A Web Architecture for Modeling and Simulation

Hilding Elmqvist Martin Malmheden Johan Andreasson
{Hilding.Elmqvist, Martin.Malmheden, Johan.Andreasson}@Modelon.com
Modelon AB, Sweden, www.Modelon.com

Abstract

A Web Architecture for Modeling and Simulation (WAMS) is presented which enables system modeling in your browser using the Modelica language. Compilation and simulations are done on a server using the Optimica Compiler Toolkit (OCT) from Modelon.

Such an architecture is appropriate for making design space explorations such as sensitivity analysis, DOE, Monte Carlo analysis, optimizations, parameter estimation, etc. efficiently.

Keywords: Web app, Cloud Simulations, Model Based Product Design, Modelica, Modeling, Simulation, Optimization

1 Introduction

Model based product design requires both intuitive and effective user interfaces and large computing power. Fortunately, modern computer systems provide a solution with client software, web apps, running in a web browser and use of cloud computing and cloud storage for simulations and storing models and results. HTML5 and WebGL provides an appropriate basis for web app development. When making design space explorations, multi-core, clusters and cloud computing provide the means for performing multi-simulations such as sensitivity analysis, DOE, Monte Carlo analysis, optimizations, parameter estimation, etc. efficiently.

Such client-server architecture and use of web apps also opens up for performing modeling, simulations and optimizations from a tablet or a smart phone as well as from your computer.

Figure 1. Simulation on the road

A future oriented use case utilizing such a solution is shown in Figure 1 (Berggren, 2017):

- Your self-driving car is taking you home after work.
- You get an idea for how to solve today's frustrating problem.
- You just need to:
  - insert one model component
  - change some parameters
  - simulate
  - look at some plots to verify
- You are eager to test it immediately on your smart phone.

This paper describes an architecture for modeling in a web app using the Modelica language and performing simulation on the cloud.

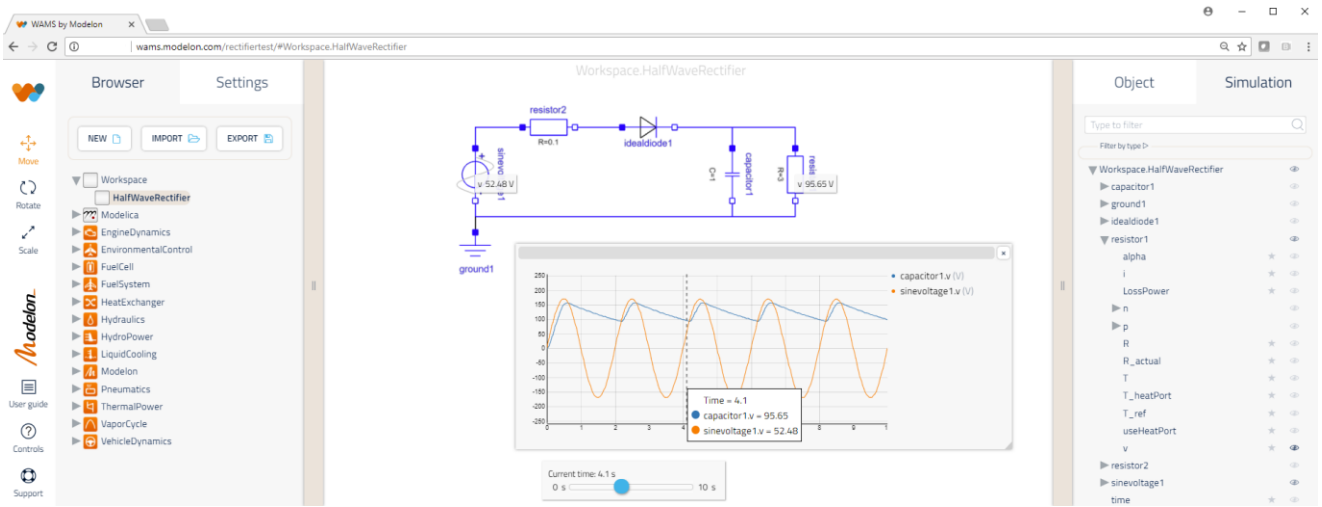

Figure 2. Web app for modeling and simulation

2 Modelica Systems Modeling

The layout of the web app is shown in Figure 2. The left pane contains a package browser for model components described in the Modelica language (Modelica Association, 2014).

The middle part contains the diagram of a system model, i.e. a model consisting of connected model components as well as variable plots.

The right part contains a variable browser for setting parameters, selecting variables to plot, etc. There are elaborate features for searching, for filtering and to declare favorite variables.

The web app has features to create and modify models, such as:

- Drag components from the model browser to the diagram
- Set parameters of a component
- Connect connectors of components
- Multiple select
- Resize, rotate component
- Copy, Paste and Duplicate
- Undo, redo
- Display of web app and model documentation
- Open hierarchical sub-levels

3 Multi-Simulation

A Modelica model is built up on the server while the model is being edited. When a simulation is requested, this Modelica model is compiled using OCT (Modelon, 2018c) into an FMU (Blochwitz, et al., 2012) which is used for the simulation.

The variable browser in the right pane of the web app contains the model hierarchy tree with all variables. A plot of a variable is obtained by dragging a variable into the diagram. If a variable is dragged into an already present plot, the variables are plotted together as shown in Figure 2.

It is also possible to show instantaneous values of variables close to the components using a concept called a *sticky*, see Figure 2. Such stickies are created by selecting a tool close to the variable names in the browser.

Parameter values can be changed in the variable browser and new simulations performed without recompiling the model.

It is possible to specify ranges of values for parameters to perform design space exploration. Multi-simulations are then automatically performed using a specified sampling technique. It is also possible to specify Monte Carlo simulations, optimizations and parameter estimation.

Figure 3 shows how uncertainties in behavior are shown as a "spaghetti" plot when parameters are specified as ranges or with random distributions. In this case, 10 simulations of a robot were made and the tool positions plotted.

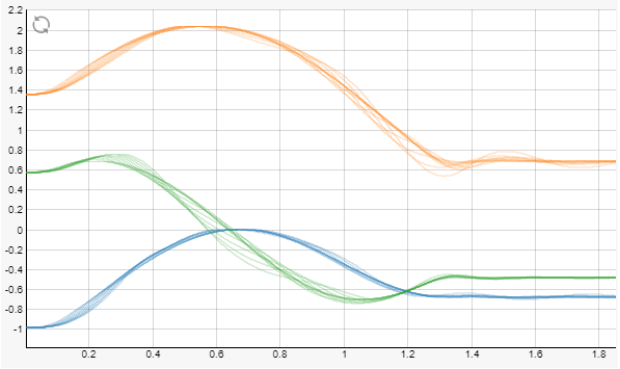

Figure 3. Uncertainties in robot motion for loads 0-300 kg.

3D plots are also available to present variations which depend on two parameter ranges. Figure 4 shows how the camber angle of a wheel suspension depend on wheel travel and steering angle.

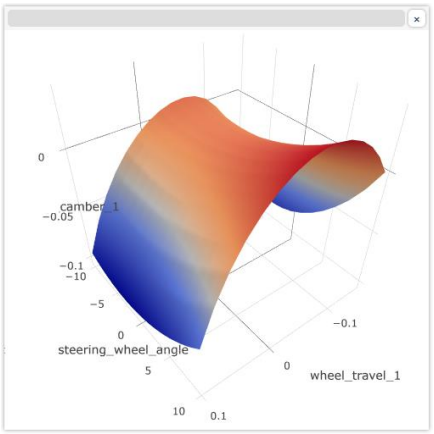

Figure 4. Camber angle versus steering wheel angle and wheel travel

3D animation is provided for models based on the MultiBody library of MSL. Figure 5 shows an animation from the Vehicle Dynamic Library (Modelon, 2018b).

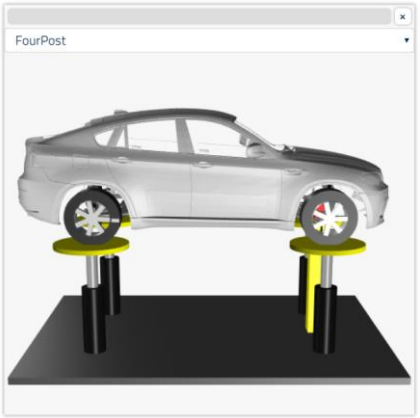

Figure 5. 3D animation of VDL model

DOI 10.3384/ecp18148255

4 Application Example: Electric Powertrain

Figure 6 shows a full vehicle thermal analysis of a battery electric vehicle with a small internal combustion engine (ICE) serving as a range extender. The vehicle has active thermal management of the electric powertrain and cooling of the ICE.

Figure 6 shows the electric powertrain, with battery and two electric machines where one serves as the traction motor and the other as generator connected to the ICE. Battery and machines are all connected to the high voltage bus. Additionally, all components have thermal connectors that allow them to interact with the cooling system. Internally, as seen in Figure 7, the machine has thermal dynamics described individually by motor and power electronics. Correspondingly, the battery thermal dynamics can be resolved at cell level. Further information about the models can be found in Modelon (2018d), Batteh (2018).

Here, the model is configured to study heat propagation as response to a standard drive cycle. The diagram is configured with plots to see the dynamic temperature change over time. Note that as a user, to generate the plot it is enough to know that you are looking for temperatures and then select for what components you want to see the values.

To get an overview of how the heat is propagated through the system, stickies are activated for the thermal components, where the display follows the time slider.

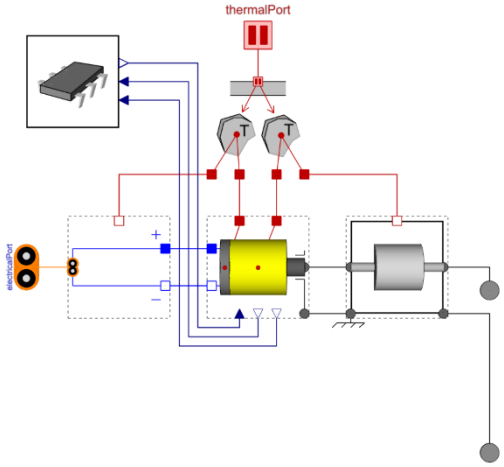

Figure 7. Electric machine.

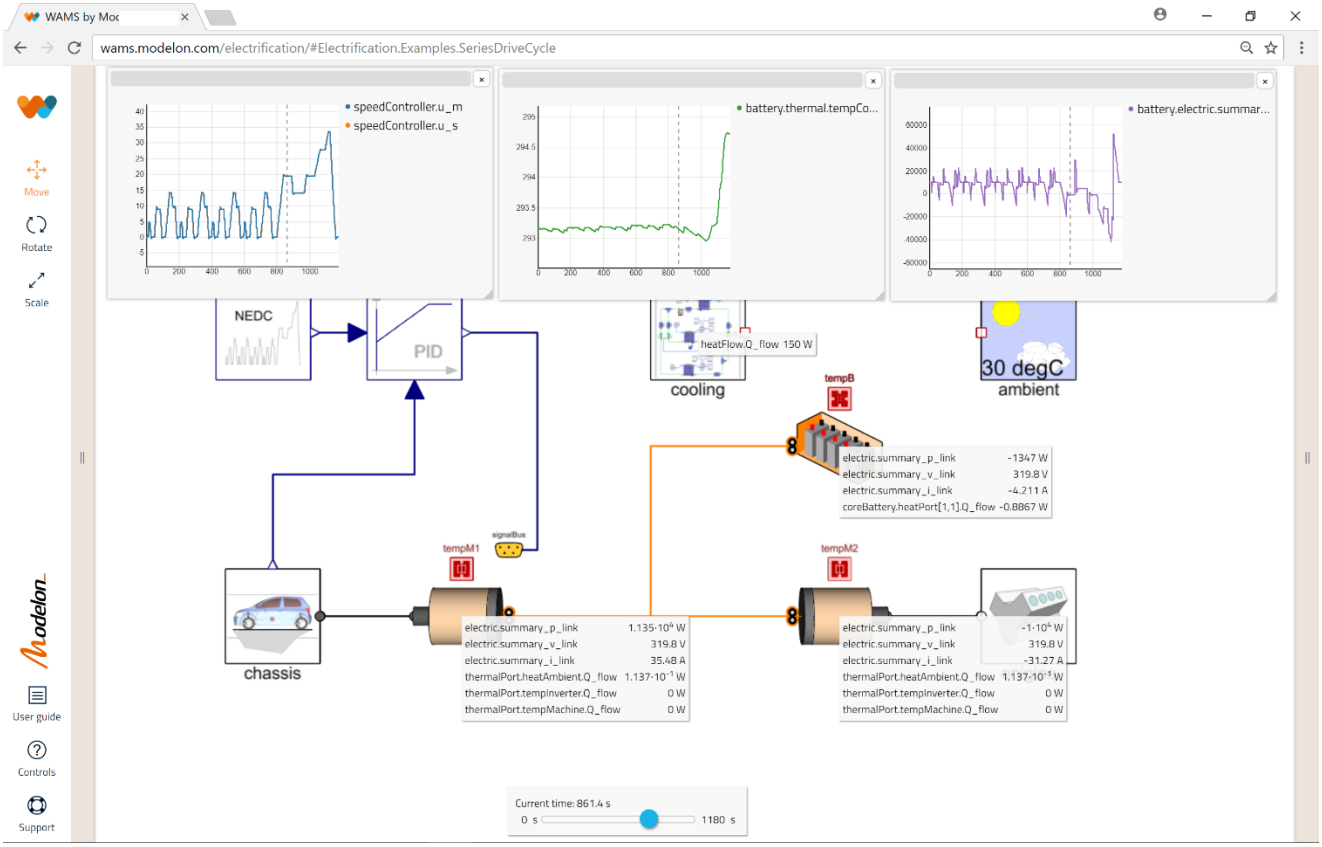

Figure 6. Series hybrid from the Electrification Library.

5 Application Example: Suspension design

Figure 8 illustrates how a model can conveniently be set for multiple executions, in this case for suspension design. Here, a suspension is mounted in a test rig from the Vehicle Dynamics Library (Modelon, 2018a), for evaluation of its kinematic behavior depending on steering and wheel travel.

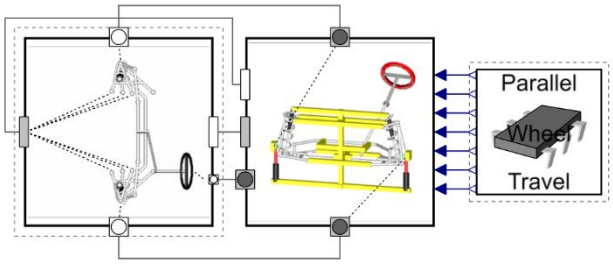

Figure 8. Wheel suspension mounted in test rig.

The analysis is relevant from an engineering perspective to characterize the main behavior of a given suspension design, and is frequently used in suspension design.

The experiment is set-up by defining ranges of the two independent variables and then the resulting executions are automatically triggered and managed. Such an execution can be either dynamic or steady-state.

As can be seen in the plots, Figure 4, the corresponding illustration is in this case a response surface over steering (x) and wheel travel (y). The response variable (z), in this case wheel hub rotations camber, could be either a value at a given time from a dynamic simulation, or the result from a steady-state calculation. In this case, 15 x 15 dynamic simulations were made, and the final steady-state solutions were plotted. The simulations were made on 20 Microsoft Azure cloud nodes.

6 Application Example: Distribution Network

Figure 9 shows a heat distribution network from the Liquid Cooling Library (Modelon, 2018b) that can be used to tune flow resistances to reach a passive distribution of the heat in the network.

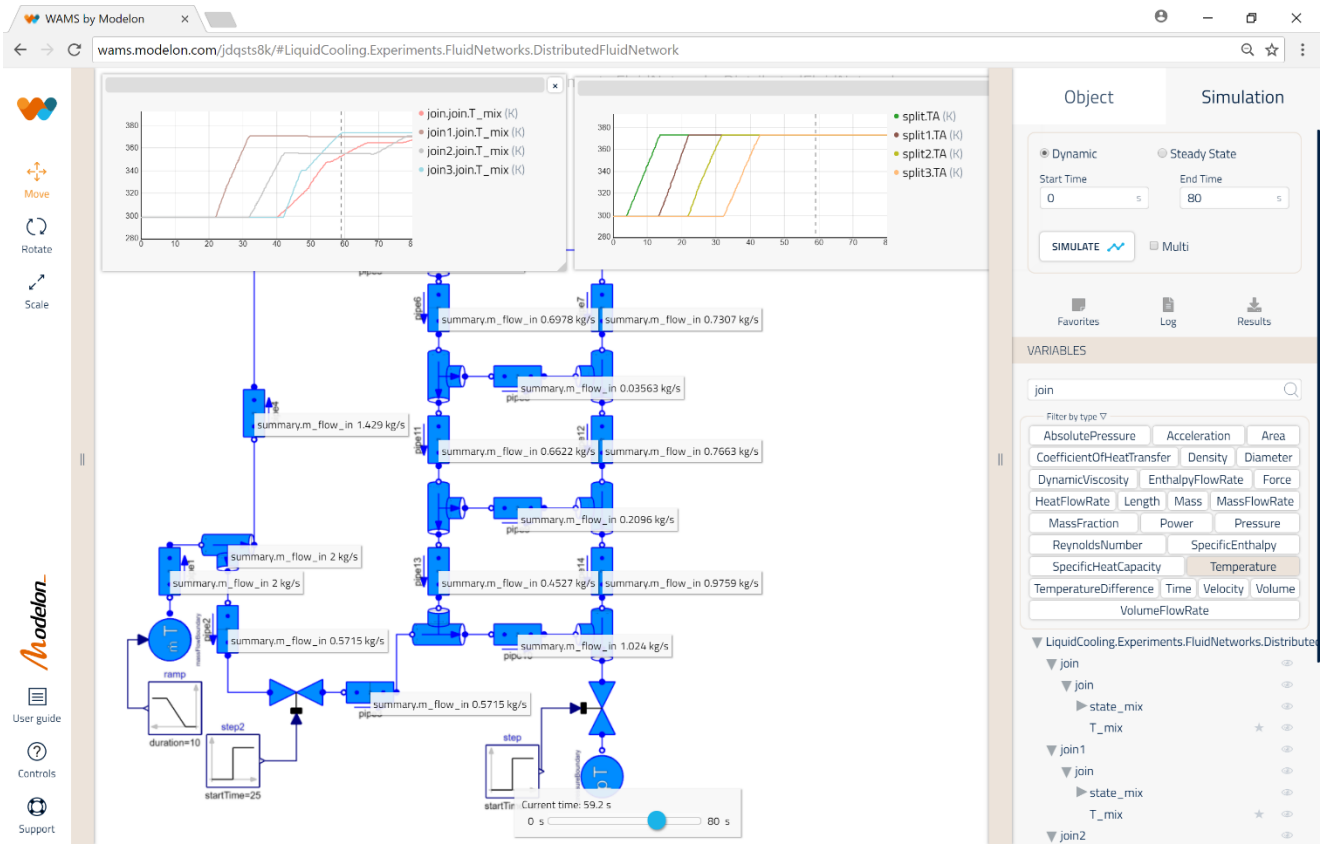

Figure 9. Heat distribution network from Liquid Cooling Library.

Here it is configured to study heat propagation as response to a ramp change of the input flow to the network. The diagram is configured with a plot to see the dynamic temperature change over time.

DOI
10.3384/ecp18148255

To get an overview of how the mass flow is propagated through the system, stickies are activated for each pipe, where the display follows the time slider.

7 Architecture

The software architecture is shown in Figure 10. The web app is written in TypeScript and utilizes the React framework and the three.js 3D package. It communicates with the server using a REST API. The Modeling API has functions for creating, deleting and changing models, instances, connections and variables as well as compiling a Modelica model into an FMU. The simulation API can change parameters, perform multi-simulations and access simulation results.

The server uses the Optimica Compiler Toolkit (OCT) for maintaining the abstract syntax tree of a model being built up in the web app as well as performing the compilation. The simulation API is implemented in Python and uses the PyFMI API (Andersson, et al., 2016). The server hosts the Modelica model repository and the data repository.

The modeling and simulation platform also utilizes Jupyter/JupyterLab notebooks for interactive exploration of models by invoking simulations in the server. JupyterLab can also be used for editing of Modelica code for component authoring. Python scripts can be uploaded to the server to extend the simulation API.

Customized user experiences can be implemented as web apps using the server API functionality and uploaded Python scrips.

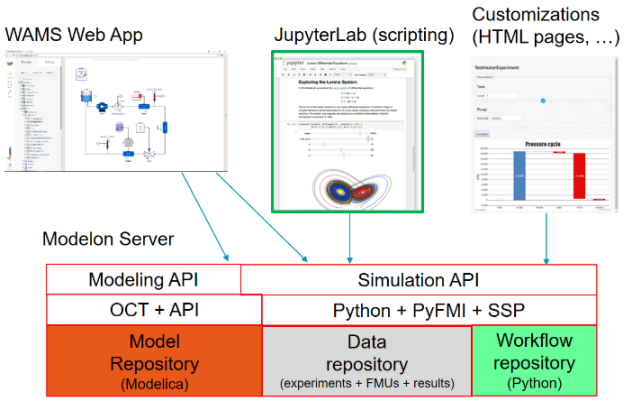

Figure 10. Software architecture

8 Related Work

WebMWorks (Qi, et al., 2012) is a web-based modeling and simulation environment for Modelica. The modeling capabilities are similar to the desktop application MWork Studio. WebGME (Web Generic Modeling Environment) (Maróti, et al., 2014) is a web-based cyberinfrastructure to support the collaborative modeling, analysis, and synthesis of complex, large-scale information systems. It has support for the Modelica language. The Playmola web app (Elmqvist, et al., 2015) is an effort to utilize the power of HTML5 and three.js for Modelica modeling and simulation in a VR environment using Google Cardboard.

Tiller (2013) discusses running simulations on the cloud for sensitivity analysis. Cloud based simulation for FMUs were introduced in Johansson (2015) to speed up simulations initiated by a plugin to Excel. Different parameter cases were defined in columns of the Excel sheet. Bittner, et al. (2015) describes cloud deployment of FMU simulations which are defined in a web app as either enumerated sets of values or ranges of values associated with parameters. All combinations are simulated.

Web apps have also been introduced for CAD, for example TinkerCAD which is one easy to use web app and OnShape and Clara which are professional tools (Ishtiaque, 2017).

9 Conclusions

The paper demonstrates the advantages of using web apps and cloud computing for model-based product design. The presented web app is used for systems modeling and to invoke simulation experiments. The simulations are done on the server which might be installed locally, on premise or on the cloud. Simulations can be distributed on the cloud for an improved user experience avoiding waiting long times for simulation results.

Acknowledgements

The authors want to thank the development teams at Modelon for the excellent implementation effort.

References

C. Andersson, J. Åkesson, and C. Führer: PyFMI (2016): A Python Package for Simulation of Coupled Dynamic Models with the Functional Mock-up Interface. https://lup.lub.lu.se/search/publication/961a50eb-e4a8-43bc-80ac-d467eef26193

J. Batteh (2018): Development and Implementation of the NASA X-57 Electric Aircraft with Aircraft Dynamics Library, Submitted for publication.

S. Berggren (2017): Web Application for Modeling and Simulation, Malmö University. http://muep.mau.se/handle/2043/23515

T. Blochwitz, M. Otter, J. Åkesson, M. Arnold, C. Clauß, H. Elmqvist, M. Friedrich, A. Junghanns, J. Mauss, D. Neumerkel, H. Olsson, A. Viel (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

S. Bittner, O. Oelsner, T. Neidhold (2015): Using FMI in a cloud-based Web Application for System Simulation. Proceedings of the 11th International Modelica Conference September 21-23, 2015, Versailles, France. https://www.modelica.org/events/modelica2015/proceedin

gs/html/submissions/ecp15118845_BittnerOelsnerNeidhold.pdf

H. Elmqvist, A.D. Baldwin, S. Dahlberg (2015): 3D Schematics of Modelica Models and Gamification. Proceedings 11th International Modelica Conference, Versailles, September 21-23, 2015. http://www.ep.liu.se/ecp/118/057/ecp15118527.pdf

J. Ishtiaque (2017): Top 5 Online CAD Tools for 3D Modeling, https://www.linkedin.com/pulse/top-5-cloud-based-online-cad-tools-3d-modeling-jabid-ishtiaque/

D. Johansson (2015): Implementing a cloud-based scalable dynamic model simulator, Chalmers University of Technology. Department of Computer Science and Engineering. Gothenburg, Sweden, January 2015. http://publications.lib.chalmers.se/records/fulltext/212552/212552.pdf

M. Maróti, T. Kecskes, R. Kereskényi, B. Broll, P. Völgyesi, L. Jurácz, T. Levendoszky, A. Ledeczi. (2014). Next generation (Meta)modeling: Web- and cloud-based collaborative tool infrastructure. CEUR Workshop Proceedings. 1237. 41-60. https://webgme.org/WebGMEWhitePaper.pdf

Modelica Association (2014): The Modelica Language Specification, Version 3.3 Revision 1, https://www.modelica.org/documents/ModelicaSpec33Revision1.pdf

Modelon (2018a): Vehicle Dynamics Library: http://www.modelon.com/products/modelon-library-suite/vehicle-dynamics-library/

Modelon (2018b): Liquid Cooling Library: http://www.modelon.com/products/modelon-library-suite/liquid-cooling-library

Modelon (2018c): OPTIMICA Compiler Tool Kit. http://www.modelon.com/products/modelon-creator-suite/optimica-compiler-toolkit/

Modelon (2018d): Liquid Cooling Library: http://www.modelon.com/products/modelon-library-suite/electrification-library

L. Qi, X. Tifan, L. Qinghua, C. Liping (2012): WebMWorks: A General Web-Based Modeling and Simulation Environment for Modelica. Proceedings of the 9th International Modelica Conference, September 3-5, 2012, Munich, Germany. https://pdfs.semanticscholar.org/30a6/c036da254693d416b754867ae0243a8baf23.pdf

M. Tiller (2013): The FMQ Web Platform. Detroit FMI Technology Day, November 6, 2013. https://www.youtube.com/watch?v=5wYvQmqCvBo