# Symbolic Techniques for Model Code Optimization: FMI Applications

*This briefing is a summary of the webcast of the same name on October 14, 2014.*

## Introduction

Since its inception, the Functional Markup Interface (FMI) standard has proven to be invaluable in many projects that design engineers are involved in. It is becoming the primary method for model transfer and co-simulation, providing a smooth, error-free technique for design engineers to deliver their work without manual integration or resorting to other proprietary tools. At the same time, advanced symbolic technologies have emerged that provide extremely fast auto-generated code for the implementation of dynamic system models and enable advanced analysis applications.

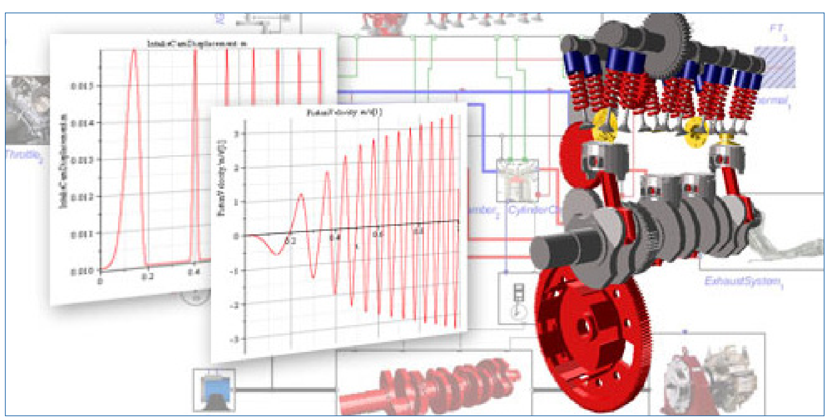
One application in which the FMI standard has been used is the new dSPACE SCALEXIO platform for hardware-in-the-loop (HIL) testing. Engineers can generate a functional markup unit (FMU) of an automatic transmission from Maplesoft's MapleSim software and implement it in SCALEXIO to perform a full-vehicle, real-time simulation.

In terms of the growing demands required for control system design, engineers are developing systems with complex subsystems that all have to interact with each other. If you look at an electric vehicle, or any vehicle, it's a highly complex system with many different domains — mechanical connected to electrical, connected to chemical in the batteries. All these different subsystems have to interact with each other in an appropriate way.

There are increasing demands in terms of customer features — performance, obviously, and how that has to be reconciled against pressures on the automotive industry concerning the environment. In the automotive industry, a lot of software is being put into a product that will be driven at high speed, so safety is of paramount importance.

There is also a proliferation of different design tools that feed into the control design process. Each engineer has to decide which tools are the most important for each part of the process. They then must come up with a way to implement the best in class when developing tools for testing and controllers, and be able to do this by reusing work and not reworking each model every time a new project comes in.

These are the high-level challenges many users are facing, and to help address that, there has been a growth in the use of model-based development tools targeting hardware-in-the-loop (HIL) testing. The notion of being able to develop a virtual prototype of the whole system is playing an increasing role in engineering system design and development. It's been proven, time and again, that it helps reduce

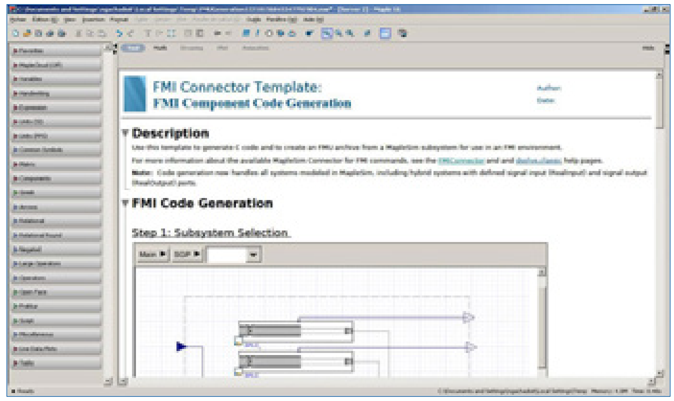


Maplesoft tools provide multi-domain, system-level models through an easy-to-use schematic interface through MapleSim.

prototyping cycles and cost, and helps engineers integrate end user demands while reconciling safety and environmental demands.

Engineers can develop a deterministic and repeatable test platform so they can develop test plans that don't require driving a vehicle around a test track. They can do much of the work in the lab.

Great demand also means that there is great demand for model fidelity — the ability to add further detail into those models. This is one of the areas that has driven developments in software tools — the dilemma between being able to achieve real-time performance and sacrificing a certain amount of fidelity to get that performance. Very often, in terms of complex systems, this can reduce the usefulness of the model. The engineer ends up with something so simple that it isn't a true reflection of what the control system will actually face.



Maplesoft offers the Connector for FMI that allows users to take a model and open up Maple using a template provided to develop the FMU, or Functional Mockup Unit.

## Perfecting Symbolic Technologies

Maplesoft tools provide multi-domain, system-level models through an easy-to-use schematic interface through MapleSim. This is tied very tightly with Maple in order to be able to take those models and do parametric studies and other optimizations to develop tools that can solve many of the issues that become apparent when developing a system-level model.

Maplesoft has been developing symbolic technologies for the past 30 years. This symbolic capability takes the system diagram and generates equations from it. Each component within the MapleSim environment has the internal equations that define the dynamic behavior of that component. Within the MapleSim solver, those equations can be extracted, and because of the way they are connected together through the topology of the model, they can be combined together to develop full system equations of motion.

These models are represented in equation form and are fully parametric. The next step is to apply many different techniques for simplifying those equations. This is what an engineer would likely be doing with a blank piece of paper, and then spending many hours or even days trying to derive system equations by hand. MapleSim does this automatically.

Maplesoft has developed very advanced solvers for differential algebraic equations (DAEs). The representation of these models produces differential equations that have many algebraic constraints incorporated in them. These DAEs, depending on their complexity, can have very high indices. A set of solvers combines numerical and symbolic techniques to reduce the complexity of the system in order to turn them into differential equations.

At some point, an engineer needs to define the inputs and outputs for the system. Everything up to this point has been treated as an acausal representation. Following decisions on inputs and outputs, the equations are resolved accordingly to provide a final set of equations of motion.

It doesn't stop there. An internal co-generation capability has many different symbolically based optimization techniques that can be used for tasks such as characterization of common sub-expressions so they can be pre-computed using trigonometric identities. Many different rules of mathematics can be used to factor out, as much as possible, any computational sequences that don't contribute to the final model.

That is then dropped into a simulation tool like dSPACE system. Fundamentally what this means is that some pretty bold claims can be made about the code generated — it's going to be the fastest possible by virtue of the highly rigorous symbolic steps taken to get there.

Users are able to create models with great detail in them and generate code from the models that are fully optimized and portable across many different platforms to achieve real-time performance where many other tools fail.

## The FMI Standard

The Functional Markup Interface (FMI) standard was developed as part of the European-funded MODELISAR project from the Modelica Association. The primary goal was to develop plant models using Modelica and integrate them into the AUTOSAR controls and system protocol. Since then, FMI has taken on a life of its own. It's been developed, initiated, and headed by Daimler in Germany, and now has been adopted by many different automotive manufacturers and other engineering groups. FMI allows you to prove your software and model HIL simulation from different vendors. It is an open standard developed by a consortium of industrial and academic research partners.

In terms of Maplesoft's support of FMI, the company offers the Connector for FMI that allows users to take a model and open up Maple using a template provided to develop the FMU, or Functional Mockup Unit. Simply by clicking a few buttons and selecting a few options, the software generates the code and enables the user to deploy the FMU into whatever tools are needed for using that work throughout the remainder of the tool chain.

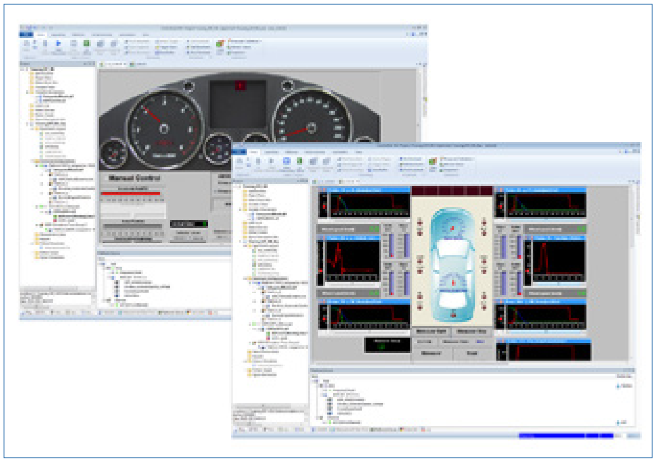

dSPACE offers hardware-in-the-loop architecture through SCALEXIO, which features software-configurable IO and built-in capabilities to support real-world hardware interfaces.

## dSPACE: FMI and Simulation

dSPACE is a global provider of embedded solutions for real time. The company's tools develop, prototype, test, and manage mechatronics control systems and range system modeling and analysis, through functional testing, rapid control prototyping, ECU production, virtual ECU testing, and ECU testing with HIL. Additional tools provide data management throughout that process.

These tools focus on FMI and simulation, and what's involved in simulation, including the back-to-back testing process or validation process where one moves from model-in-the-loop, to software-in-the-loop, to hardware-in-the-loop. dSPACE tools that can help do that include VEOS, a virtual ECU simulation tool. The architecture in the HIL arena is called SCALEXIO, which is a best-in-class HIL system that features software-configurable I/O, built-in capabilities that truly support real-world hardware interfaces, and advanced multi-processor, multicore simulation capabilities for real time.

The key to using these tools throughout the process is the ability to exchange models and provide



ControlDesk from dSPACE lets users define controls, knobs, dials, and gauges to connect to parameters within the model.

seamless access to these platforms. In order to do this, there must be a consistent and open interface like the FMI. It's one way to provide a seamless interface to connect simulation tools together to provide the cohesiveness required to do validation.

Whether using a specific modeling environment or FMI, users must take into account the real-time capabilities of these systems. This can definitely be enhanced with multicore and multi-processing capabilities, but there is always the paradigm that users have to look at in terms of model complexity, solver type, and co-simulation approaches versus tightly coupled modeling environments.

It's also important to take into account platform independence, and whether or not source code needs to be shared as part of a tool or an exchange. There's also the concept of intellectual property protection. FMI provides a capability based on compiled libraries and pre-selection of target platforms to share models between suppliers and OEMs that can provide an advanced approach to handle the types of systems being developed today.

## Tools for FMI

In the simulation arena, physics-based tools offer an opportunity for a greater plant-model capability and simulation capability. Maplesoft is one of the leaders in this area. The FMI standard is in its 2.0 release, and includes HIL and virtual validation. Users can have tunable parameters in the system so they can change parameters during runtime and transport sample times across models, providing greater capability for co-simulation and for simulating systems across either multiple cores or multiple processors.

Areas of FMI 2.0 support provided by dSPACE include FMI for Model Exchange in which models are exchanged and use a common solver technology. Tightly coupled subsystems are connected at the equation level, and are used with a single solver technique.

FMI for Co-Simulation is used when an FMU is exchanging a model in its own system with a solver included. This is a more loosely coupled technique for sharing models, but it's more flexible in terms of its ability to support diverse modeling environments and different types of simulation systems like dSPACE real-time systems. Once a system is running, FMI for Applications addresses how to use that model and how to interface to it. FMI for Product Lifecycle Management is used for sharing models and managing the model environment.

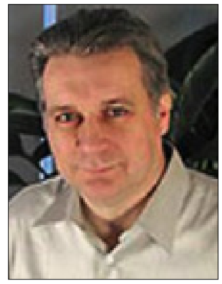
When building FMUs for HIL simulation, dSPACE has a tool called ConfigurationDesk, which provides the ability to design the total signal chain of the interface of a system used for HIL testing. Users can define everything from I/O functions, to conglomerations of I/O functions, to the model interfaces that are part of the system. Model subsystems can be combined whether they connect to I/O or not. ConfigurationDesk also address whether the models run on a single core, across multiple cores, or even across multiple processors.

Once the connections of I/O and models in a system are mapped, the next step is to configure the tasks of a system. Different FMUs can be placed within different tasks that run at a timed or periodic rate, or they can be dependent on tasks that are part of the system. The real-time constraints in the architecture of the system are defined as it executes in real time. When the HIL system is running, users can access it via a GUI or a manual user interface.

ControlDesk lets users define controls, knobs, dials, and gauges to connect to parameters within the model. MotionDesk allows a user to animate a system, and AutomationDesk enables automated testing of a system.

FMI is an increasingly important standard for engineering system design, testing, and integration. Symbolic technology is proven physical modeling technology that significantly improves model fidelity without sacrificing performance. Maplesoft and dSPACE have proven rapid plant model development for HIL testing using FMI and the SCALEXIO platform.

## Authors/Presenters



Paul Goossens is a mechanical engineer with more than 20 years of experience in both engineering and software business management. Previous positions included senior management for companies in engineering modeling solutions and high-performance real-time simulations. He has built a strong reputation as an expert in real-time applications, particularly in high-fidelity simulation of mechatronic systems for HIL applications, spending a good part of his career promoting model-based methodologies within the automotive industry.

Jace Allen is an electrical engineer in systems engineering at dSPACE Inc., having designed and managed hundreds of hardware-in-the-loop (HIL) system implementations for various customers. In the past 25 years, he has handled many diverse modeling, controls, and simulation test applications in the automotive, commercial vehicle, and aerospace areas. His background includes model and core product development for vehicle controls/testing, safety/security systems, and data management systems.

**View the original webcast at https://event.webcasts.com/starthere.jsp?ei=1041930**