From Physical Modeling to Real-Time Simulation : Feed back on the use of Modelica in the engine control development toolchain

Zakia Benjelloun-Touimi[1]; Mongi Ben Gaid[1]; Julien Bohbot[1]; Alain Dutoya[2]; Hassan Hadj-Amor[1]; Philippe Moulin[1]; Houssem Saafi[3]; Nicolas Pernet[1]

[1] IFP Energies nouvelles
1 & 4, avenue de Bois-Préau, 92852 Rueil-Malmaison Cedex, France

[2] INCKA SA
85 avenue Pierre Grenier, 92100 BOULOGNE-BILLANCOURT

[3] Ecole Nationale d'Ingénieurs de Sousse
Technopôle de Sousse, 4054 Sousse, Tunisie

Abstract

This article provides feedback from using Modelica in the "System Modelling" area, involving modelling (behavioural and dynamic modelling), direct simulations, control and real-time applications.

The described work was undertaken within three Europeans projects: Eurosyslib, Modelisar and Open Prod.

Our aims are to attest Modelica language in an overall model of a vehicle consisting of vehicle dynamics, combustion engine, transmission, drive line, brakes and control systems.

ModEngine is a complete IFPEN[1] library, resulting from our participation in those European projects. It allows the modelling of a complete engine with diesel and gasoline combustion models. It may be interfaced with control algorithms written in Simulink thanks to the new Functional Mock-up Interface specification from Modelisar project.

Both versions under commercial software Dymola and free one OpenModelica are available.

Feedback will concerns also problems encountered and advantages in use Dymola and OpenModelica platforms.

Keywords: Control, Eurosyslib, FMI, Library, Modelica; Modelisar; ModEngine, Modelisation, Openprod, Simulation, Real-time.

1 Introduction

The use of the Modelica™ language [19] for hierarchical physical systems modelling is booming thanks to an international effort, but mainly because it meets what engineers and researcher expect for their development today.

In this paper we propose to provide a feedback about our experience in using Modelica for an industrial application in European projects (Eurosyslib [23], Modelisar [24], and OpenProd [25]): the development of control strategies for automotive engines. In this field, IFPEN [22] has promoted an approach where simulation tools play a crucial role at the different stages of the development process. This approach has been described in various publications. It utilizes the modelling library IFP-Engine developed in C language, commercially available under the LMS Imagine.Lab AMESim [26] environment.

The different stages of the development process and the associated needs for simulation are the following:

1. System understanding: a detailed model of the system is needed, including the representation of all the physical phenomena. The physical accuracy is important.
2. Control strategy development: the focus is on the control part, the system model is not modified. However, it is coupled with the control strategies, executed in Simulink. The execution platform is important.

[1] *IFPEN : IFP Energies nouvelles, new name of IFP*

3. Control validation: the model must be compiled and executed in an environment that is representative of real-time. The ability to compile and export the model is important.
4. System integration: the interactions between the engine and the other parts of the vehicle (transmission, after treatment,) are considered, the engine model and its control must be coupled to other component models, developed in various modelling environments. The model integration capabilities are important.

The plan of this paper will follow the structure given by these different stages. Each of them will be described in detail in a different section that details the design choices, the advantages and drawbacks of Modelica in this context.

2 System understanding: development of ModEngine library

Requirements for the ModEngine library were derived from the existing IFPEN AMESim library (Engine). The users can quickly assemble blocks that result in vehicle simulators. ModEngine is now functional in Dymola [20] and OpenModelica [21]; it contains more than 250 sub models. We continue to contribute and to optimize it in order to obtain accurate and fast calculations for control and real-time applications, respecting the procedure described by *figure 1*below.

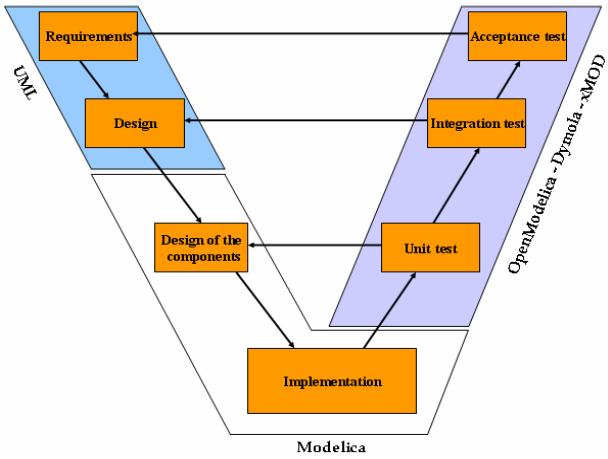


Figure1: Modelling /Simulation/Control/Real-time validation Cycle

The ModEngine library has been developed to allow the simulation of a complete virtual engine using a characteristic time-scale of the order of the crankshaft angle. A variety of elements are available to build representative models for engine components, such as turbocharger, wastegate, gasoline or Diesel injectors, valve, air path, EGR loop etc... Figure 3 shows these ModEngine components. Moreover, the library uses an advanced modelling approach to take accurately into account the relevant physical phenomena taking place in the engine [5]. The computed gas consists of 3 species: fresh air vaporized fuel and burnt gas. Thanks to the object oriented language Modelica, it can be automatically extended to n-gases for future development of the library. Generally, 3 gases is a thermodynamic assumption that has been identified as sufficient for engine simulation results. More than 3 gases, generally 12 gases have to be used for pollutant emissions modelling. Main relevant orifice and pipe elements are available such as air throttle, valve, straight and kneed ducts. Friction and inertial effects can be also taken into account. Heat exchanges are modelled for each element and specific heat exchanger models are also available with air and water cooling systems. An ideal camshaft system is proposed. Inlet and exhaust valves are piloted by valve lift trajectories and cross section characteristics. The elements to build most of the turbocharger technologies are proposed. The modelling approach is based on turbocharger manufacturer's maps.

Concerning the combustion process, a first level of modelling is available with an empirical model based on the Wiebe's law [2]. Generally, this model is used through a mapping of the combustion phenomena based on experimental cylinder pressure; the coefficients of Wiebe's law are calculated defining a map covering the engine operating conditions. This combustion model is a mathematical based model that gives an evolution law for the heat release. 4 coefficients are needed to fit the model on experiment pressure signal for each phase of the combustion. Generally, due to their simplicity, these models are used for real-time applications. The parameters of this function are optimized and mapped according to experimental results. Bohbot *et al.* [8] have used a simple Wiebe law, coupled with an automatic tool to create the parameter maps using both experimental and 3D CFD results. This model can be use either gasoline or Diesel engine simulation. For Diesel simulation, a double Wiebe equation is generally used.

The second level of modelling is given by efficient phenomenological models. For the spark ignition engines, the CFM-1D model is used [3]. This com-

bustion model is based on the CFM combustion model [6] developed at IFPEN in the 3D code IFP-C3D [7]. The coherent flame model (CFM) is a combustion model adapted to the flamelet regime for premixed mixtures. This approach is representative of the premixed flame combustion, which represents the main oxidation mechanism in spark ignition (SI) engines. To calibrate this model, 6 different physical coefficients must be defined to calculate the initialization of the turbulence, the dissipation of the turbulence, the turbulence mixing scale, the flame wrinkling, the flame initial volume and the tumble value. The first 5 are constant coefficients, while the last one defining the tumble coefficient value can be defined as a function of the volumetric efficiency of the engine.

For Diesel engine, an advanced Barba [1] model developed at IFP is implemented in ModEngine [9]. The Barba's model can reproduce the conventional Diesel combustion process, using only 2 zones (a first zone for the description of the pre-mixed combustion and a second one for the diffusion mode). With a reduced number of parameters, it can be used for a wide range of operating points. In this model, the combustion process is divided in 2 steps. In a first step, the fuel is burnt using a premixed model with the hypothesis of flame propagation in the pre-mixed zone. In a second step, when the pre-mixed zone is burnt, the remaining fuel is oxidized using a mixing controlled combustion model. The different hypothesis and equations of the Barba's combustion model are presented in [10].

Cylinder wall thermal exchanges can be taken into account following Woschni models [4]. Injection models allow governing the injected fuel mass rate using maps or algebraic functions. The fuel can be injected in gaseous phase or in liquid phase. The vaporization process is governed by a characteristic timescale for Direct or Port Fuel Injection.

As shown in Figure 3, the ModEngine library contains 22 different packages and at least 250 different models.Figure 4 shows a direct injection single-cylinder modelling and a Mean Value Engine Modeling with the ModEngine library. All models have been validated with dedicated test cases to ensure the non regression of each component (Figure 5) and with functional tests to validate the whole library. The validation of elementary sub models has been done using as reference results the IFP-Engine library developed by IFPEN in the LMS Imagine lab Platform AMESim, functional validation of complete engines, comparison with experimental data, steady state and transient data from test campaigns made at IFPEN.

Figure 2 shows a numerical comparison obtained with ModEngine and IFP-Engine using the Diesel combustion model (Barba) on a one-cylinder Diesel Direct injection engine that validates the good implementation of the Barba model in the Modelica language.

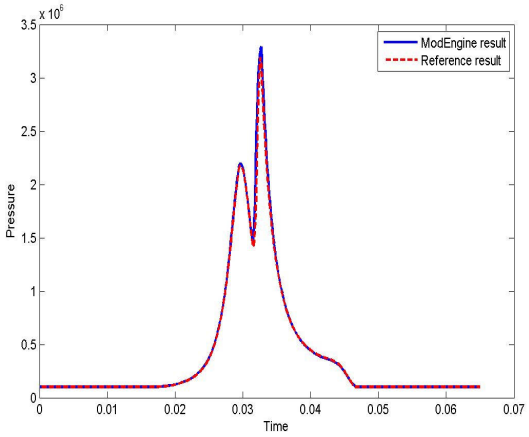


Figure 2: In-cylinder pressure comparisons obtained with Barba Model between ModEngine / Reference (IFP-Engine).

To connect different components, 9 connectors have been implemented in ModEngine. These connectors allow the connection of mechanical part, liquid flow, gaseous flow and thermal flow.

For instance, the connector which links the air path to the cylinder chamber is the PFlowPort connector. Enthalpy and mass flow rate with the mass fraction are defined as input and the output are containing the thermodynamic state and the partial densities.

```
connector PflowPort "Input Pflow Port"
    parameter Integer ngas = 3 "Gaz number";
    input SI.EnthalpyFlowRate dh "input enthalpy flow rate";
    input Real dm "input mass flow rate [kg/s]";
    input Real x[ngas] "input mass fraction vector [null]";
output SI.Temperature temperature "output temperature";
ouput SI.Pressure pressure "output pressure";
ouput Real rhoOut[ngas] "output density vector [kg/m**3]";
end PflowPort;
```

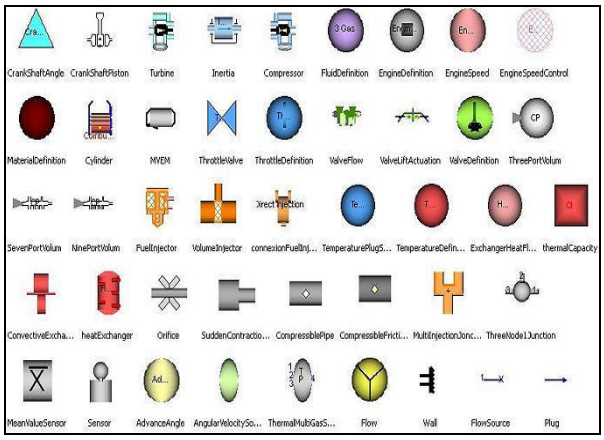


Figure3: ModEngine library

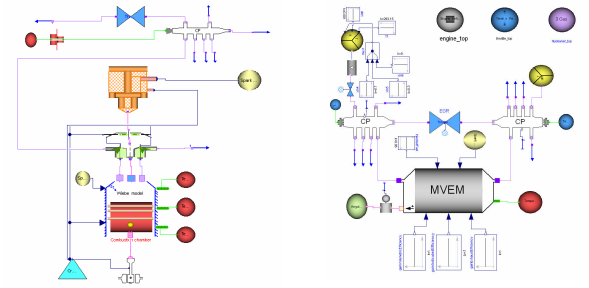


Figure 4: Single cylinder engine and MVEM engine models.

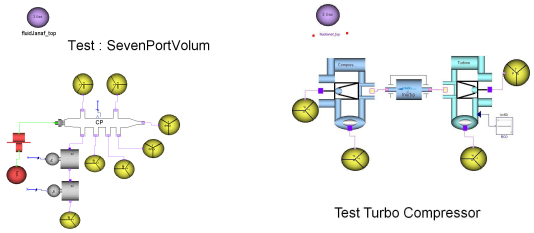


Figure5:
Non regression test case for each component

3 Control development

For the development of engine control strategies, models representing the complete engine are needed. The ModEngine library described above can be used to design such models in Dymola. This first subsection below shows validation results for a complete engine model.

Furthermore, the multiplatform capabilities of Modelica language can be very interesting from a cost point of view, because it allows using a different environment for the execution of the model at this stage, than that used for the model design. In the following subsection the library results obtained with different platforms are compared.

3.1 Complete engine model validation in Dymola

The engine considered here is a four cylinder gasoline engine with fixed geometry turbocharger. Its model is shown in figure 6. The following approach has also been undertaken for Diesel engines, though it is not exposed here for lack of space.

Two types of tests have been performed: steady state and transient. In steady state the model results are compared with experimental measurements for operating points covering the whole engine range. For transient tests the comparison is made on a driving cycle measured on the test bench.

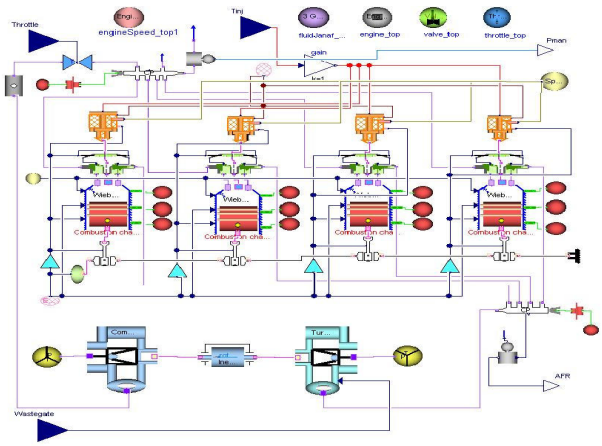


Figure 6 : complete engine model in ModEngine

Steady State tests

The model results are compared for various operating points with experimental measurements. These operating points are defined by :

- engine speed
- intake manifold pressure

The following variables are controlled to setpoints :

- intake manifold pressure is controlled by either the throttle or the wastegate to the setpoint defined by the operating conditions
- the air fuel ratio is controlled at stoichiometry by the injected fuel mass

The thermodynamic conditions along the air system and cylinder are model outputs. The following figure shows a good match between the torque obtained experimentally and with the model. A comparison of

the thermodynamic conditions along the air system would show similar results.

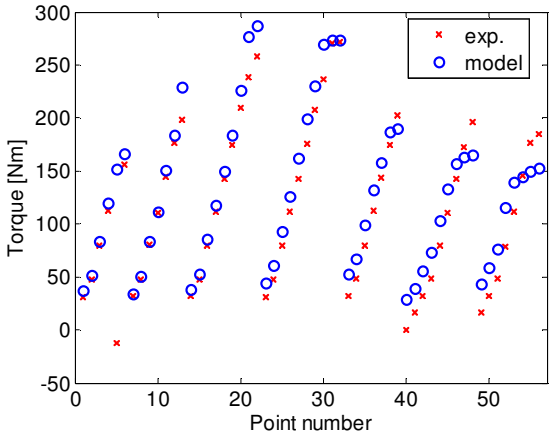


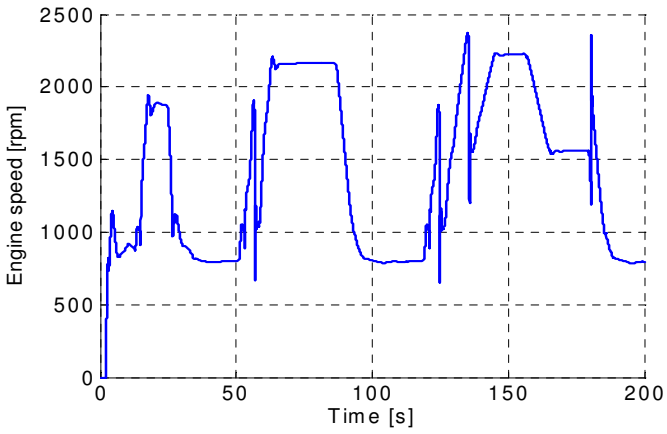
Figure 7: The torque match

Transient tests

The model is now compared with transient results. It is plugged to engine control software, running in co simulation in Simulink. The inputs of the model are:

- engine speed
- torque setpoint

The engine control software determines in closed loop the commands of the engine (actuators positions: throttle and waste gate, injection timing, spark advance) based on the sensor values received from the model. The results are shown in the figure 8, enhancing again the good behavior of the simulator with respect to experiments.



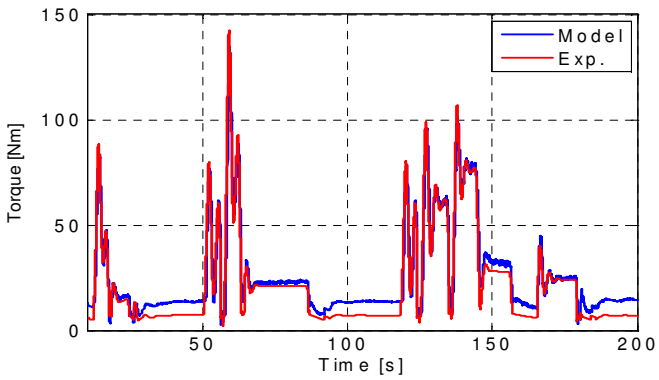


Figure 8

3.2 Comparisons: Dymola/Openmodelica

All the elementary models developed in Dymola were translated in Openmodelica. The translation was not immediately done and needed some reshuffle. Comparisons used a fixed and variables step integrators (Runge-Kutta / Euler; Dassl, /Dassl2).

The main variables which were selected to verify the accuracy of the results using Dymola and OpenModelica are the temperature, the pressure in the cylinder (figures 9, 10, 13), and the Dissipative kinetic energy (figures 11, 12, 14).

We show following some comparisons between Dymola and OpenModelica for 3 combustions models.

Wiebe model

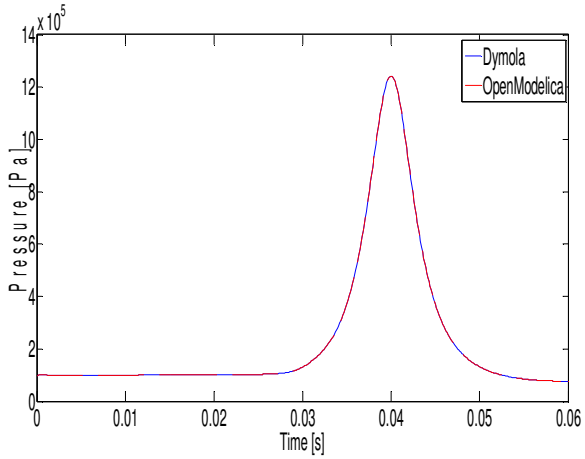


Figure9: Cylinder Pressure

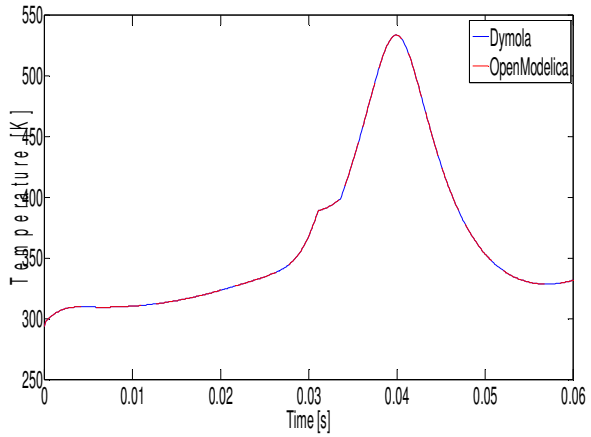


Figure10: Cylinder temperature

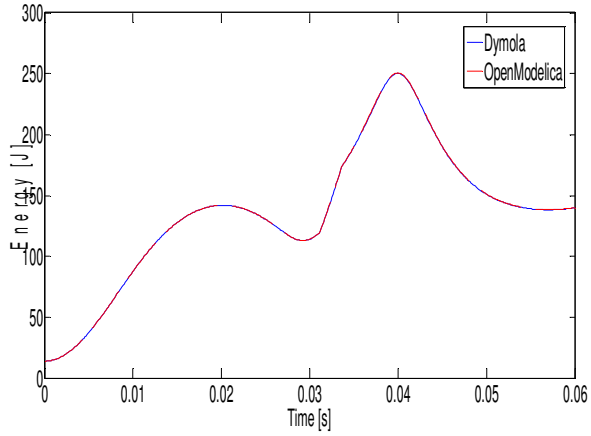


Figure11 : Energy model

CFM model

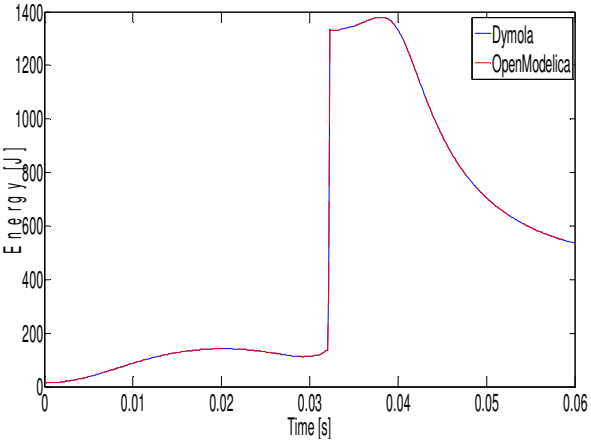


Figure12: Energy model

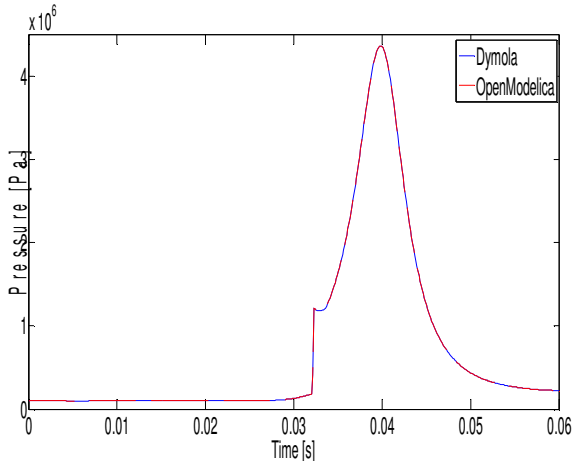


Figure13: Pressure model

Barba model

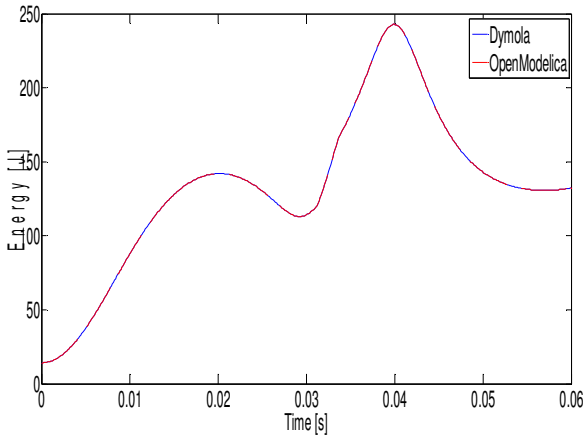


Figure14: Energy model

3.3 Conclusions

ModEngine simulation results are performed respecting the same integration conditions using the two platforms Dymola and OpenModelica.

Models contain 683 variables and equations; with 235 zeros crossing and without numerical jacobians.

For the same final time simulation and integrators having the same tolerances it seems that Dassl and RungeKutta in Dymola take acceptable equivalent time, while integration with OpenModelica with both fixed or variable step takes more longer time.

The two platforms give similar results; the errors are less than 1%.

4 System Integration and control validation

The multidisciplinary aspect of complex systems leads to use different tools for the design step. That's why the simulation step requires co-simulation techniques in order to exchange data between simulators. The concept of functional mockup provides advancement to model exchange during the product design and validation cycles. Using models through their interfaces allows hiding their implementation details and making their usage easier.

4.1 xMOD

xMOD [12] is a platform which combines an integration environment for various heterogeneous models, together with a virtual test laboratory. xMOD offers a range of different functionalities, such as the integration of heterogeneous models (Simulink, AMESim...), confidentiality management for models when they are imported, virtual instrumentation, test automation, etc. The purpose of xMOD is to make it possible for models to be used by people other than those who created them, and for them to be shared. xMOD provides simulation functions in various simulation schemes: real-time, extended time or as soon as possible. Its execution kernel can be used to process various integrated models in multiprocessor and multicore. xMOD is built around the following key ideas:

- Using a unified representation of all heterogeneous models that is simple and complete enough for them to be integrated and co-simulated, and for the expertise that they contain to be protected.
- Abstracting the modelling language through a virtual instrumentation, such that the models can be easily understood by people other than those who created them, or by people who do not have knowledge of the languages in which they were written.
- Focusing on using the models (they are always built in the usual modelling environments), and providing ergonomically-designed features for interacting with the simulations, running the tests and using the results.

4.2 FMI

The ITEA2 project MODELISAR is providing solutions enabling the integrated design, test and management of automotive systems. One result of this project is a new open Functional Mockup Interface (FMI) to support co-simulation between simulation tools, in particular Modelica, for system modelling and AUTOSAR for embedded control software generation [24]. The FMI specifies C and XML interfaces for dynamic systems to be used as an interchange format between different tools. This interface is to be implemented by an executable called FMU (Functional Model Unit). The FMI functions are called by a simulator to create one or more instances of the FMU, called models, and to run these models, typically together with other models. An FMU may either be self-integrating (co-simulation) or require the simulator to perform numerical integration. The FMI goal is to describe models of dynamic systems which are, in general, described by differential, algebraic and discrete equations with time, state and step events. The interface is designed so that large models can be described and consists of the following two parts: A model interface: All needed equations are evaluated by calling standardized C functions. A model description schema: All variables in the model are defined in a standardized way in a XML file. The C-code could then be executed in an embedded system without the overhead of the variable definition.

4.3 Integrating FMI in xMOD

In order to extend the capabilities of our co-simulation tool we chose to integrate the FMI functionalities to support more heterogeneous models coming from Modelica based tools. This extends the capabilities of xMOD allowing it to integrate models (in the form of FMUs) coming from various Modelica compatible authoring tools like AMESim Dymola, SimulationX, Simpack...

To integrate FMI-for-cosimulation functionalities in xMOD, we opted for the wrapper approach. This solution is applicable for tools offering library interfaces with the ability to call functions or methods. In xMOD, each instantiated model has its library interface providing common generic functions to evaluate the model dynamic equations. To integrate FMI functionalities, we chose to develop a wrapper library whose main purpose is to load FMU models and call their FMI functions. All the FMU model functions are wrapped in the common generic functions. The class diagram below shows the part of the design of the FMU wrapper. The common generic functions are listed in the xMODFMU wrapper class. Based on this approach, we succeed to make xMOD FMI compatible.

4.4 Control validation

Before their implementation in the actual electronics unit, it is necessary to validate the control strategies in a "real-time representative" simulation environment. Thanks to xMOD and to the implementation of the FMI concept, it is possible to build global co-simulations involving the physical models (from ModEngine), imported in xMOD as FMUs, as well as the control strategies, which are usually developed in Simulink, and which are imported in xMOD using the xMOD Target for Real-Time Workshop [14]. This global co-simulation may be used for control laws parameters tuning and pre-calibration as well as closed loop system performance assessment. (Figure 15).

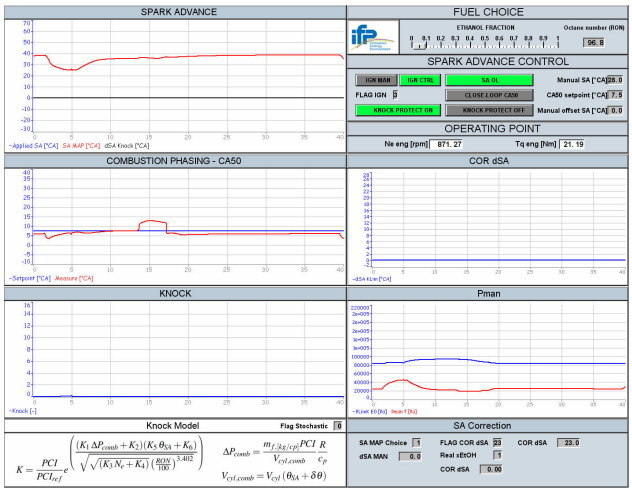


Figure 15: Screen shot from xMOD execution window illustrating the validation of a co-simulation platform integration an FMU of a flex-fuel combustion engine, which was generated from Dymola, running together with Simulink models (controls) and AMESim models (vehicle Dynamics)

5 Perspectives: real-time simulation

Real-time simulation of a single model is a non-sense. Real-time simulation is needed when models are supposed to interact with physical part of the whole system, for example when coupling simulated and real components in a Hardware-in-the-Loop (HiL) process. Indeed, this process is representative and successful only if components are unable to distinguish if others components are real or simulated. It implies that a simulated component must have the same timing behaviour than its corresponding real component. Real-time constraints are consequently inherited from the needed data exchange between components. For example, let consider we want to connect our engine models to a real hardware controller which acquire sensor data and send its actuators command every 500µs. To make possible this HiL process we must ensure that our model simulation accuracy is sufficient to always be able to simulate 500µs of engine behaviour in less than 500µs of real-time. Notice that verifying that you can simulate 30 seconds of the engine behaviour in less than 30 seconds of real-time is not sufficient to guarantee the previous requirements. Indeed, in the HiL process, even if 500µs of engine behaviour are simulated in less than 500µs, the engine simulation cannot go on before the end of the 500µs period in order to receive controller data.

Consequently, improving performance for Madelia models is a necessary condition for simulations to reach real-time. Engineers often think about improving their models efficiency at the end of the design phase. This leads often to non efficient simulations. In [13] we showed that the needs for efficiency should be considered as soon as the modelling step starts. A set of general methods based on a closer view on the Modelica's modelling and simulation processes are presented to give hints to the designers in order to reach real-time requirements.

The Functional Mock-up Interface could also help to gain in efficiency for Modelica models. Indeed, the FMI and especially, the FMI for Model Exchange, gives freedom to the user to handle model's execution in different ways. For example, the FMI does not enforce any predefined event handling mechanism like the one provided by Dymola or other Modelica tools.

6 Modelica contribution and future works

We end this article on the observations and reflections on the use of language and focused on a return of the technical problems [15] that in fact open to other possibilities to extend the language.

Today we are participating intensively in various activities on Modelica. Among the areas for future work , we will give the possible directions we intend to take .

6.1 Dymola and OpenModelica implementation feedback

The goal with the OpenModelica effort is to create a comprehensive Open Source Modelica modeling, compilation and simulation environment based on free software distributed in binary and source code

for research, teaching and industrial usage. However, we think that for this latter case, Dymola is ahead of OpenModelica tool. Until today, we notice that OpenModelica doesn't support yet all Modelica specifications. Thereby, the original models have been to be depreciated to run correctly. We noticed also that the OpenModelica fixed step solver is much accurate then Dymola fixed step solver. For example, the figure 15 compares the RungeKutta 4 solver of the two platforms against the Dassl solver which is taken as the reference:

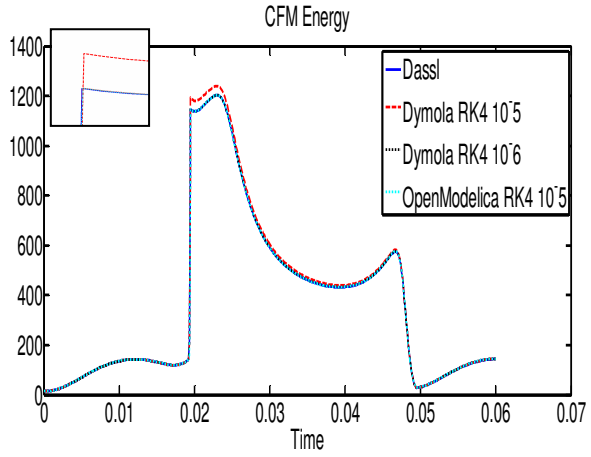


Figure 15

Finally, we can notice also that Dymola provides unique support for real-time and hardware-in-the-loop simulation (HILS) instead of the actual version OpenModelica.

6.2 Future directions

LMS [31] and INRIA [27] have developed a compiler based on Modelica, named Modelicac[18], during the SIMPA2-C6E2 project [16, 17], which is still in progress. Our aims in the near future is to both continue to test Modelica langage on different platform AMESim/ScicosLab, OpenModelica and Dymola, in order to have heterogenous, interoperable and multiplatform librairy. We will continue our involvement in European projects, more particularly on the control, FMI , xMOD and real-time applications involving Modelica language.

Key elements of the long-term orientation of our work is the success of European projects. Indeed, if it turns out that Modelica provides benifit and is the "standard" recognized by the industry, several fields of engineering systems will adopt Modelica language.

References

[1] Barba C., Burkhardt C., Boulouchos K., Bargende M. (2000) A phenomenological combustion model for heat release rate prediction in high speed DI Diesel engines with common rail injection, *SAE Technical Paper* 2000-01-2933.

[2] Wiebe, I.I., "Semi-empirical expression for combustion rate in engines", Proceedings of Conference on piston engines, *USSR Academy of sciences, Moscow, pp. 186-191, 1956.*

[3] F-A. Lafossas, O. Colin, F. Le Berr, P. Menegazzi, “Application of a new 1D combustion model to gasoline transient engine operation”, *SAE 2005 Fuels and Lubricants Meeting Exhibition and Congress, May 11-13 2005, Rio de Janeiro, Brazil - SAE 2005-01-2107.*

[4] Woschni G., "Universally Applicable Equation for the Instantaneous Heat Transfer Coefficient in the Internal Combustion Engine", *SAE paper 670931, SAE Trans., vol. 76, 1967.*

[5] Menegazzi, P., Aubret, P., Vernhes, P.-L., "Conventional and Hybrid Vehicle Emission, Fuel Economy and Performance Analysis System Simulation", *FISITA 2004, 23-27 May, Barcelona, Spain*

[6] Colin, O., Benkenida, A., Angelberger, C., "A 3D Modelling of Mixing, Ignition and Combustion Phenomena in Highly Stratified Gasoline Engines", *Oil & Gas Science and Technology, vol. 58, pp. 47-62, 2003*

[7] "IFP-C3D: an Unstructured Parallel Solver for Reactive Compressible Gas Flow" J. Bohbot, N. Gillet, A. Benkenida, *Oil Gas Sci. Tech., 64 (2009), 309-336*

[8] Bohbot J., Lafossas F.-A., Miche M., Chraibi M., Menegazzi P. (2004) A new coupling approach using a1D system simulation software and a 3D combustion code applied to transient engine operation, *SAE Technical Paper* 2004-01-3002.

[9] F.-A. Lafossas, M. Marbaix and P. Menegazzi "Development of a Coupling Approach between 0D D.I. Diesel Combustion and Pollutant Models: Application to a Transient Engine Evolution" *Oil & Gas Science and Technology - Rev. IFP, Vol. 63 (2008), No. 4, pp. 479-494*

[10] Barba C., Burkhardt C., Boulouchos K., Bargende M. (2000) A phenomenological combustion model for heat release rate prediction in high speed DI Diesel engines with common rail injection, *SAE Technical Paper 2000-01-2933.*

[11] Modelisar Functional mock_up interface for model exchange version 1.0. Technical Report 07006, *MODELISAR consortium, January 2010*.

[12] M. Ben Gaïd, G. Corde, A. Chasse, B. Léty, R. De La Rubia, M. Ould Abdellahi. Heterogeneous Model Integration and Virtual Experimentation using xMOD: Application to Hybrid Powertrain Design and Validation In Proc. *7th EUROSIM Congress on Modeling and Simulation, Prague, Czech Republic, September 2010.*

[13] H. Hadj-Amor, C Faure, M. Ben Gaïd, N. Pernet, "Towards a Modelica Real-time co-simulation with FMI", Multiphysics Simulation - Advanced Methods for Industrial Engineering Conference, Fraunhofer, 22-23 June 2010.

[14] Coppin Thomas, Grondin Olivier, Le Solliec Guenael, Maamri Nezha, Rambault Laurent. "Control-oriented mean – value model of a fuel-flexible turbocharged spark-ingnition engine". SAE World congress, Detroit USA, 13-15 april 2010.

[15] Benjelloun Zakia; Moulin Philippe, Najafi Masoud, Shen Xduong. "Simulation of the mean-value internal combustion engine in modelica" MathMod International Conference on Mathematical Modelling, Vienna, Austria, 11-13 February 2009.

[16] Benjelloun Zakia, Najafi Masoud. "Using modelica for modelling and simulation of spark ignited engine and drilling station in IFP". International modelica conference, Bielefeld, Germany, 3-4 march 2008.

[17] Benjelloun Zakia, Najafi Masoud. "Modelling complex system with modelica in scicos: application to mean value spark engine". ESM European Simulation and Modelling conference, St.Julian's, Malta, 22-24 October 2007.

[18] Masoud Najafi, Ramine Nikoukhah, Serge Steer, Sebastie Furic. "New features and new challenges in modelling and simulation in Scicos" Proceedings of the 2005 IEEE Conference on Control Applications. Toronto, Canada, August 28-31, 2005.

[19] http://www.modelica.org/

[20] http://www.3ds.com/products/catia/portfolio/dymola

[21] http://www.openmodelica.org/

[22] http://www.ifpenergiesnouvelles.fr/

[23] http://www.eurosyslib.com/

[24] http://modelisar.org/

[25] www.openprod.org

[26] http://www.amesim.com/

[27] http://www-rocq.inria.fr/scicos/

[28] http://www.pme.gouv.fr

[29] http://www.itea2.org/

[30] http://www.ida.liu.se/labs/pelab/modelica/OpenSourceModelicaConsortium.html

[31] http://www.LMSINTL.com

Thanks

This work was realized thanks to the labeling of the projects Eurosyslib, Modelisar, and OpenProd by the European Organisme ITEA 2 (Information Technology for European Advancement)[29] and, thanks to financial support from DGCIS (Direction Générale de la Compétitivité, de l'Industrie et des Services)[28].

IFPEN is an OSMC [30] member and thus we benefits from fruitful discussions during the adaption of the library ModEngine in OpenModelica