A MODULAR MODEL ARCHITECTURE IN MODELICA FOR RAPID VIRTUAL PROTOTYPING OF CONVENTIONAL AND HYBRID GROUND VEHICLES

John J. Batteh, PhD
Emmeskay, Inc.
Plymouth, MI

Michael M. Tiller, PhD
Emmeskay, Inc.
Plymouth, MI

ABSTRACT

This paper presents a flexible, modular model architecture in Modelica for system modeling and simulation of military ground vehicles. The model platform and implemented interfaces are flexible enough to support virtual prototyping of conventional and hybrid vehicles with various physical architectures such as series, parallel, hydraulic, and plug-in implementations. Several sample model implementations of conventional and concept hybrid military ground vehicles are presented to illustrate the usage and flexibility of the model architecture to support systems engineering activities by maximizing model re-use throughout the product development process from concept assessment and design through testing and verification.

INTRODUCTION

Model-based systems development for commercial and military vehicles has proven effective in reducing development time and increasing product quality and reliability. ADVISOR [1] was one of the first widely-used tools to support system-level analysis of vehicle powertrains. More recently tools such as PSAT [2], developed by Argonne National Laboratory for commercial vehicles, and VPSET [3], developed by TARDEC and Southwest Research Institute for military vehicles, provide configurable platforms with associated model implementations for fuel economy and performance simulations with conventional and hybrid propulsion systems. ADVISOR and PSAT have been implemented in MATLAB/Simulink platform. These tools offer either forward looking (VPSET, PSAT) or backward facing capability (ADVISOR).

This paper presents a flexible, modular model architecture in Modelica [4] for system modeling and simulation of military ground vehicles. The model platform and implemented interfaces are flexible enough to support virtual prototyping of conventional and hybrid vehicles with various physical architectures such as series, parallel, hydraulic, and plug-in implementations. The architecture supports models of varying levels of details at multiple levels in the model hierarchy so that the same model architecture may support engineering activities over the entire systems engineering V. By using a common framework and promoting substantial model reuse with localized model refinement, implementations derived from this single model architecture can support a range of model-based engineering efforts including high level concept assessment, requirements-driven component and system design, control system design, and associated verification and validation (V&V) activities. Plug-n-play capability enabled by formal Modelica language constructs at the system, subsystem, and component level allows rapid model configuration of different vehicle implementations. The architecture includes representations for the driver, environment, control system, and vehicle physical system. Expandable elements within the architecture can support multi-voltage electrical buses, electricification in all major subsystems, distributed controller implementations, and thermal modeling throughout the vehicle system including the cabin.

Several sample model implementations of conventional and concept hybrid military ground vehicles are presented to illustrate the usage and flexibility of the model architecture and compatible multi-domain component model libraries to support systems engineering activities throughout the product development process. Sample analyses include fuel economy assessments over mission profiles, vehicle attribute evaluation to targets, and detailed energy accounting based directly on the physical model implementation. Extensions of the implementations to support vehicle drivability, NVH, and energy/thermal management are also discussed.

MODELICA

Before discussing the details of the model architecture, a short introduction to the Modelica modeling language is provided. Modelica [4] is a non-proprietary, object-oriented, high-level modeling language that supports continuous and discrete differential algebraic equations (DAE). Modelica supports both causal and acausal modeling and is especially suited for multi-domain physical modeling due to its connector-based formulation. With a familiar physical representation of physical system components, Modelica is a key enabler for efficient, first principles modeling and model-based systems development (MBSD). A few important features of the language are as follows:

- Non-proprietary nature of the language supports tool neutrality and competition while avoiding vendor lock-in
- Natural physical modeling via connectors with built-in support for algebraic/kinematic constraints (DAEs)
- Causal and acausal representations support a range of controls and physical modeling formalisms
- Equation-oriented modeling allows symbolic processing and optimization for efficient code generation
- Configuration management natively provided in the language
- Model reuse via inheritance
- Standard library provides interfaces and basic components in multiple physical domains
- Active development community with a growing list of free and commercial application libraries

Though the focus of this work is on the model architecture and not the modeling language, key enabling features of the Modelica language will be briefly highlighted in the context of the model architecture and its capabilities.

VEHICLE MODEL ARCHITECTURE

Vehicle models can be used throughout the systems engineering V to support a range of model-based engineering efforts from concept assessment and design through testing and verification. Architecture-based modeling is a key element of rapid virtual prototyping of vehicles to support model-based systems development. A formal model architecture provides consistent interfaces and system decomposition to promote distributed model development and plug-n-play interoperability between models and application libraries.

Figure 1 shows the top-level view of the vehicle model architecture in Modelica. This architecture has been developed to support flexible, configurable modeling of both conventional and hybrid vehicles. The architecture is based on the original Vehicle Model Architecture (VMA) [5] and the subsequent improvements implemented in the VehicleInterfaces library [6]. The architecture includes elements for the external environment, driver, control system, and all main subsystems of the vehicle. Each element contains some subset of controller and multi-domain physical connectors which are connected appropriately in the architecture. Note that the architecture simply defines interfaces between components, not the implementation details. The architecture is designed to support plug-n-play modeling of conventional and hybrid ground vehicles at the system, subsystem, and even the component levels. While the architecture as shown is very modular and flexible and can comprise a wide variety of vehicles and model-based engineering applications, it is trivial to extend from it to add additional top-level systems or even reconfigure for vehicles with different topologies such as multi-axle vehicles while maximizing model reuse. Each top-level subsystem will be discussed in detail to provide context for the types of models and implementations that are possible within the vehicle model architecture to support a range of model-based engineering activities.

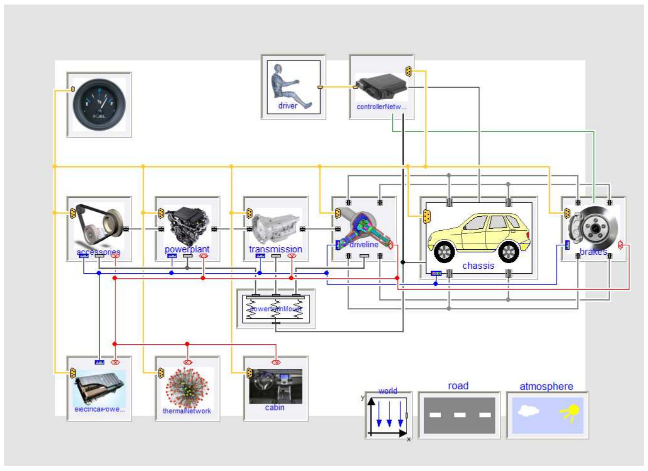

Figure 1: Vehicle model architecture

The vehicle model architecture shown in Figure 1 includes both multi-domain physical and control system connections between component connectors. Mechanical connections between component connectors are shown in grey, electrical connections in blue, thermal connections in red, and controller connections in yellow. Note that the mechanical connections between components can be either 3D multi-body connections or simple 1D rotational connections depending on the component implementation. While Modelica has been used extensively for detailed vehicle dynamics modeling [7]-[8], this paper focuses on powertrain torsional dynamics with 1D rotational connections.

The architecture utilizes expandable connectors [4] in Modelica to provide additional flexibility. Expandable connectors can be used to construct buses of signals or even other connectors, including physical connectors. Unlike regular Modelica connectors whose contents are defined *a priori*, expandable connectors are constructed dynamically based on the union of all the connections supplied in the various components. Expandable connectors are used in the vehicle model architecture to create the following elements:

- Flexible, hierarchical control bus
- Multi-voltage electrical bus for electrical interactions
- Multi-node thermal bus for thermal interactions

Driver

The driver model interfaces with the control systems and provides a range of typical driver inputs to the system such as key on, accelerator demand, braking demand, gear selection (for manual transmissions), steering command, *etc.* corresponding to a particular mission profile. Various driver implementations are available to provide different ways of driving the vehicle model. A direct driver model implementation actuates the model in a forward-facing way based on accelerator and brake inputs with resulting vehicle speed response. This driver implementation is used for performance simulations such as accel tests. Driver models have also been implemented to allow the vehicle to follow a prescribed drive cycle. By taking advantage of the acausal nature of the Modelica formulation for the underlying models, *it is possible to run both backward and forward models* by simply changing the driver model under the assumption that the underlying physical model is invertible. The driver implementations contain a list of available drive cycles based on available literature which are accessed via a drop-down list from the model architecture. Additional drive cycles are easily added either from a file containing the speed trace or by entering the data directly into the model.

In addition to the commands to drive the vehicle, the driver model can also be used to command other elements of the mission profile such as auxiliary loads, climate control functions, and other mechanical or electrical power demands modeled via its interface to the control system. Sample duty cycles for military mission profiles can be found in [9].

Environment

The external environmental conditions are represented by the road and atmosphere components. The road model provides dynamic information regarding grade, surface conditions, *etc.* The atmosphere component provides dynamic conditions such as ambient temperature and pressure, humidity, *etc.*

Controller Network

The control system is modeled as a distributed controller network. The controller element interfaces with the driver and with the rest of the system via an expandable, hierarchical control bus. The controller network is configurable and allows for additional controllers to be added along with additional elements to the physical system. The following interfaces are included as part of the base architecture as shown in Figure 2:

- Vehicle system controller
- Engine controller
- Transmission controller
- Driveline controller
- Brakes controller
- Battery controller
- Climate controller

Interfaces are also included for the following controllers which are added to the controller network as needed:

- Electric motor controller
- Electric generator controller

The controller implementations are based on individual control features such as drive-by-wire vehicle system control for power/energy/thermal management, transmission shift strategy, battery state of charge and temperature, cabin temperature control, *etc*. The interfaces are flexible enough to support controller implementations of varying level of detail as required by the simulation. These control models can be implemented natively in Modelica or integrated into the Modelica tool from external sources such as C code. Another very common scenario is for the controller model to interface with Simulink to support closed loop control between Simulink models of the controller and Modelica models of the vehicle [10].

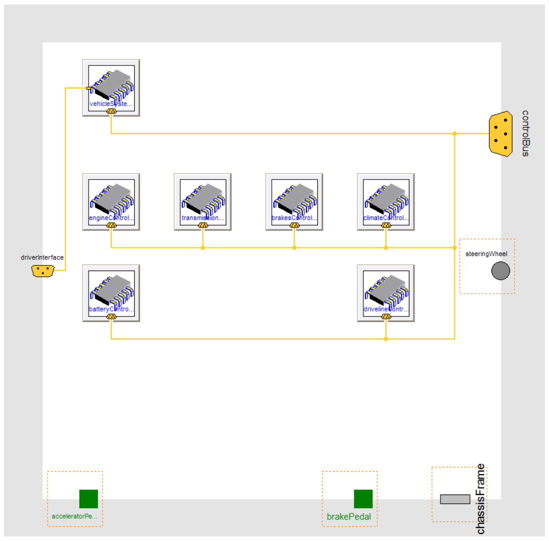

Figure 2: Distributed controller network

Accessories

The accessories component includes a mechanical connection to the powerplant and interacts with the electrical system and thermal system via expandable connectors. Implementations for this component range from simple prescribed loads on the crankshaft to detailed models of mechanical and electrical system loads including auxiliary components critical to the mission profile such as the climate control systems, weapon and targeting systems, and auxiliary power units. Figure 3 shows a sample accessories implementation with an electric AC system. Note the multi-domain nature of this component with the electric motor connected to the electrical bus, the AC model connected to the thermal bus, and the fixed power accessories connected to the powerplant flange via the rotational connector.

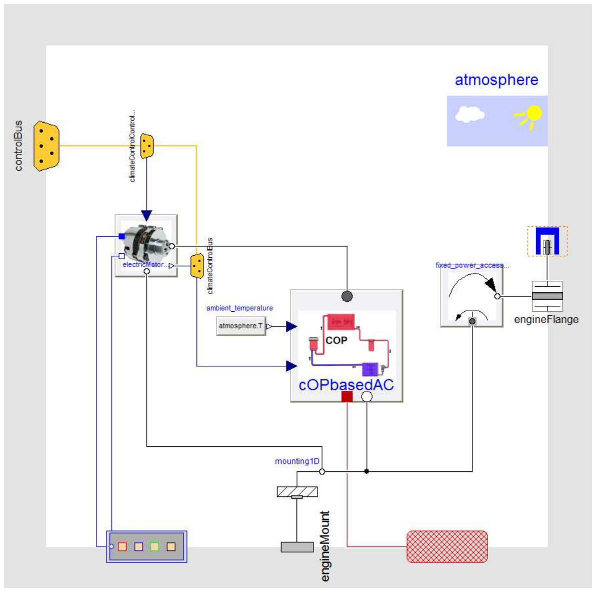

Figure 3: Accessory implementation with electric air conditioning system

Powerplant

The powerplant subsystem interfaces with the control system, electrical and thermal systems, and includes mechanical flange connections to the accessory subsystem and transmission subsystem. Powerplant implementations range from conventional engine models of varying level of detail to pure electric models. The architecture can support both cycle-resolved mapped engine models for long time scale simulations as shown in Figure 4 and detailed, crank-angle resolved models including component-based models of the intake, exhaust, cylinder, and combustion dynamics.

While mapped engine implementations are typically appropriate for drive cycle simulations, detailed engine model implementations may be required to support engine start-stop simulations and driveline NVH applications where simulation of the detailed thernodynamics and transient engine dynamics are required. Modelica has been used extensively for detailed engine modeling [11]-[14].

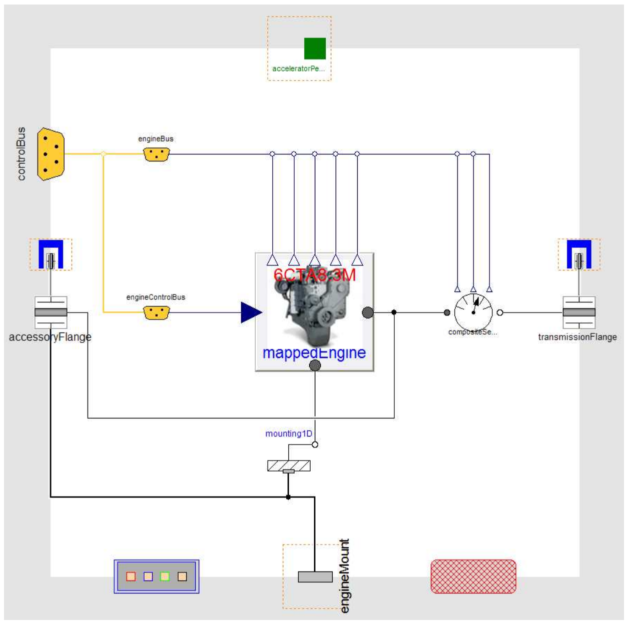

Figure 4: Powerplant implementation for a mapped engine

Transmission

The transmission subsystem includes connections to the control system, electrical system, and mechanical connections to the powerplant and the driveline subsystems. The transmission subsystem implementation can be used to represent a wide range of conventional and hybrid architectures. Transmission models of varying level of detail are supported within the architecture. The simplest implementation is a standard transmission with prescribed engagements at fixed gear ratios. More detailed conventional models include continuously-variable transmissions (CVTs), dynamic engagements with clutches, synchronizers, hydraulic actuation, dampers, and multiple shaft configurations.

Hybrid propulsion and transmission systems can also be implemented within the transmission interface. These implementations can include multiple motors and generators, associated gearing, hydraulic systems, clutches, and isolation elements. The use of various hybrid implementations will be illustrated in subsequent sections.

Driveline

The driveline subsystem includes connections to the control system, electrical system, and mechanical connections to the transmission subsystem and wheel-based connections to the chassis. Various driveline implementations are possible within the interface including front wheel drive, rear wheel drive, and all-wheel drive. Figure 5 shows a simple, rigid AWD implementation with a transfer case model. As with the other subsystems, varying levels of detail are supported within the interface. For example, distributed inertias and driveline compliance can be used to support driveline NVH analyses.

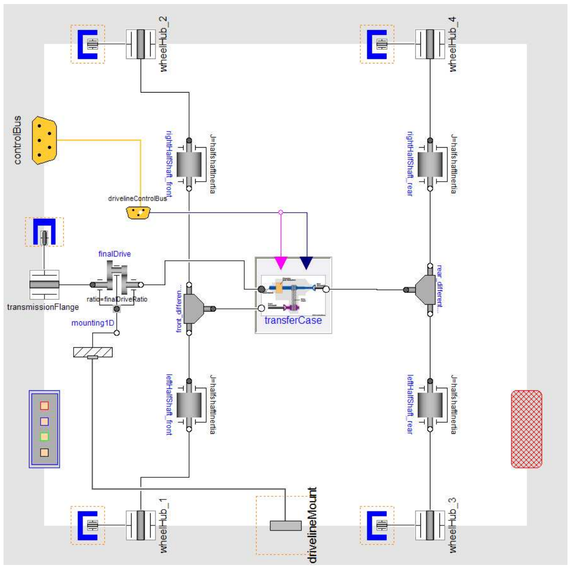

Figure 5: Driveline implementation for an AWD system

Chassis

The chassis subsystem includes connections to the control system, electrical system, and wheel-based mechanical connections to the driveline and brakes subsystems. A simple chassis implementation is shown in Figure 6. This implementation includes a no-slip tire model, lumped vehicle inertia, and configurable models for aerodynamic and rolling resistance loads. Since this model does not consider each tire individually, it is not appropriate for drivability simulations. Available chassis implementations with additional detail include individual tires with slip, distributed vehicle inertias, and dynamic suspension components.

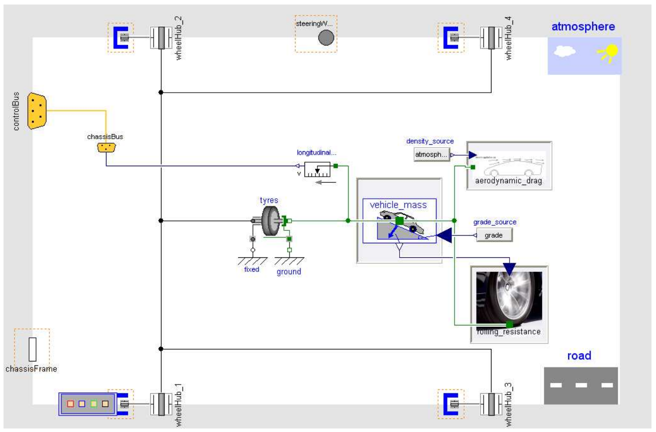

Figure 6: Simple chassis implementation with vehicle loads

Brakes

The brakes subsystem interfaces with the control system, electrical system, thermal system, and the chassis via wheel-based mechanical connectors. Available implementations for the brakes subsystem range from simple prescribed torque brakes to detailed models of the ABS system including pump and hydraulic actuation dynamics.

Powertrain Mounts

The impact of the various driveline systems on the powertrain mounts is often easily overlooked. Thus, the subsystem interface for the powertrain mounts is included in the model architecture to encourage consideration of the impact of the drivetrain on the mounting system and to support driveline NVH analyses. The implemented mechanical subsystems include connections to the mounts as shown in Figure 1. In the simplest implementation, the mounts can be considered a rigid grounded element. More detailed mount implementations can include detailed 3D multibody dynamics including hydraulics considerations.

Electrical Power Network

The electrical power network subsystem interfaces with the control system, electrical system, and thermal bus. Via the expandable electrical bus, the electrical power network can support multi-voltage electrical buses. Implementations of varying levels of details ranging from simple fixed voltages to detailed models of batteries and power converters are easily accommodated. Figure 7 shows a simple dual voltage system with fixed capacity batteries with thermal effects.

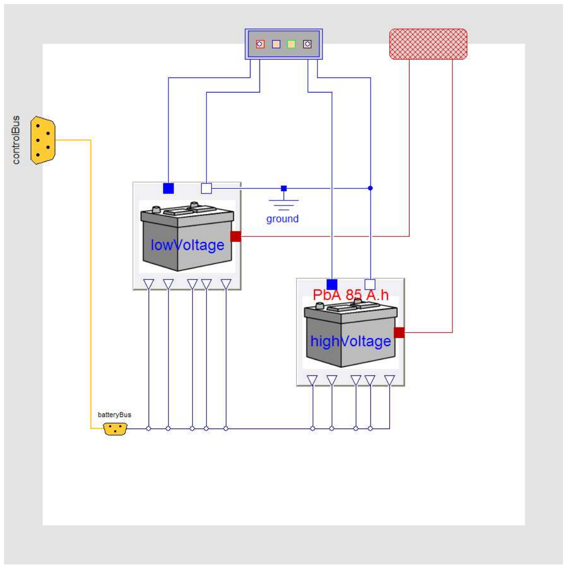

Figure 7: Simple, fixed capacity dual voltage electrical power network implementation

Thermal Network

The thermal network subsystem models the various thermal pathways between the thermal nodes in the expandable thermal bus. The inclusion of this component enables additional modularity and flexibility in the modeling of the vehicle thermal system. Furthermore, this component can be used to isolate subsystems from changes in the level of detail in the thermal architecture from neighboring subsystems. For example, the thermal network is used to route cooling from the air-conditioning system to multiple cabin zones without requiring a change to the lumped air-conditioning model used for a single zone cabin system.

Cabin

The cabin subsystem is included to support thermal modeling of the vehicle cabin. Cabin implementations of varying level of detail, including multiple zones, are available to support climate control system design [10] and control and occupant thermal comfort analyses.

SAMPLE VEHICLE IMPLEMENTATIONS

To illustrate the usage of the vehicle model architecture described in the previous section, sample implementations for a military ground vehicle are provided. In an attempt to highlight the modularity and flexibility of the architecture, a baseline conventional vehicle model is created and then configured in several different hybrid implementations while reusing elements from the baseline implementation. Sample results from different types of engineering simulations for the various vehicles are shown in the following section.

The base vehicle chosen solely by Emmeskay for the modeling study is the M1117 Armored Security Vehicle (ASV) from Textron Marine and Land Systems. Note that this study is entirely based on vehicle data available in the open literature for the ASV and thus may not be representative of the actual vehicle performance due to the large amount of vehicle parameterization which was not available publicly. Furthermore, the hybrid implementations are purely illustrative for the purposes of this paper and should not to be construed as potential vehicle applications.

Conventional ASV

The baseline conventional ASV vehicle model implementation is shown in Figure 8. This implementation is based on the ASV data sheet [15] and includes representations for the following components to support drive cycle and performance analyses:

- Mapped engine for Cummins 6CTA8.3 260hp diesel engine as shown in Figure 4
- Allison MD3560, 6 speed transmission model with prescribed engagements
- Simple chassis with conventional loads at 13,408 kg (29,560 lb) with rigid 14.00 R20XZL tires as shown in Figure 6
- Simplified driveline with all-time four wheel drive based on model shown in Figure 5
- Prescribed torque brakes
- Electrical power network with fixed voltage battery
- Rigid, grounded powertrain mounts
- Controller network with simple transmission controller with speed-based gear shift strategy
- Null implementations for battery, thermal network, and cabin

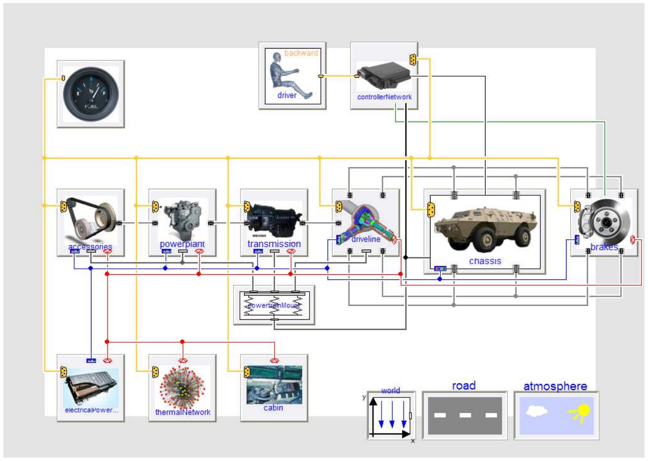

Figure 8: Conventional vehicle implementation

Series Hybrid ASV

Based on the conventional vehicle model, the sample series hybrid model shown in Figure 9 was implemented by reusing the base model with the following changes:

- Dual voltage electrical power network with 85A.h lead-acid battery
- Series hybrid transmission with generator and single motor
- Controller network with series hybrid vehicle system control based on simple battery state-of-charge (SOC) control including regenerative braking

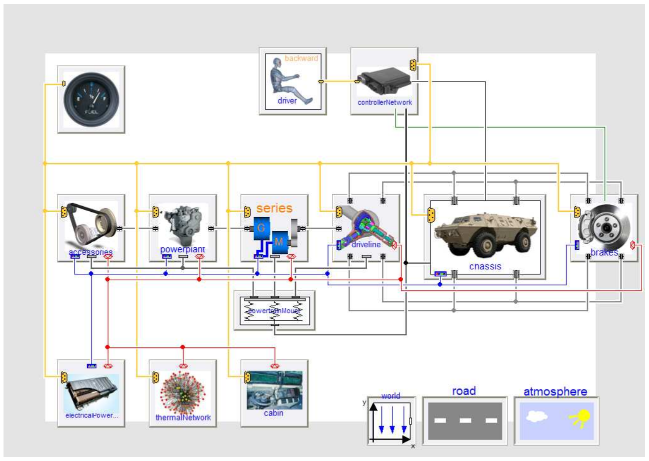

Figure 9: Series hybrid vehicle implementation

The vehicle model architecture relies on the formal configuration management features of the Modelica language to create model variants. Using inheritance, it is possible to extend from an existing model to maximize model re-use and apply compatible changes based on formal interface requirements. Figure 10 shows the graphical configuration of the series hybrid model by selecting the series transmission model from the tool-created list of compatible models. This sort of configuration is available at the top-level of the architecture for all subsystems and also at the component and primitive level in the various systems. Thus, new vehicle implementations can be created in a plug-n-play fashion with substantial model re-use at the system, subsystem, and component levels.

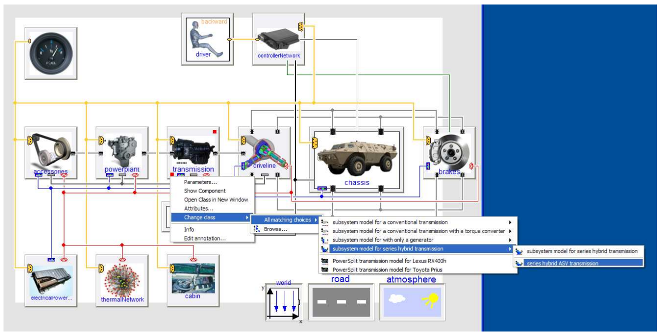
backward
driver
controllerNetwork
accessories
powerplant
transmission
driveline
chassis
brakes
Parameters...
Show Component
Open Class in New Window
Attributes...
Change class
Info
Edit annotation...
All matching choices
Browse...
subsystem model for a conventional transmission
subsystem model for a conventional transmission with a torque converter
subsystem model for with only a generator
subsystem model for series hybrid transmission
PowerSplit transmission model for Lexus RX400h
PowerSplit transmission model for Toyota Prius
subsystem model for series hybrid transmission
series hybrid ASV transmission
electricalPower...
thermalNetwork
cabin
world
road
atmosphere

Figure 10: Model configuration for series hybrid vehicle

Series Hybrid ASV with Wheel Motors

A variant of the series hybrid ASV shown in Figure 9 can be created with wheel motors. The wheel motor variant shown in Figure 11 extends from the series hybrid implementation with the following changes to the powertrain model:

- Series hybrid transmission implementation with generator only
- Driveline implementation with motors at each wheel as shown in Figure 12

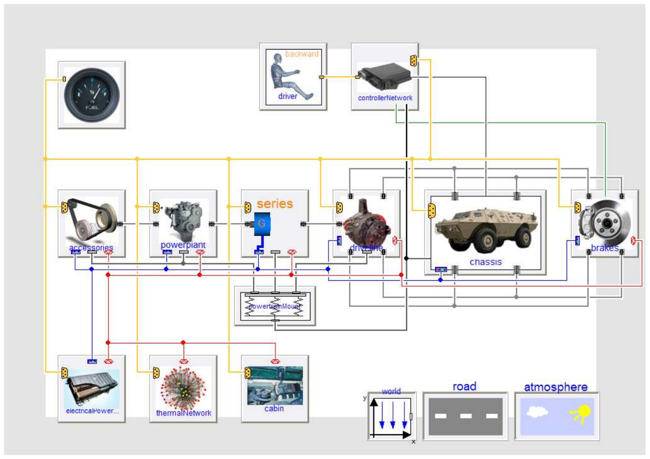

Figure 11: Series hybrid vehicle with wheel motors implementation

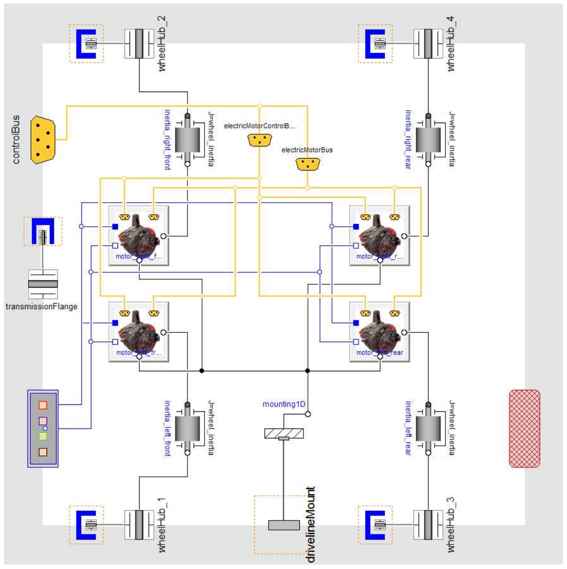

Figure 12: Driveline implementation with wheel motors

Parallel Hybrid ASV with PowerSplit

The vehicle model architecture accommodates parallel hybrid implementations as well. The parallel hybrid variant of the ASV with a PowerSplit configuration shown in Figure 13 is created from the baseline conventional vehicle model with the following changes:

- Reuse of the dual voltage electrical power network model from the series hybrid model
- PowerSplit hybrid transmission with two motor/generators

- Controller network with PowerSplit hybrid vehicle system control based on simple battery SOC control including regenerative braking

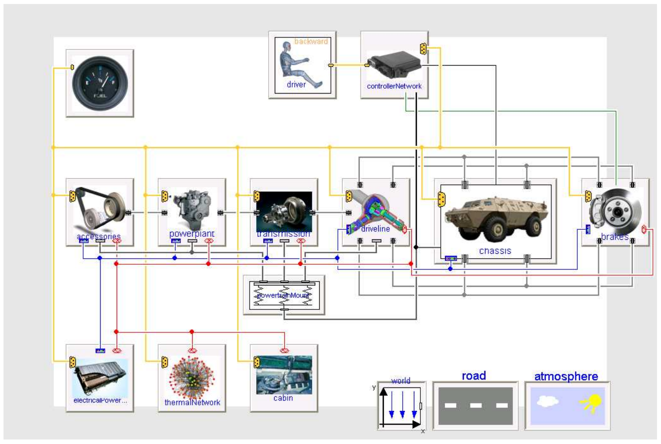

Figure 13: Parallel hybrid PowerSplit implementation

The PowerSplit hybrid vehicle system controller shown in Figure 14 highlights another key feature of the model architecture in Modelica, namely the ability to use model inversion to avoid the need to develop control strategies. Since the PowerSplit configuration introduces an additional degree of freedom for the engine, the engine speed can be controlled independent of the vehicle. Rather than implement a speed control strategy for the engine, model inversion is used in Figure 14 to directly calculate the generator torque required such that the resulting engine speed is equal to the desired engine speed. The ability to perform model inversion is a direct result of the acausal formulation in Modelica.

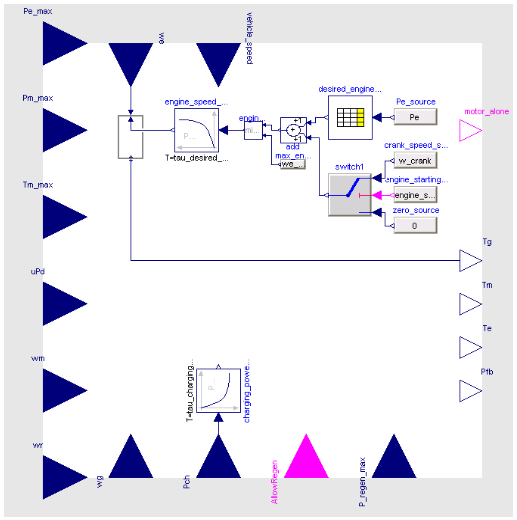

Figure 14: PowerSplit hybrid vehicle control features with perfect control for generator

Parallel Hybrid ASV with E4WD

The parallel hybrid variant with E4WD shown in Figure 15 is created based on the conventional vehicle with the following changes:

- Re-use of the dual voltage electrical power network model from the series hybrid model
- E4WD hybrid driveline with single motor driving one of the axles and the other driven conventionally via the engine-transmission as shown in Figure 16
- Controller network with E4WD hybrid vehicle system control based on simple battery SOC control including regenerative braking

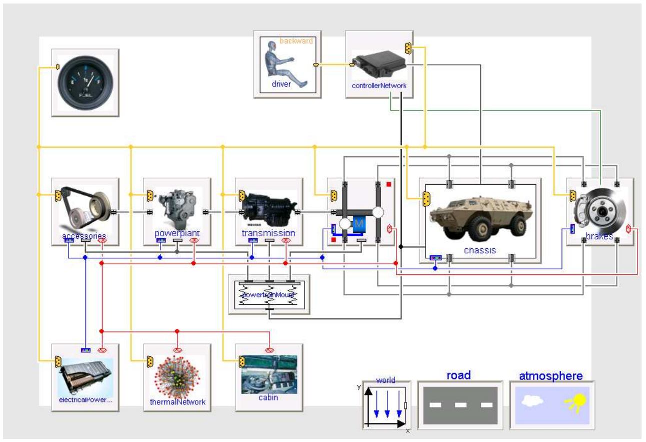

Figure 15: Parallel hybrid vehicle with E4WD implementation

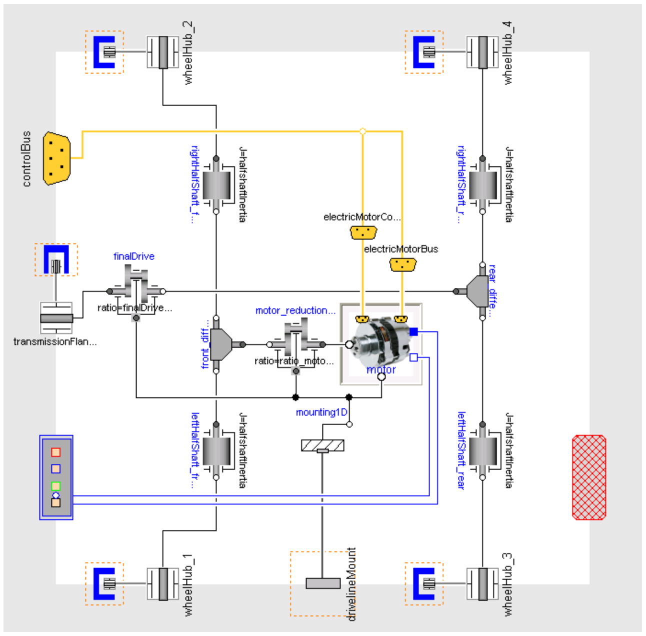

Figure 16: E4WD driveline implementation

SAMPLE RESULTS

This section includes sample results from simulations of selected implementations discussed in the previous section. Note that there is neither complete vehicle parameterization nor validation data associated with any of these implementations so the results should be viewed as purely illustrative.

Figure 17 shows a performance run for the conventional ASV model from Figure 8. The driver model in this simulation simply commands max torque from the engine. The vehicle performance as measured by the time for 0-32 km/hr is quoted as less than 7 sec [15]. Given the uncertainty in the vehicle parameterization and gear shift strategy, the model yields quite reasonable performance predictions.

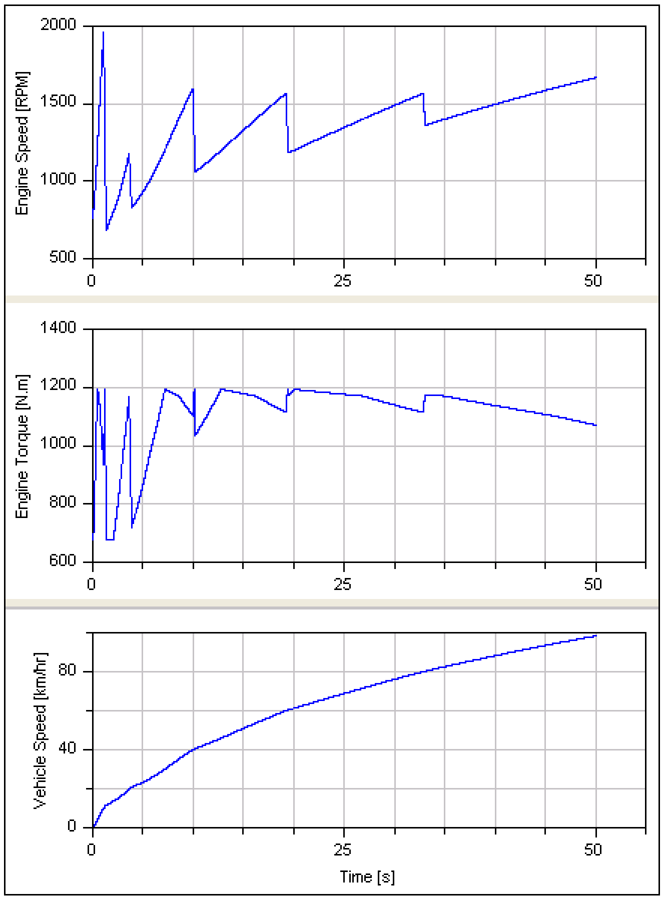

Figure 17: Performance run, conventional ASV

Figure 18 shows some sample results from simulation of the conventional ASV model over a drive cycle. The drive cycle simulated is the West Virginia University Suburban (WVUSUB) drive cycle [16]. The vehicle speed trace is shown in the top figure, followed by the engine torque, and the fuel consumption. The backward model was simulated thus requiring that the vehicle meet the commanded speed trace without regard for max torque constraints. Violation of the max torque constraints can be an indication that the drive cycle is too aggressive for the vehicle being modeled or that the modeled control strategy needs additional refinement.

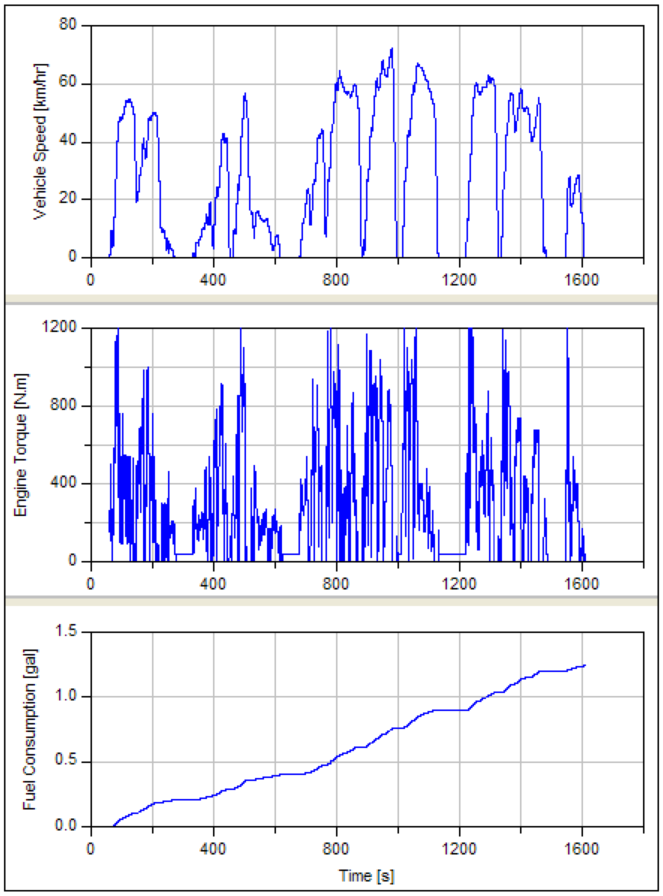

Figure 18: Conventional ASV, WVUSUB drive cycle

By simply selecting the forward driver, the exact same physical model can be run in forward-facing mode. Figure 19 shows the normalized power demand for the conventional ASV over the same WVUSUB drive cycle from the backward and forward model. Though there are very slight deviations, the backward and forward models yield nearly identical power demand traces and drive cycle fuel consumption (1.24043 and 1.24086 gallons for the backward and forward models, respectively). While there can be advantages to running either a backward or forward model, the acausal nature of the underlying Modelica model formulation of the vehicle architecture allows the unique ability to run in either mode with only trivial changes to the driver model (assuming the physical model is invertible).

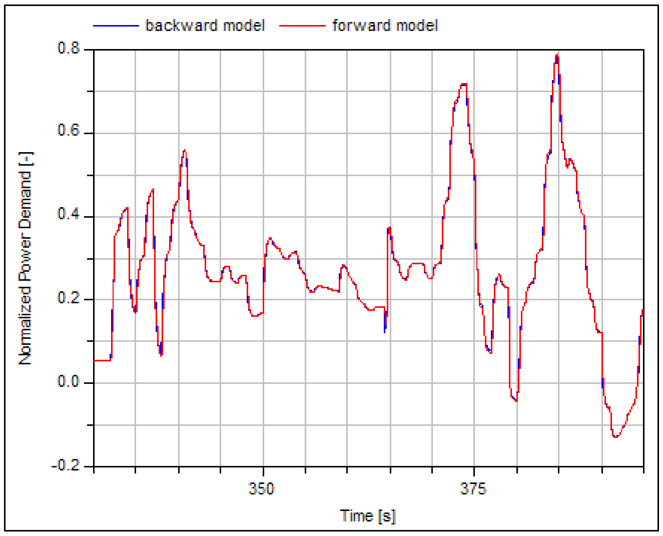

Figure 19: Normalized power demand comparison between backward and forward model, conventional ASV

Figure 20 shows the results of running the series ASV model implementation shown in Figure 9 over the WVUSUB drive cycle. The engine and motor speeds are shown in the top plot, followed by plots of the state of charge (SOC) in the high voltage battery and the fuel consumption. The battery charging characteristics, such as the min and max SOC limits and the charging rate, are controlled by the implementation of the battery controller. The engine operation during charging is controlled via the vehicle system controller implementation. Note that no attempt was made to balance the battery SOC at the start and end of the simulation though they are reasonably close in this particular simulation. Thus, care should be exercised when comparing fuel consumption results between vehicle implementations.

As expected in series hybrid operation, the motor provides the tractive power for the vehicle, and the engine operates as needed to charge the battery. For the WVUSUB cycle simulated, there are roughly two complete charge and discharge cycles. Regenerative braking during vehicle deceleration provides additional battery charging throughout the cycle. The cycle fuel consumption stays flat when there is no charging since the engine is shut down.

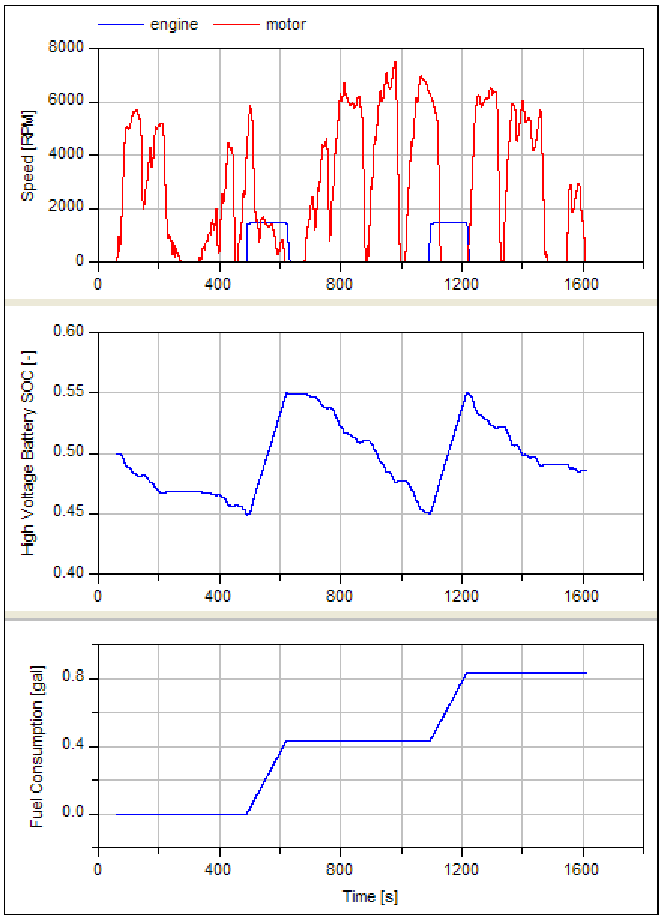

Figure 20: Series ASV, WVUSUB drive cycle

As is well understood, fuel efficiency improvements in hybrid vehicles are a result of both the hybrid architecture and physical system components and the vehicle control strategy which manages the vehicle power demands. Thus, in addition to good representations of the efficiency and losses in the physical system, it is just as important to model a sufficiently detailed control strategy that captures the intrinsic capabilities in the modeled hybrid implementation. For this reason, the controller network in the vehicle model architecture supports distributed control strategies of user-specified level of detail. Figure 21 shows the impact of the vehicle charging strategy on the resulting fuel consumption on the WVUSUB drive cycle. The top plot shows the effect of increasing the regenerative braking capability of the vehicle. As more braking energy is captured, the resulting fuel consumption decreases as expected. The bottom plot shows the effect of the commanded charging rate on the fuel consumption.

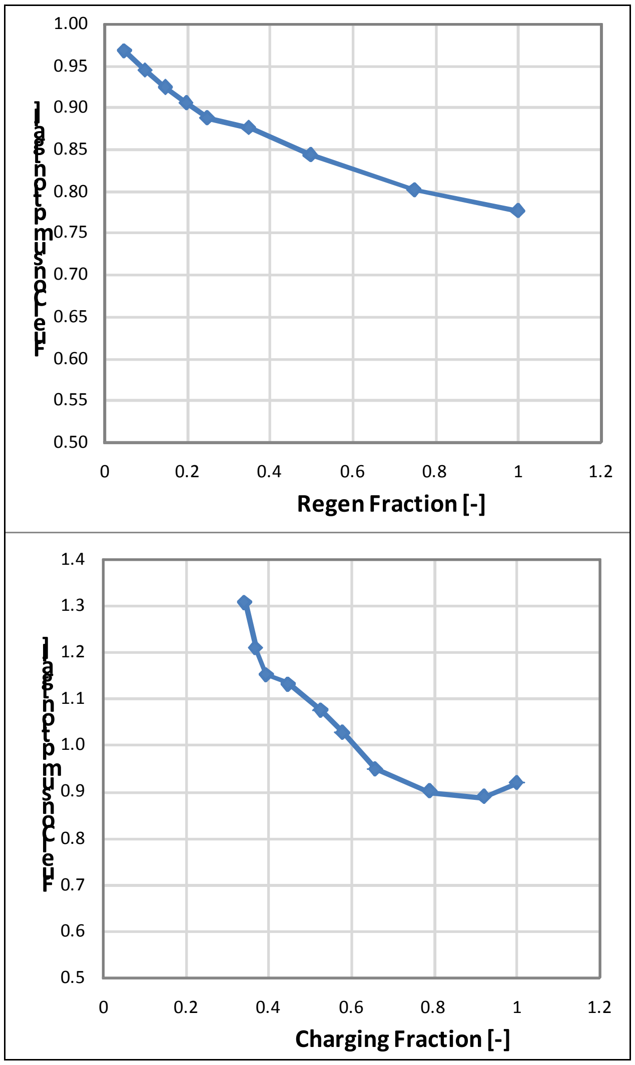

Figure 21: Impact of charging strategy on series ASV fuel consumption, WVUSUB drive cycle

Figure 22 shows the results of running the parallel PowerSplit ASV model shown in Figure 13 on the WVUSUB drive cycle. The simulation results are from a backward model. The top plot shows the speeds of the engine, motor, and generator followed by plots of the high voltage battery SOC and the fuel consumption. Again, note that no attempt was made to balance the battery SOC at the start and end of the simulation. For this simulation, the ending SOC exceeds that at the beginning of the simulation.

As in the series hybrid vehicle, there are roughly two charge and discharge cycles during the WVUSUB drive cycle. The resulting device speeds and the battery charging are controlled via the battery charging feature and the vehicle system controller which manages power demand.

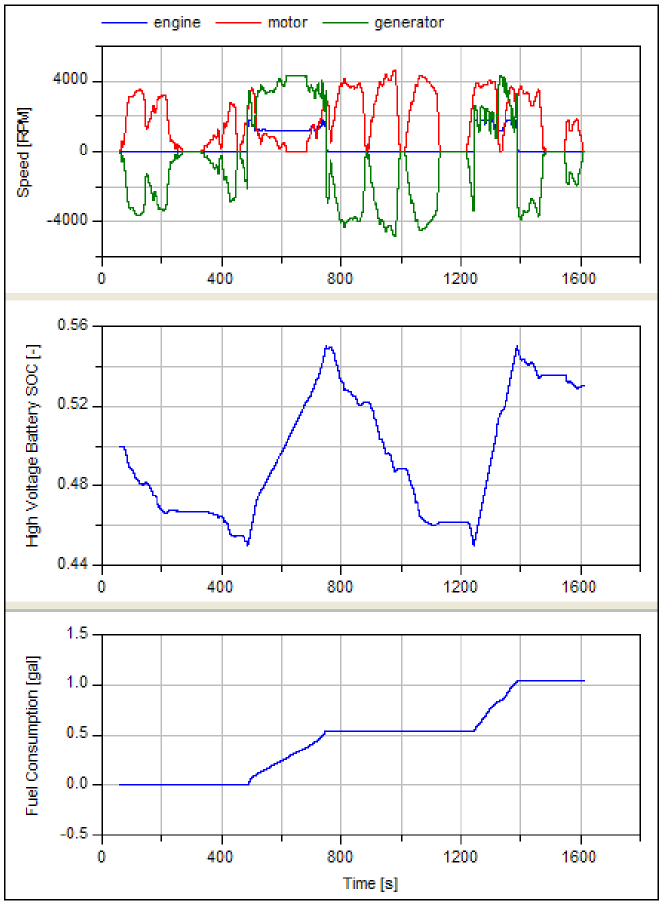

Figure 22: Parallel PowerSplit ASV, WVUSUB drive cycle

EXTENDED ANALYSES

The results shown in the previous section focused on performance and fuel economy simulations over various drive cycles. In these sorts of simulations, it is often appropriate to implement simplified models with rigid shafts, simplified tire dynamics, *etc.* However, the model architecture can support subsystem models of varying fidelity and thus enables a range of analyses for model-based systems development throughout the product development process, including controls development. Furthermore, component, and subsystem model libraries can be shared across applications to support reuse of validated models. Some sample analyses are discussed in this section.

Drivability

Drivability is another key vehicle performance and safety metric for military ground vehicles. Within the vehicle model architecture, selective refinement of the transmission, driveline, and chassis models enables a whole class of drivability analyses. For example, Figure 23 shows a slightly more dynamic gearbox model which includes clutches and hydraulic interfaces to support detailed transmission shift dynamics including hydraulic effects. Some sample supported applications include the following:

- Transmission shift dynamics including hydraulic actuation dynamics
- Vehicle performance with detailed road-tire interactions over varying surfaces and real-world military duty cycles
- Traction control and ABS system impacts on vehicle performance over military duty cycles

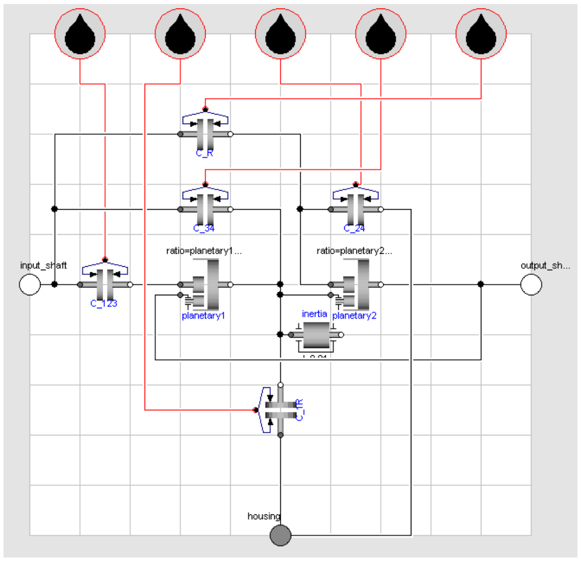

Figure 23: Four speed gearbox with clutches and hydraulic interfaces

NVH

Noise, vibration, and harshness (NVH) is a key consideration in the design and optimization of powertrains in military ground vehicles. In particular, NVH issues can be particularly challenging in hybrid vehicles with multiple sources of tractive power and resulting power paths through the drivetrain. The vehicle model architecture can support upfront analysis of vehicle level NVH issues with selective refinement of key powertrain components. By introducing compliant shafts into the system and a tire slip model, a wide variety of NVH simulations are possible within the vehicle model architecture [17]. Sample analyses include:

- Transmission damper design, optimization, and performance with detailed engine excitation
- Engine start-stop NVH
- Dynamic driving and braking simulations that excite unusual driveline NVH modes

A sample simulation result from a braking event in an electric vehicle modeled within the vehicle model architecture is shown in Figure 24.

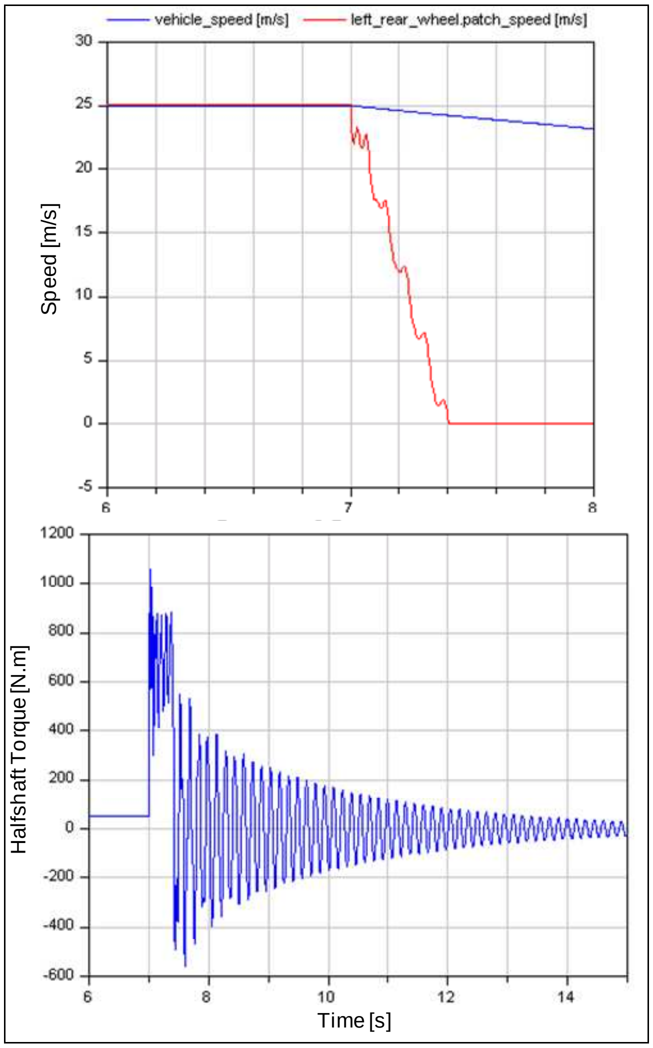

Figure 24: Dynamic braking simulation in an electric vehicle with associated NVH response

Energy/Thermal Management

Energy and thermal management analyses are supported within the model architecture. The expandable thermal bus allows thermal modeling of varying levels of detail in all the vehicle subsystems. The model architecture can support the thermofluid modeling of coolant and refrigerant loops with component-based dynamic heat rejection models, climate control system models of varying levels of detail, battery and electrical system cooling requirements, and thermal comfort predictions over real world military duty cycles.

Another advantage of the connector-based Modelica formulation is the ability to perform detailed energy accounting and model conservation checking at all levels of the model hierarchy [18]. Using the multi-domain model physical connectors, it is possible to perform energy balance checks directly from the Modelica source code to ensure that the model is conservative. Furthermore, it is possible to perform detailed energy accounting at all levels of the model hierarchy to better understand efficiency losses throughout the system. Figure 25 shows an example of how these losses can be effectively visualized by losses among sibling subsystems or as a function of time. Furthermore, it is possible to select subsets of such data sets to analyze losses, *e.g.* in a particular vehicle speed range, transmission gear, or during regenerative braking.

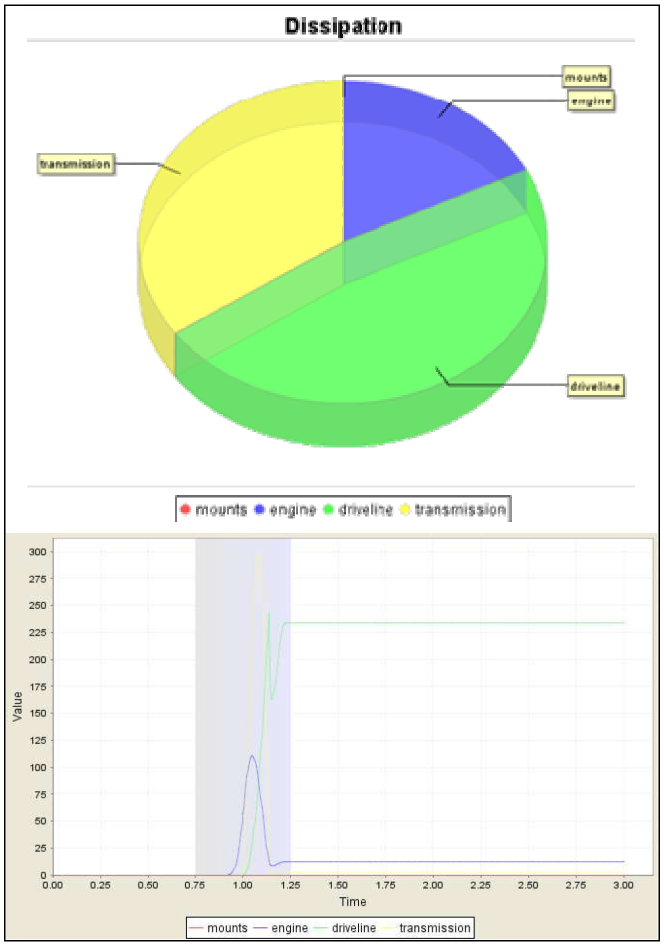

Figure 25: Energy accounting based on Modelica models (from presentation of [18])

Model-Embedded Control

The vehicle model architecture can also support distributed controller development at both the vehicle system and component controller levels. An advanced usage of Modelica physical models is to directly synthesize a controller based on control objectives and the plant model [19]. The vehicle model architecture can support model embedded control approaches utilizing either instantaneous optimization or dynamic programming strategies.

CONCLUSIONS

Model-based systems development for military ground vehicles is a key enabler for rapid prototyping and design optimization. This paper describes a flexible, modular vehicle model architecture in Modelica to support a range of systems engineering analyses of conventional and hybrid vehicle architectures throughout the product development process. The architecture supports models of varying levels of details at multiple levels in the model hierarchy so that the same model architecture may support engineering activities over the entire systems engineering V with substantial model reuse and selective, localized model refinement. Plug-n-play capability enabled by formal Modelica language constructs at the system, subsystem, and component level allows rapid model configuration of different vehicle implementations.

Sample implementations illustrate the usage of the vehicle model architecture and compatible multi-domain component model libraries to create model implementations of a military vehicle in both conventional and hybrid architectures. The models and sample simulation results highlight the flexibility of the model architecture and the wide range of engineering analyses that can be supported.

REFERENCES

[1] Wipke, K.B., Cuddy, M.R., and Burch, S.D., “ADVISOR 2.1: A User-Friendly Advanced Powertrain Simulation Using a Combined Backward/Forward Approach,” IEEE Transactions on Vehicular Technology, vol. 48, pp. 1751-1761, 1999.
[2] Rousseau, A., and Pasquier, M., "Validation of a Hybrid Modeling Software (PSAT) Using Its Extension for Prototyping (PSAT-PRO)," Global Powertrain Congress, Detroit, 2001.
[3] Nedungadi, A., Pozolo, M., and Mimnaugh, M., “A General Purpose Vehicle Powertrain Modeling and Simulation Software-VPSET”, WAC 2008 Conference, 2008.
[4] Modelica Association, “Modelica Language Specification, Version 3.0”, http://www.modelica.org/documents/ModelicaSpec30.pdf
[5] Tiller, M., Bowles, P., and Dempsey, M., “Development of a Vehicle Modeling Architecture in Modelica”,

Proceedings of 3rd International Modelica Conference, Modelica Association, pp. 75-86, 2003.

[6] Dempsey, M., Gafvert, M., Harman, P., Kral, C., Otter, M., and Treffinger, P., "Coordinated Automotive Libraries for Vehicle System Models", Proceedings of 5th International Modelica Conference, Modelica Association, pp. 33-41, 2006.

[7] Andreasson, J. and Gäfvert, M., "The VehicleDynamics Library - Overview and Applications", Proceedings of the 5th Modelica Conference, Vienna, Austria, Modelica Association, 2006.

[8] Philipson, N., Andreasson, J., Gäfvert, M., Woodruff, A., "Heavy Vehicles Modeling with the Vehicle Dynamics Library", Proceedings of 6th International Modelica Conference, Bielefeld, Germany, Modelica Association, p. 629-634, 2008.

[9] Brudnak, M., Pozolo, M., Meldrum, A., Mortsfield, T., Shvartsman, A., Smith, W., Goodell, J., and Holtz, D., "Virtual Combat Vehicle Experimentation for Duty Cycle Measurement", SAE 2008-01-0776, 2008.

[10] Pitchaikani, A., Jebakumar, K., Venkataraman, S., Sundaresan, S.A., "Real-Time Drive Cycle Simulation of Automotive Climate Control System", to be published in the Proceedings of the 7th International Modelica Conference, Como, Italy, Modelica Association, 2009.

[11] Newman, C., Batteh, J., and Tiller, M., "Spark-Ignited-Engine Cycle Simulation in Modelica", 2nd International Modelica Conference Proceedings, pp. 133-142, Modelica Association, 2002.

[12] Batteh, J., Tiller, M., and Newman, C., "Simulation of Engine Systems in Modelica", 3rd International Modelica Conference Proceedings, pp. 139-148, Modelica Association, 2003.

[13] Bowles, P. and Batteh, J., "A Transient Multi-Cylinder Engine Model Using Modelica", SAE 2003-01-3127, SAE Powertrain and Fluid Systems Conference, Pittsburgh, PA, 2003.

[14] Batteh, J. and Newman, C., "Detailed Simulation of Turbocharged Engines with Modelica", Proceedings of 6th International Modelica Conference, Bielefeld, Germany, Modelica Association, p. 69-75, 2008.

[15] Textron Marine and Land Systems, "M1117 Armored Security Vehicle", http://www.textronmarineandland.com/pdfs/datasheets/asv_datasheet.pdf, 2008.

[16] Clark, N., Daley, J., Nine, R., and Atkinson, C., "Application of the New City-Suburban Heavy Vehicle Route (CSHVR) to Truck Emissions Characterization", SAE 1999-01-1467, International Fuels and Lubricants Meeting, Dearborn, 1999.

[17] Pitchaikani, A., Koppu, K., Venkataraman, S., Tiller, M., and Batteh, J., "Powertrain Torsional Vibration System Model Development in Modelica for NVH Studies", to be published in the Proceedings of the 7th International Modelica Conference, Como, Italy, Modelica Association, 2009.

[18] Tiller, M., "Verification and Validation of Physical Plant Models", SAE 2009-01-0527, SAE Congress, Detroit, 2009.

[19] Tate, E., Sasena, M., Gohl, J., and Tiller, M., "Model Embedded Control: A Method to Rapidly Synthesize Controllers in a Modeling Environment", Proceedings of 6th International Modelica Conference, Bielefeld, Germany, Modelica Association, p. 493-502, 2008.