fine-grained interpolation in MST

Figure 6. Vertical and horizontal positions of the lower end of the double pendulum model. Dashed lines are the exact reference solutions.

Thermodynamic connection

The presented industrial application consists of two connected FMUs for cosimulation generated from one Modelica model. The two FMUs are connected via a TLM element with a characteristic impedance of Zc = 700000 sPa/m3 and Δt = 0.24 ms. These transmission line settings stem from a pipe with an approximate length of 0.1 m in which an ideal and incompressible gas flows. The physical quantities not accounted for in the TLM connection, in this case specific enthalpy and two phase water content, are passed in both directions directly between the two FMUs as delayed signal connections.

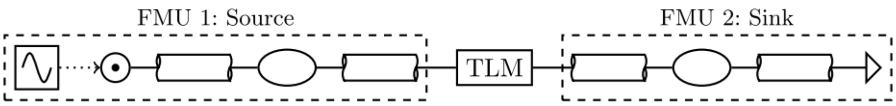
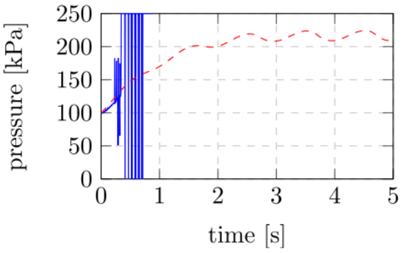
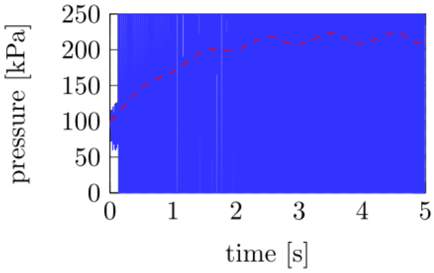


Figure 7. Schematic description of industrial use-case. Two FMUs are connected via a TLM element. The FMUs originate from one Modelica model that is parametrized such that it can represent a source and a sink.

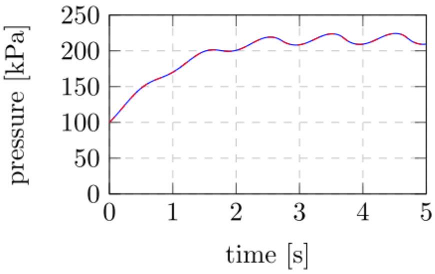
In Figure 7, the left-hand side FMU instantiation acts as a source generating an oscillating mass flow to the second instantiation which acts as a sink. The FMU sink and source specific characteristics are specified via input parameters.



(a) Constant extrapolation



(b) Coarse-grained interpolation



(c) Fine-grained interpolation inside FMU

Figure 8. Pressure in the TLM connection of the thermodynamic model. Dashed line is the Modelica simulation reference solution.

The composite model presented in Figure 7 is simulated using constant extrapolation, coarse grained interpolation, and fine-grained interpolation inside the FMU. The latter two are achieved by means of modifying the Modelica model prior to FMU export. Fine-grained interpolation using callbacks is not possible with FMI 2.0, as a result a fixed pre-defined resolution of inputs variables need to be defined. This is a significant drawback both in terms of simulation performance and robustness; however, fine grained interpolation is possible even though it is cumbersome. An interpolation table in the adaptor is populated at the start of each communication step; the pressure input to the original model is then available at all necessary internal times by means of linear interpolation. The adaptor presented in listing listing 1.4 receives information on wave variables and their corresponding time derivatives from the master at the beginning of each communication step. The composite model input pressure can then be estimated for any necessary internal times by means of the forward Euler method.

Figure 8

~~Figure 8~~ clearly visualizes the advantage of having interpolated input information available as numerical stability is maintained when using fine-grained interpolation. Coarse-grained interpolation and constant extrapolation results in obvious stability issues. The simulation is terminated prematurely as a result of the severe stability issues resulting from constant extrapolation.

Formaterat: Normal, Vänster

3.3 Required Patents

At best of our knowledge, there are no patents required for implementation of this proposal.

4. References

[1] R. Braun and P. Krus, "Tool-Independent Distributed Simulations Using Transmission Line Elements And The Functional Mock-up Interface," in *proceedings of the 53rd SIMS conference on Simulation and Modelling*, Reykjavik, 2013.

[2] M. Sjölund, R. Braun, P. Fritzon and P. Krus, "Towards Efficient Distributed Simulation in Modelica using Transmission Line Modeling," in *proceedings of 3rd International Workshop on Equation-Based Object-Oriented Languages and Tools* , Oslo, Norway, 2013.

[3] A. Siemers, D. Fritzson and I. Nakhimovski, "General meta-model based co-simulations applied to mechanical systems," *Simulation Modelling Practice and Theory,* vol. 17, no. 4, pp. 612-624, 2009.

[4] R. Braun, R. Hällqvist and D. Fritzon, "TLM-Based Asynchronous Co-Simulation with the Functional Mock-up Interface," in *Proceedings of the IUTAM Symposium on Co-simulation and solver coupling*, Darmstadt, Germany, 2018.

[5] OpenCPS, "Open Cyber-Physical System Model-Driven Certified Development," 2018. [Online]. Available: https://www.opencps.eu/. [Accessed 09 02 2018].

[6] The Royal Institute of Technology, "KTH," 2018. [Online]. Available: https://www.kth.se/. [Accessed 09 02 2018].

[7] Linköping University, "Linköping University," 15 02 2018. [Online]. Available: https://liu.se/. [Accessed 15 02 2018].

[8] SKF, "SKF," 15 Februari 2018. [Online]. Available: http://www.skf.com. [Accessed 15 Februari 2018].

[9] EQUA Simulation AB, "EQUA," EQUA Simulation AB, 15 February 2018. [Online]. Available: https://www.equa.se/en/about-us/about-equa. [Accessed 15 February 2018].

[10] RISE SICS East, "RISE," 15 February 2018. [Online]. Available: https://www.sics.se/groups/rise-sics-east. [Accessed 15 February 2018].

[11] Saab AB, "Saab Corporate," Saab AB, 15 February 2018. [Online]. Available: https://saabgroup.com/about-company/. [Accessed 15 February 2018].

[12] SIEMENS, "SIEMENS Ingenuity for life," SIEMENS, 15 February 2018. [Online]. Available: https://www.siemens.com/global/en/home/products/energy/power-generation.html. [Accessed 15 February 2018].

[13] VTT Technical Research Centre of Finland Ltd, "VTT," VTT Technical Research Centre of Finland Ltd, 15 February 2018. [Online]. Available: http://www.vttresearch.com/. [Accessed 15 February 2018].

[14] ELTE-Soft, "Welcome to ELTE-Soft!," ELTE-Soft, 15 February 2018. [Online]. Available: http://www.eltesoft.hu/en/. [Accessed 15 February 2018].

[15] IncQueryLabs, "IncQueryLabs," IncQueryLabs, 15 February 2018. [Online]. Available: https://incquerylabs.com/en. [Accessed 15 February 2018].

[16] list CEA tech, "list CEA tech Smart digital systems," list CEA tech, 15 February 2018. [Online]. Available: http://www-list.cea.fr/en/. [Accessed 15 February 2018].

[17] EDF, "EDF," EDF, 15 February 2018. [Online]. Available: https://www.edf.fr/en/meta-home. [Accessed 15 February 2018].

[18] ESI Group, "ESI get it right," ESI Group, 15 February 2018. [Online]. Available: https://www.esi-group.com/. [Accessed 15 February 2018].

[19] RTE, "RTE," RTE, 15 February 2018. [Online]. Available: http://www.rte-france.com/en. [Accessed 15 February 2018].

[20] Inria, "Inria inventors for the digital world," Inria, 15 February 2018. [Online]. Available: https://www.inria.fr/en/. [Accessed 15 February 2018].

[21] SHERPA Engineering, "SHERPA Engineering Expertise in modeling and control command," SHERPA Engineering, 15 February 2018. [Online]. Available: http://www.sherpa-eng.com/en/?noredirect=en_US. [Accessed 15 February 2018].

[22] Sirehna Naval Group, "Sirehna Naval Group," Sirehna Naval Group, 15 February 2018. [Online]. Available: http://www.sirehna.com/. [Accessed 15 February 2018].

[23] FMI development group, "Functional Mock-up Interface for Model Exchange and Co-Simulation 2.0," FMI development group, 2014.

Examples:

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., Wolf S. (2011): **The Functional Mockup Interface for Tool independent Exchange of Simulation Models**. 8th International Modelica Conference, Dresden 2011. http://www.ep.liu.se/ecp/063/013/ecp11063013.pdf

Blochwitz T., Otter M., Akesson J., Arnold M., Clauß C., Elmqvist H., Friedrich M., Junghanns A., Mauss J,, Neumerkel D., Olsson H., Viel A. (2012): **Functional Mockup Interface 2.0: The Standard for Tool independent Exchange of Simulation Models**. 9th International Modelica Conference, Munich, 2012. http://www.ep.liu.se/ecp/076/017/ecp12076017.pdf

XML: www.w3.org/XML, en.wikipedia.org/wiki/XML

First Appendix

If needed add material to the appendix.

Subchapter of appendix

If needed add material to the appendix.