



MBSE

The Rise of the Machines?

Electric Vehicle Technologies Workshop;
Session EV-03: Model Based Drivetrain Design

Aymeric Rousseau
rousseau@anl.gov
www.autonomie.net

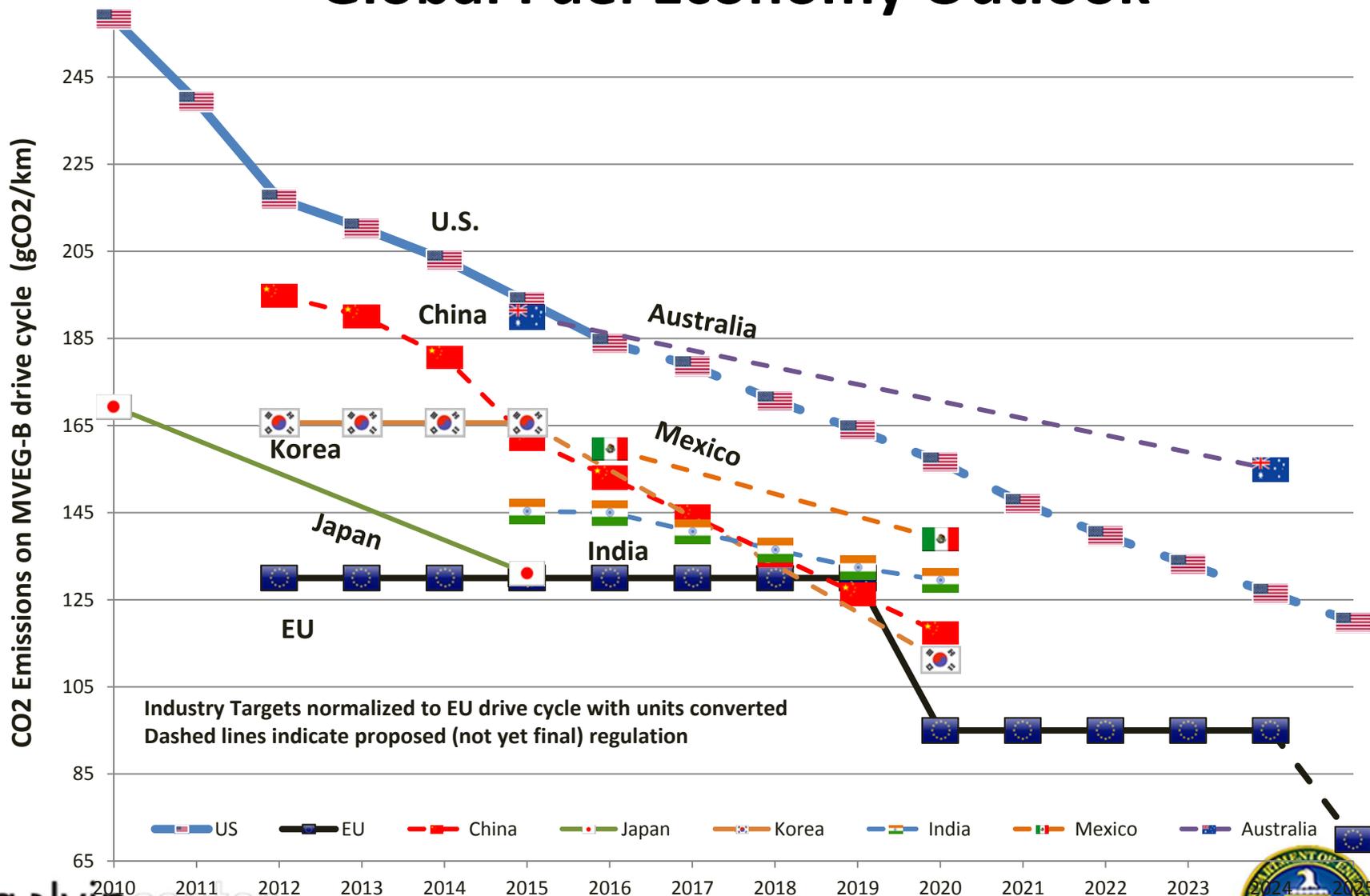


SECTION 1

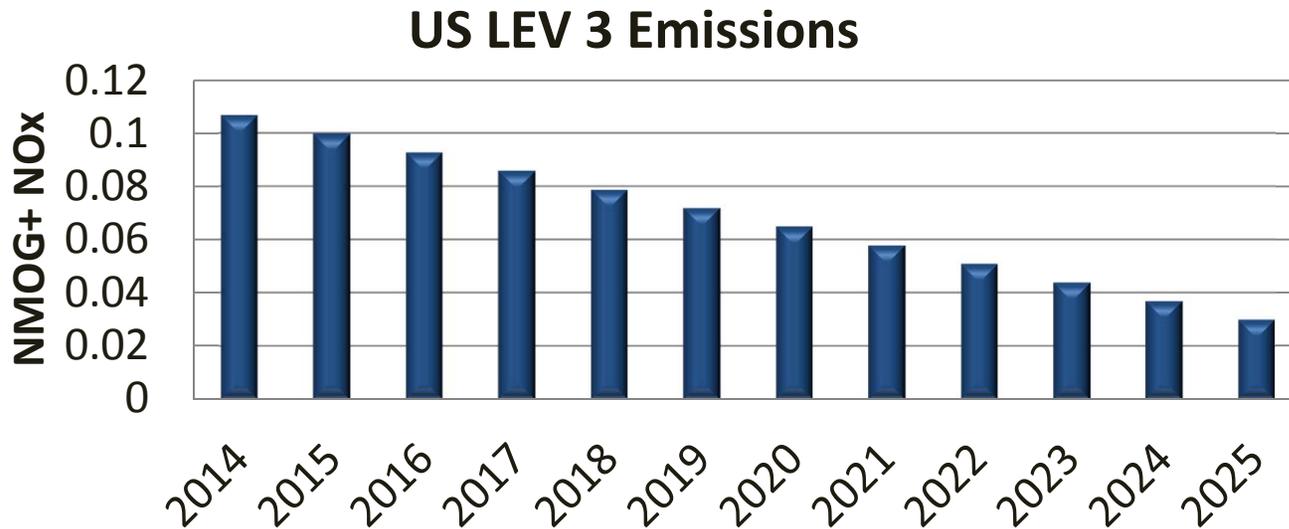
MBSE

DEFINITION, BENEFITS & REQUIREMENTS

Global Fuel Economy Outlook



Emission Regulations Are Becoming More Stringent



- Fuel Economy / Emissions Tradeoff:
 - Diesel injection timing (NOx vs FE)
 - Gas cold start: throw away fuel for catalyst heating
- Drive quality: customers will not/should not accept compromise

People Spend More and More Time In Traffic

India



China



Turkey



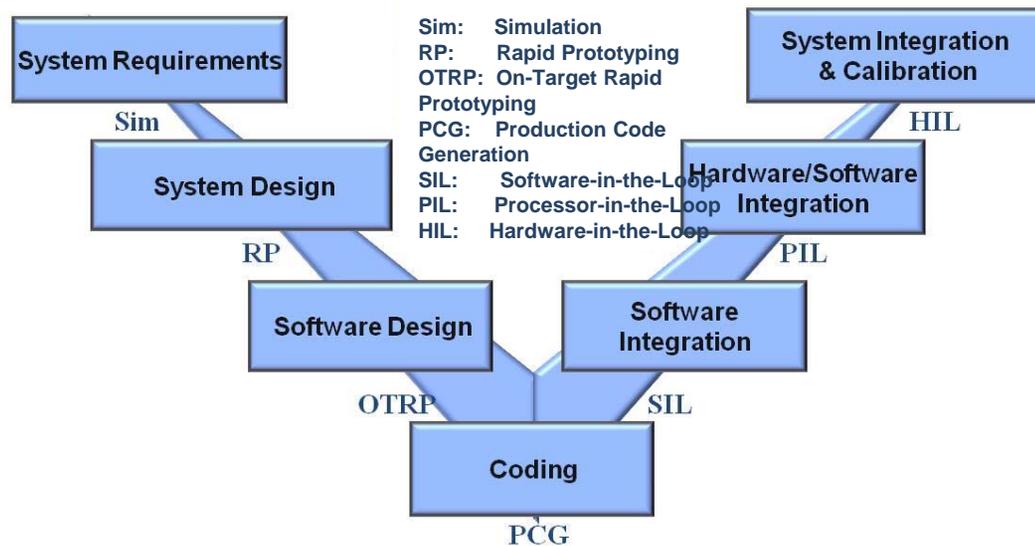
Thailand



Charlie Klein, GM Vehicle Engineering, LMS Automotive Conference 2010

Virtual Engineering Required to Accelerate the Vehicle Development Process

Virtual Engineering Process



Problem:

- Heavy reliance on hardware builds leads to high cost and longer development time
- Integration of new technologies in a system lowers the expected benefit

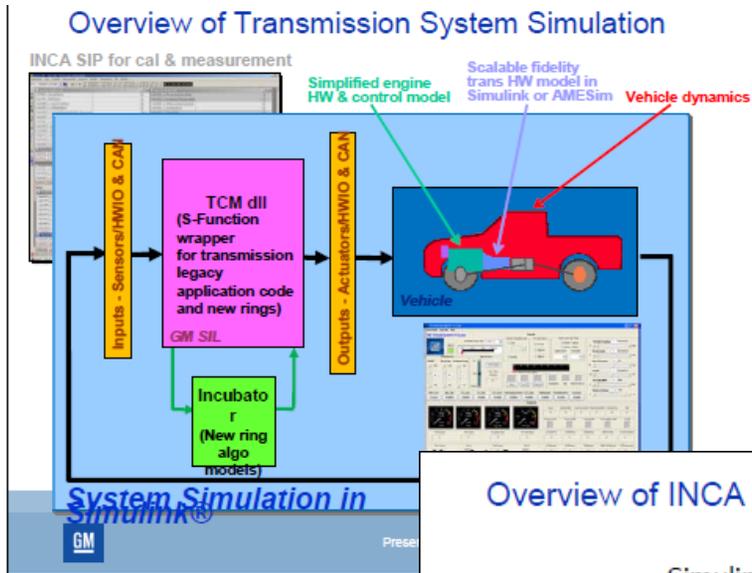
Result:

Wasted Opportunities, Time, and Resources (People & \$)

Solution:

OEMs are moving towards an increasing reliance on modeling and simulation to accelerate the introduction of advanced technologies

Automotive Companies are Moving Away from Hardware

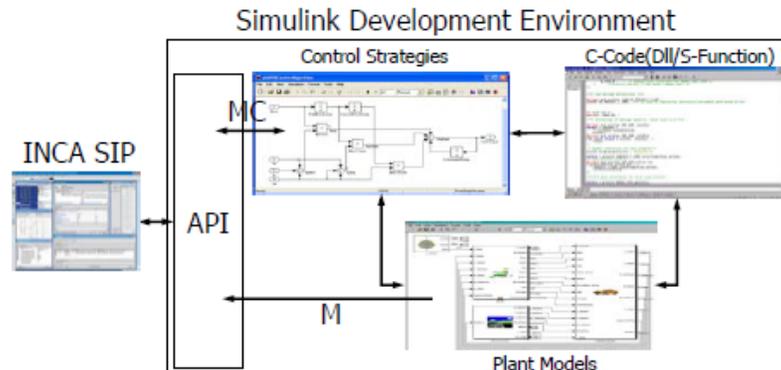


Transmission Algorithm Development using System Simulation

Transmission algorithm development can be accomplished in many ways using the System Simulation:

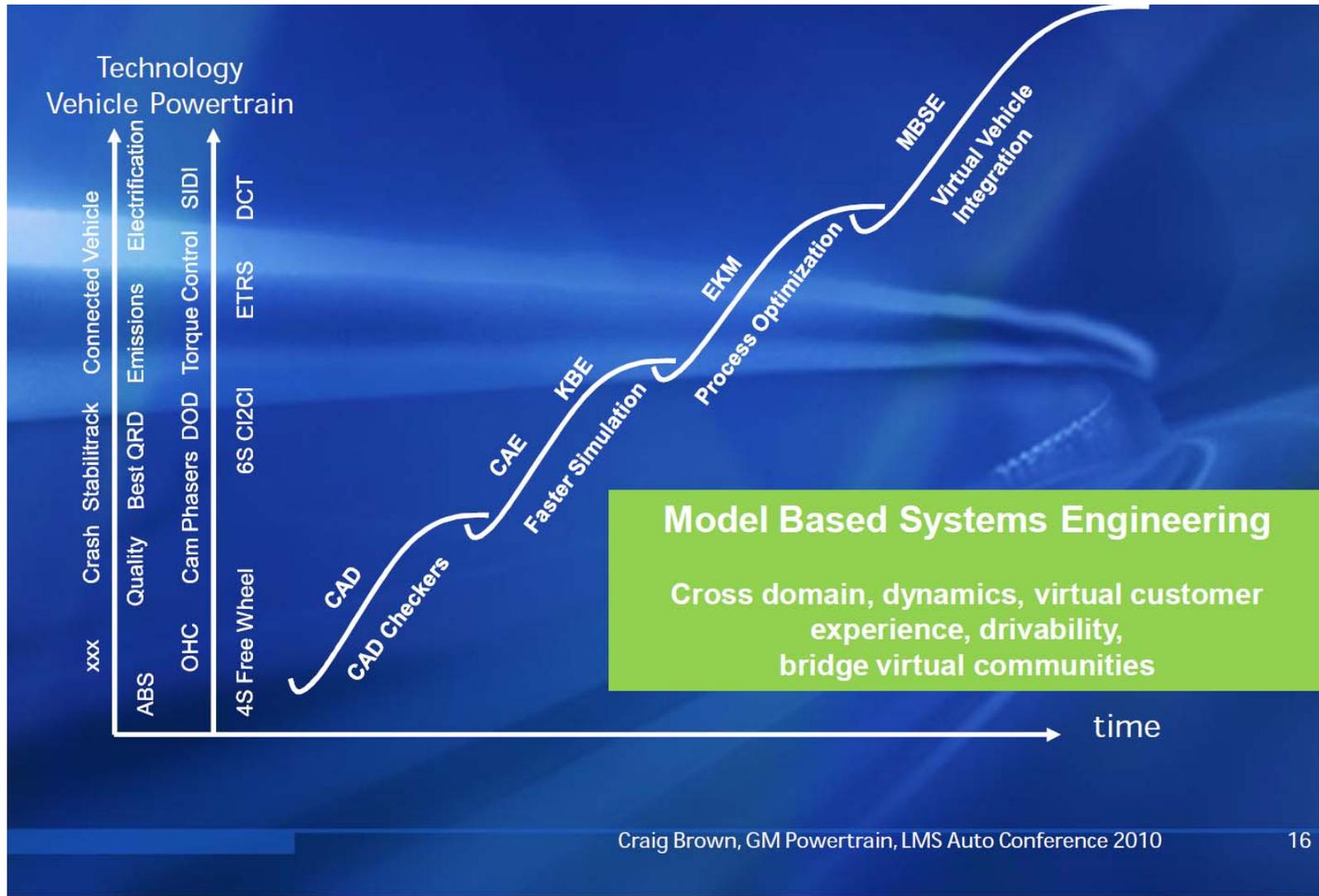
- Add brand new algorithm models in the Incubator or new hand code wrapped inside the TCM SIL block
- Modify existing algorithm models in the Incubator or existing hand code wrapped inside the TCM SIL block
- By-pass – use algorithm models in the Incubator to replace the existing ones wrapped inside the TCM SIL block
- Comparison – create multiple versions of the algorithm models and compare the results under the same driving conditions

Overview of INCA SIP



Presented at ETAS Connections, May 26, 2010 9

Product Complexity Drives Math Tool Technologies



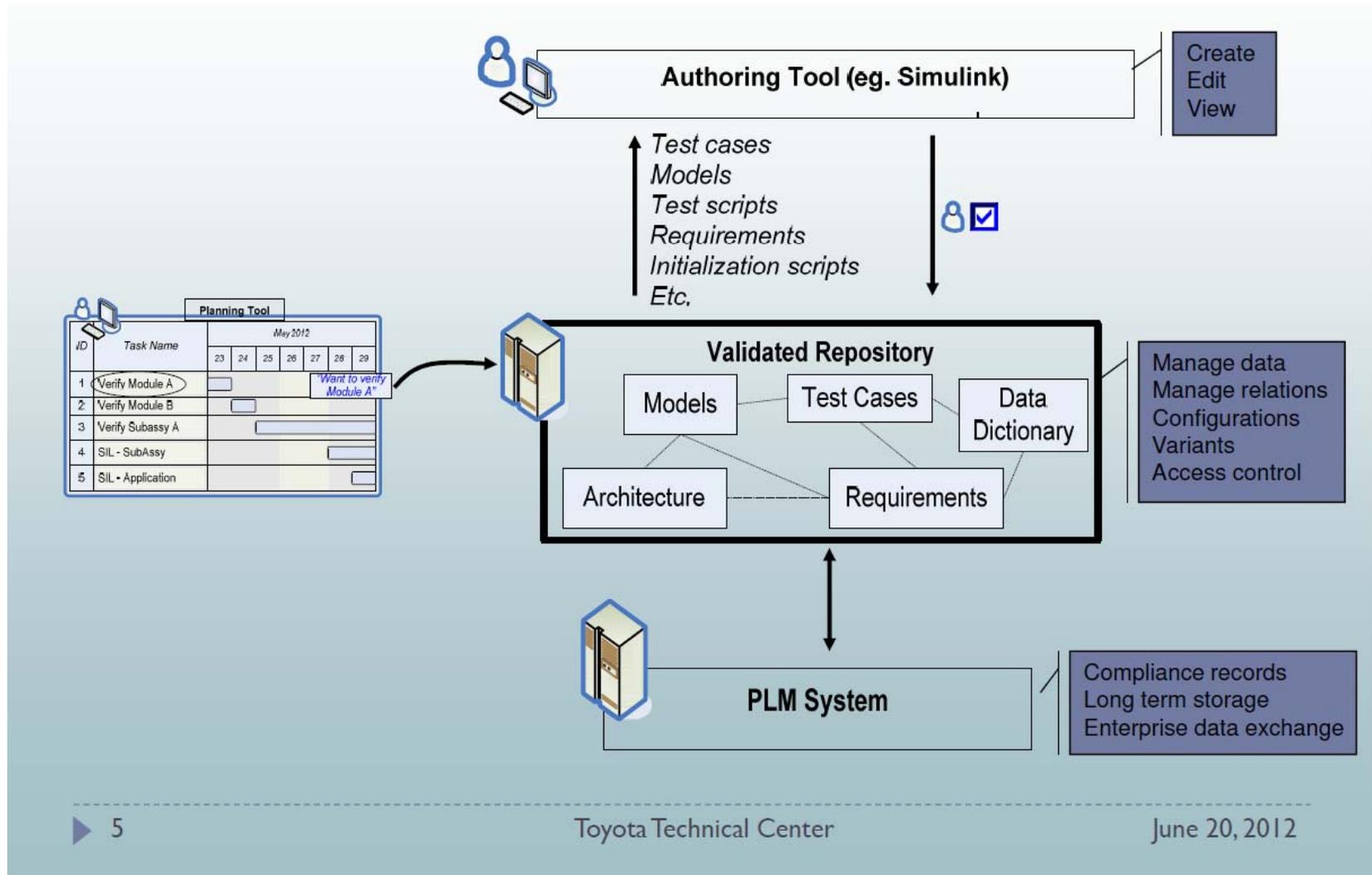
16

MBSE Definition

“Model-based systems engineering (MBSE) is the *formalized application of modeling* to support system requirements, design, analysis, verification and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phases.”

INCOSE SE Vision 2020 (INCOSE-TP-2004-004-02), Sept 2007

MBSE Requires a Collaborative Environment

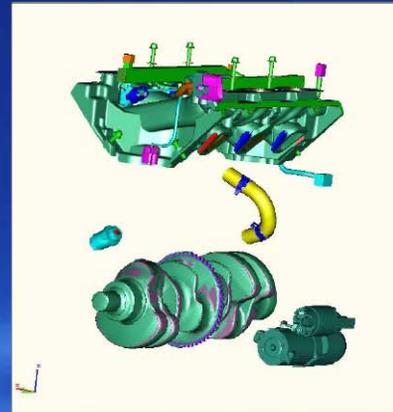


Models with Different Levels of Fidelity Have to be Developed & Integrated

Model Types

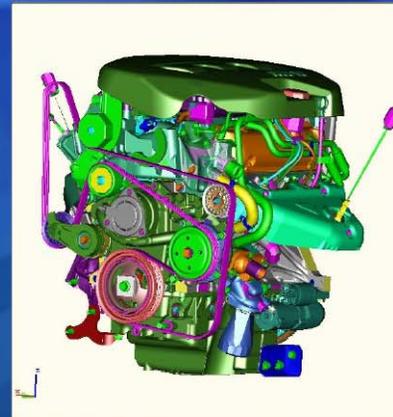
1. Component CAD Models

- Represents individual parts or sub assemblies in the as-manufactured state.
- Includes tolerance information in models in addition to nominal geometry.
- May also include alternate representations of the model to depict as-installed condition.



2. Engine Installation Assemblies

- Are packaging models for an engine application.
- Are assembled from Component CAD Models



Craig Brown, GM Powertrain, ETAS Connections 2008

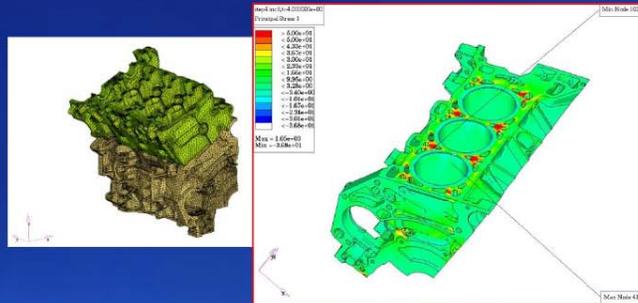
11

Models with Different Levels of Fidelity Have to be Developed & Integrated

Model Types

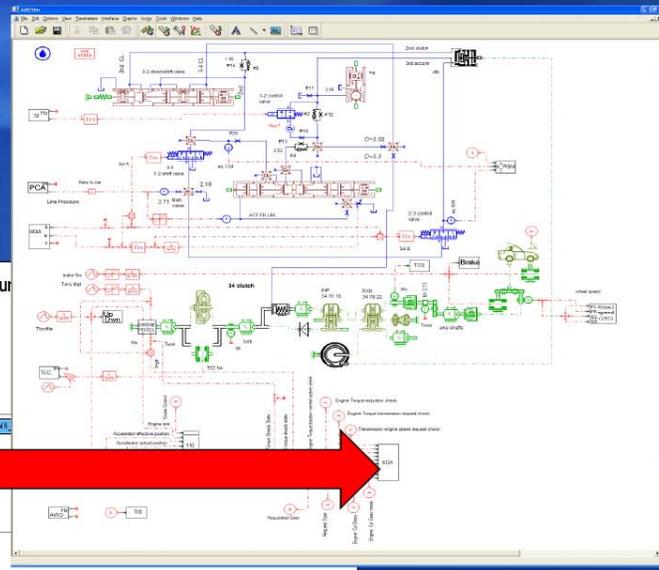
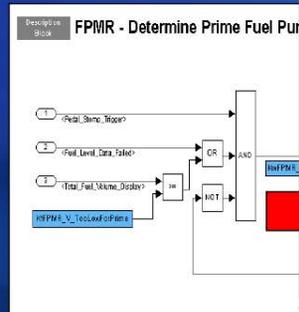
3. CAE Models

- Represent systems, sub-assemblies and individual parts
- Are created and revised by the analysts as needed from CAD Models.



4. Controls Algorithm Models

- Models that Support Controls System Design
- Signal Delivery/Variation Analysis
- Hydraulics Models for Transmissions & Engine Actuator Systems
- Engine Performance

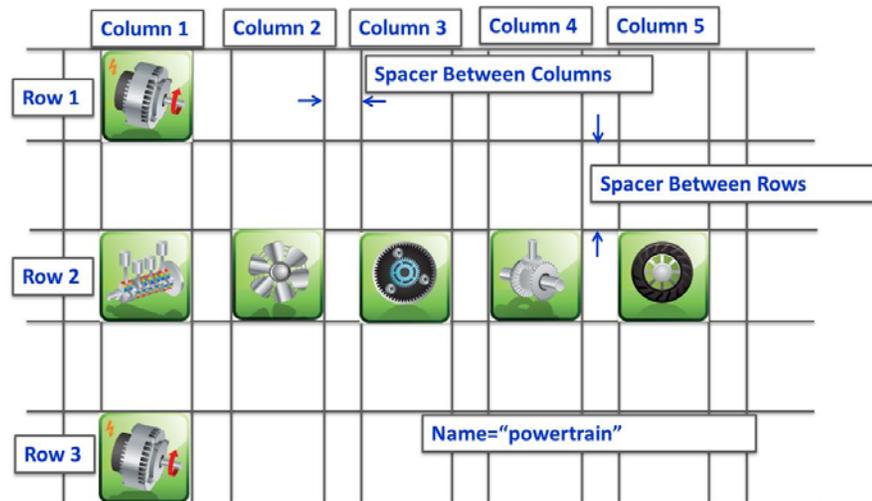


Craig Brown, GM Powertrain, ETAS Connections 2008

12

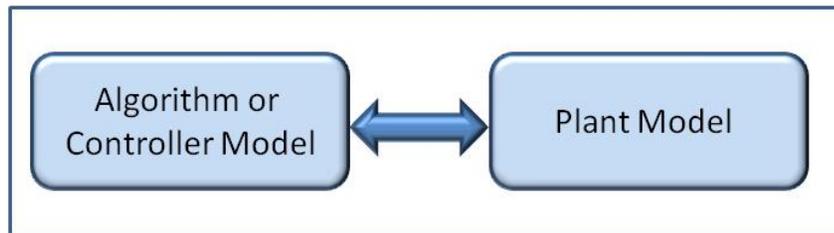
Models Have to be Assembled

- Due to the large number of system combinations (architecture x models x controls x processes), building every combination by hand would be time consuming and error prone.
- The key is to connect all the individual models and controls automatically to ensure model integrity and compatibility (i.e., I/O, units, data type...).

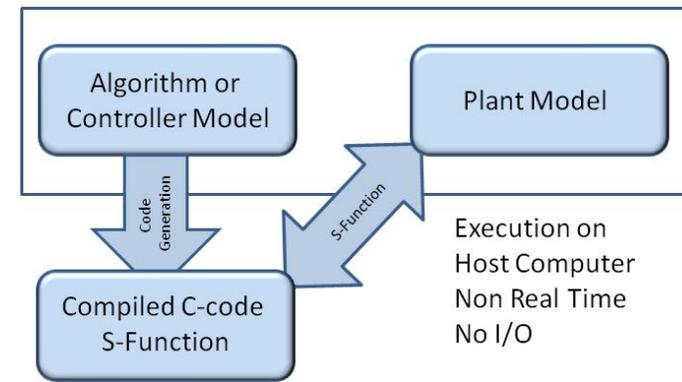


Different Methodologies Shall be Considered

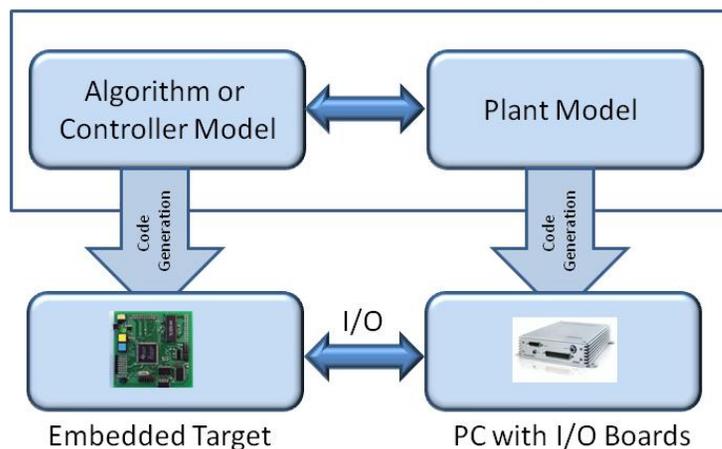
Model-in-the-Loop



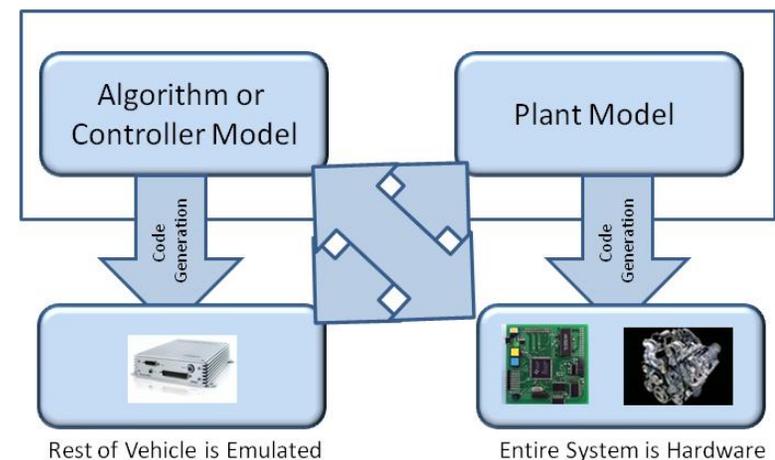
Software-in-the-Loop



Hardware-in-the-Loop



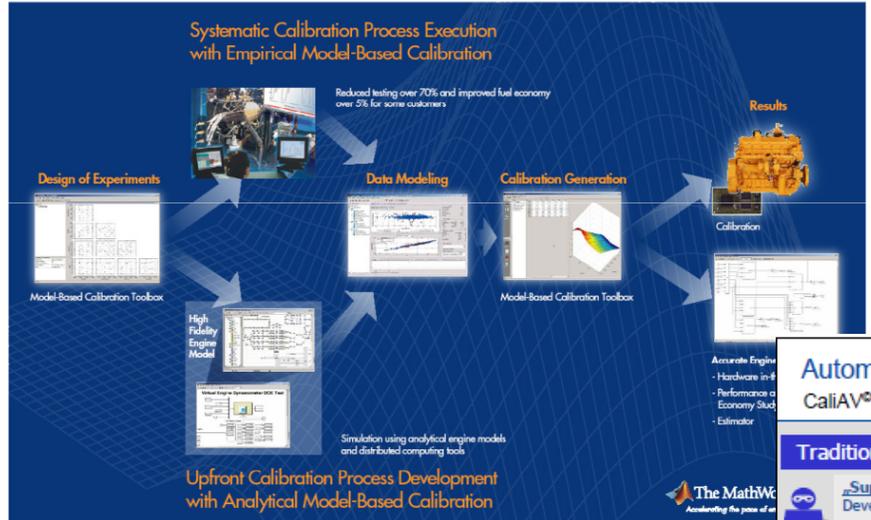
Component-in-the-Loop



Software Companies are Developing Generic Processes to Capture the "Know-How" of Experts

LMS Solutions in support of MBSE processes

Model-Based Calibration Process Overview



- Physical System Models Multi-physics & Scalable
- Model Based Controls Process, Methods, Tools ...
- Control Systems - IP Reuse
- Interdepartmental MBSE process support
- Frontloading Verification and Validation of Controls

Automated vehicle calibration with CaliAV® - Motivation

CaliAV® - Traditional vs. Guided Paradigm

Paradigm	IT skills	Calibration skills
Traditional Calibration	IT skills	Calibration skills
Super-Calibrator® Develop and run calibration procedures manually; Quality and efficiency depends on individual skills	IT skills	Calibration skills
Scripting	IT skills	Calibration skills
IT Expert Develop and deploys reusable calibration procedures	IT skills	Calibration skills
Calibrator Uses and runs standardized calibration procedures. No possibility to adapt the script	IT skills	Calibration skills
CaliAV®	IT skills	Calibration skills
Calibrator Develop and deploys reusable calibration procedures	IT skills	Calibration skills

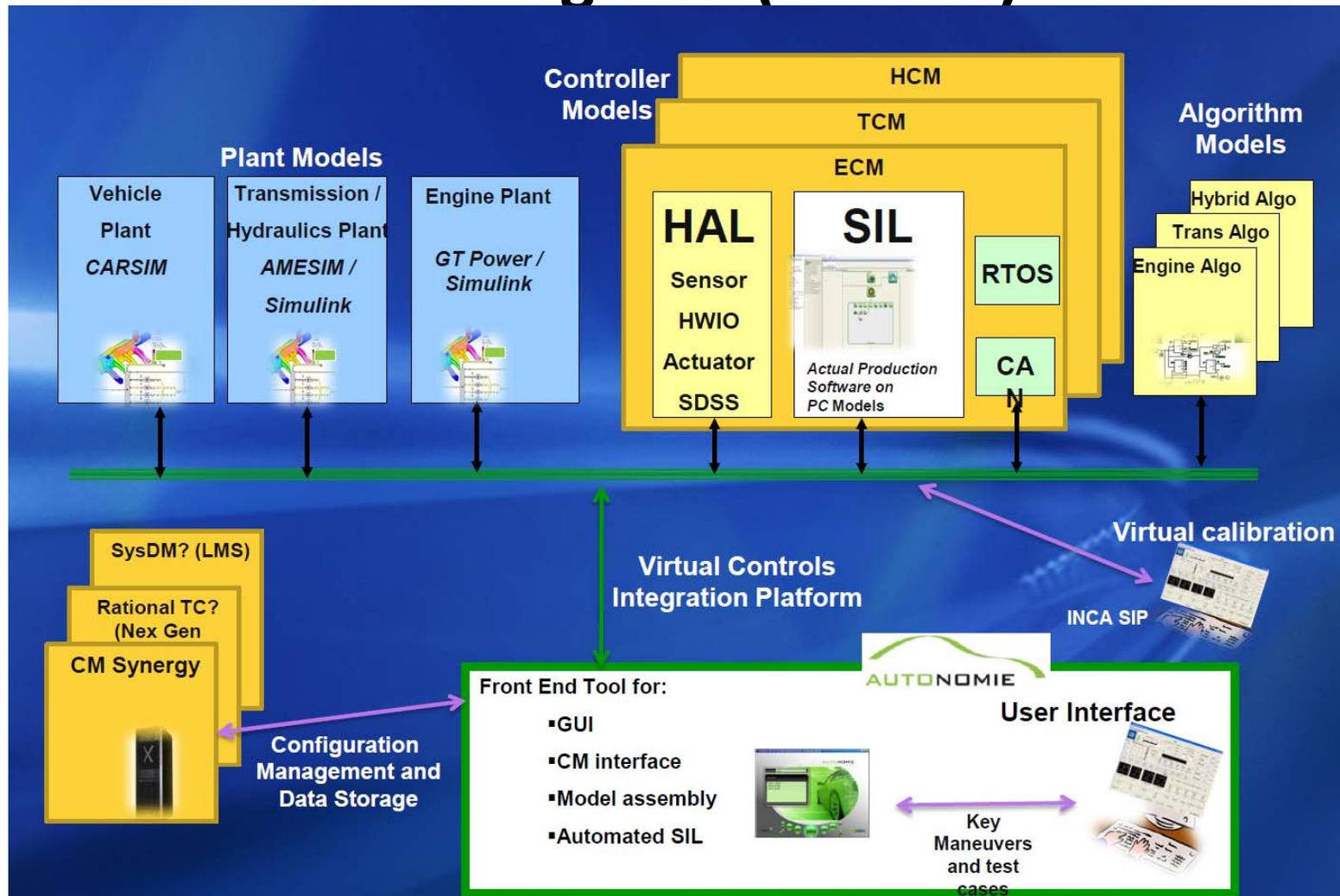
Calibration procedure will be not documented; no automation whatsoever

Calibration procedure will be documented as script file; Automation realized; IT skills necessary to perform scripting

Calibration procedure will be documented with a flow chart; Automation realized; Calibration engineer performs "scripting" with graphical elements without IT skills

Presented at ET&E Connections May 26, 2010

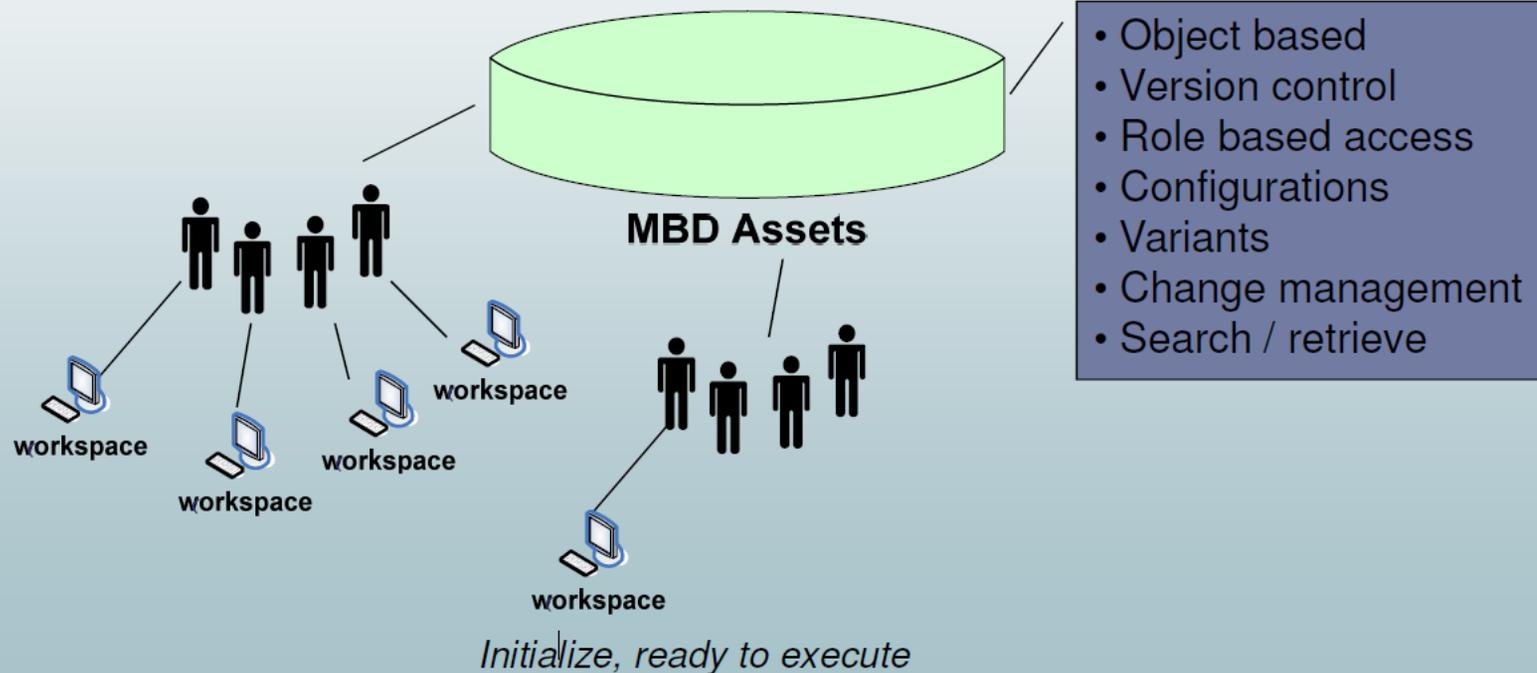
Use Case: Low level control development and test through SIL (or VHIL)



Models Have to be Managed & Accessible

▶ Collaborative Environment

- ▶ Centralized repository
- ▶ Download, ready-to-use



▶ 6

Toyota Technical Center

June 20, 2012

A close-up shot of a metallic, humanoid robot head. The robot has a highly detailed, reflective surface with visible rivets and seams. Its eyes are glowing with a bright red light. The background is dark and out of focus, showing industrial structures and lights, suggesting a factory or a futuristic city at night.

Will the
Machines Take
Over?

Testing will Always be Necessary but Its Role Has to Change



Test “Only”
(and Physical Simulation)



Test > < (Numerical) Simulation

*“Everybody believes the test data except the test engineer,
Nobody believes the simulation result, except the analysis engineer...”*

Seamless Test and Simulation Integration...

“Hybrid Engineering”

- 1) Test to Support Simulation
- 2) Simulation to Support Test

...for Better Data, Faster,
Enabling Frontloading of Development

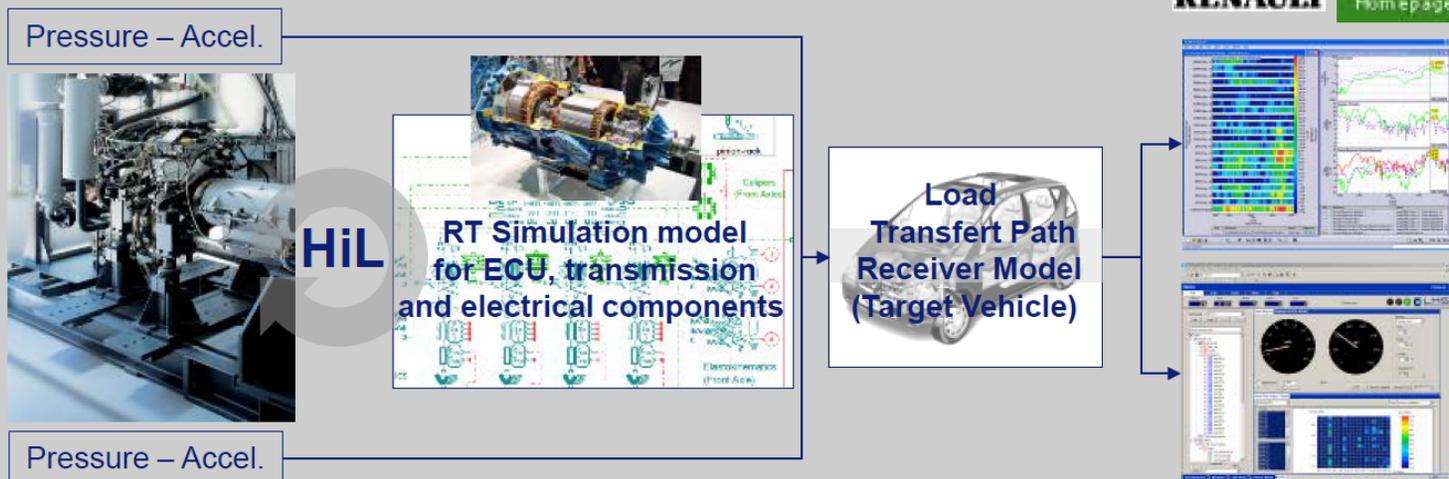
Copyright LMS International – 2012

Frontloading Physical Testing and Validation Based on Combined Use of Test and Simulation

- Simulate on the test cell the “working” of target build-in environment
- Process and analyze test cell data in context of target build-in environment

Example: Hybrid Powertrain Development

Testing and calibration of ICE to be used in hybrid powertrain



Copyright LMS International – 2012

Simulation is key to enable frontloading of testing and validation

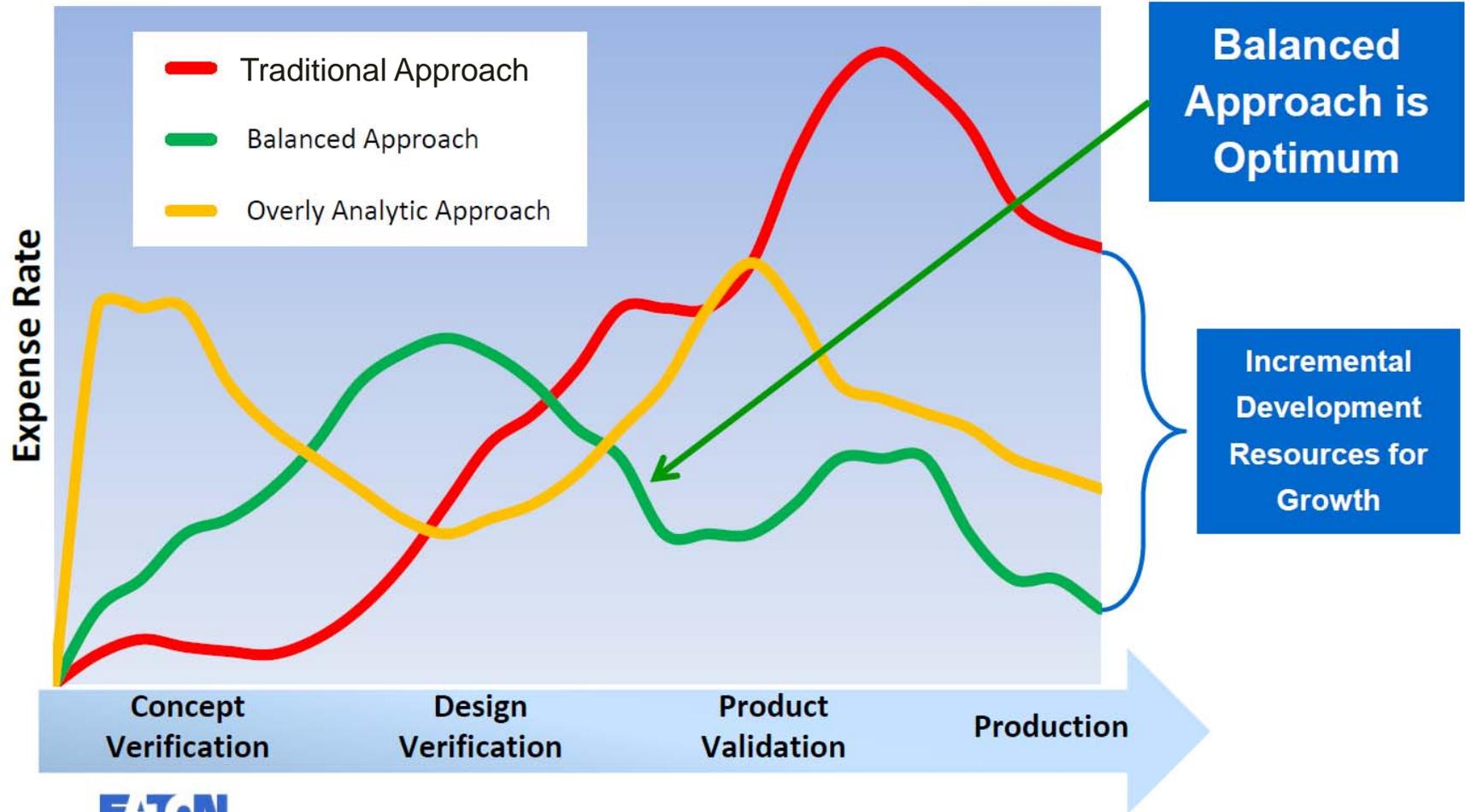
Leading Partner in
Test & Mechatronic Simulation

5



But the Process Needs to be Well Balanced

Product Development Spending Profile



Eaton Corporation Proprietary Information



MBSE is the Cornerstone of Future Vehicle Development Process (VDP)

- To make MBSE a reality, many challenges have to be simultaneously addressed:
 - Reusability (i.e., models, processes...)
 - Interoperability of multiple entities for control / plant (i.e., Simulink, AMESim, GTPower, CarSim, C-code...)
 - Flexibility (i.e., large number of options, simulations...)
 - Efficiency (i.e., quickly build a system based on numerous complex models, model fidelity vs. run time...)
 - Traceability
 - Change management (i.e., version control...)
 - Perform testing to support MBSE



SECTION 2

MBSE

IMPLEMENTATION IN

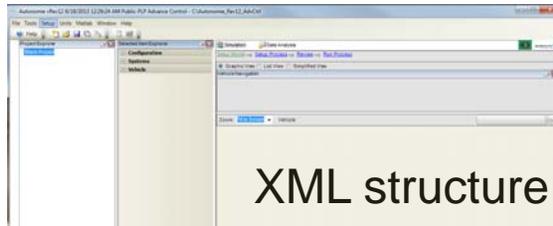
AUTONOMIE

Autonomie is Currently Composed of Two Distinct Entities

System Navigation,
Integration and Simulation

Applications

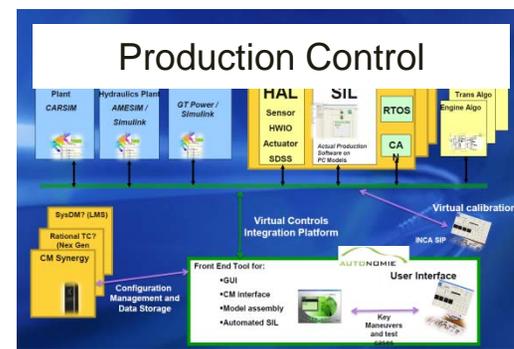
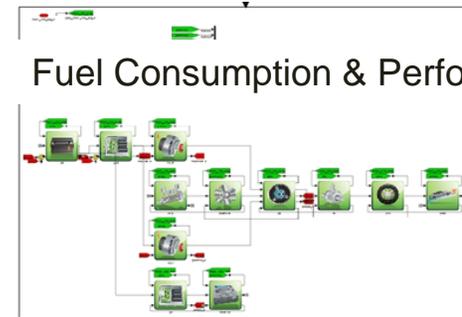
Empty Interface



XML structure

```

xml version="1.0" standalone="yes">
<process DisplayName="Run 0004 Cycle" FileName="run_cycle_script.m" operator="false" gcc="
<pre optional="false" DisplayName="Model Building" FileName="cell_system_builder.m" />
<pre optional="false" DisplayName="Initialize Data" FileName="units_initializer.m" />
<pre optional="false" DisplayName="Initialize the Cycle" FileName="cycle_initializer.m
<parameter name="file" value="aut0.mst" readonly="true" />
<parameter name="grade" value="0" />
<parameter name="mode" value="1" />
<parameter name="simul" value="1" />
<Editor dll_name="Core.UI.Common.dll" class_name="Core.UI.Common.CycleParametersCo
</pre>
<pre optional="false" DisplayName="Initialize all Systems" FileName="cell_system_initi
<pre optional="false" DisplayName="Initialize all Simulink Variables" FileName="setup
<Editor dll_name="Core.UI.Common.dll" class_name="Core.UI.Common.SimulinkParametersC
</pre>
<post optional="true" DisplayName="System Postprocessing" FileName="cell_system_postpr
<post optional="false" DisplayName="Manage Results Directories" FileName="results_dirc
<post optional="false" DisplayName="Property Manager" FileName="property_manager.m" />
<post DisplayName="Save Results" optional="true" FileName="save_results.m" />
<post DisplayName="Save Summary" optional="true" FileName="results_summarizer.m">
<parameter name="filename" value="summary_skeleton_n_report" />
<Editor dll_name="Core.UI.Common.dll" class_name="Core.UI.Common.ResultSummarizerPa
</post>
<post DisplayName="Property Manager" optional="false" FileName="property_manager.m">
<parameter name="class_properties" value="true" />
</post>
<post DisplayName="Warning Handler" optional="false" FileName="warning_handler.m" />
</process>
    
```



System Navigation, Integration and Simulation Relies on Four Pillars

Integrated functionalities through GUI

Ref. Configuration

- Tool Neutral architecture & links (XML)
- Tool Specific layout (XML)
- Large database

Model & Files Definition

- Interface definition (XML)
- Parameter definition (XML)
- Tool Neutral consistency check

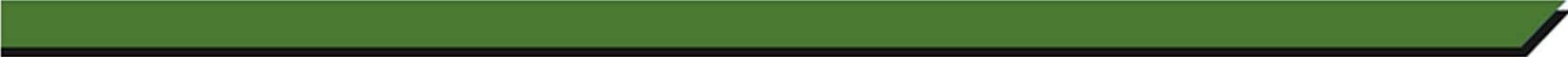


Composition Configuration

- SLK
- Multi-layer
- Patent

Process

- Flexible (XML)
- Building algorithm (modifiers)



1 - CONFIGURATION MANAGEMENT

Relationships between all Files (i.e., System) is Managed by XML

Models

Files

Configuration



```

Editor - C:\psat_gm\system\vehicle_propulsion_system\engine\plant\initialization\eng_si_1600_B5_civic.m
File Edit Text Go Cell Tools Debug Desktop Window Help
+ - 1.0 + 1.1 x
1 %% File description
2 % Name : eng_si_1600_B5_civic
3 % Author : A.Boussereau - ANL
4 % Description : Initialize the 1.6L B5KW Civic gasoline engine
5
6 %% File content
7 eng.list.init = ('time_response','eng_mass','tank_mass','fuel_mass');
8
9 eng.plant.init.technology = 'SI';
10 eng.plant.init.num_cyl_init = 4;
11 eng.plant.init.material = 'Aluminum'; %Assumption
12
13 % Fuel Parameters for Diesel
14 eng.plant.init.fuel_density_val = 0.749; % kg/L
    
```



XML

Includes list of systems, connection between systems, buses, location in Simulink...



XML

Includes documentation, parameters, user defined plots, bus information, compatibility, language, CAD files...

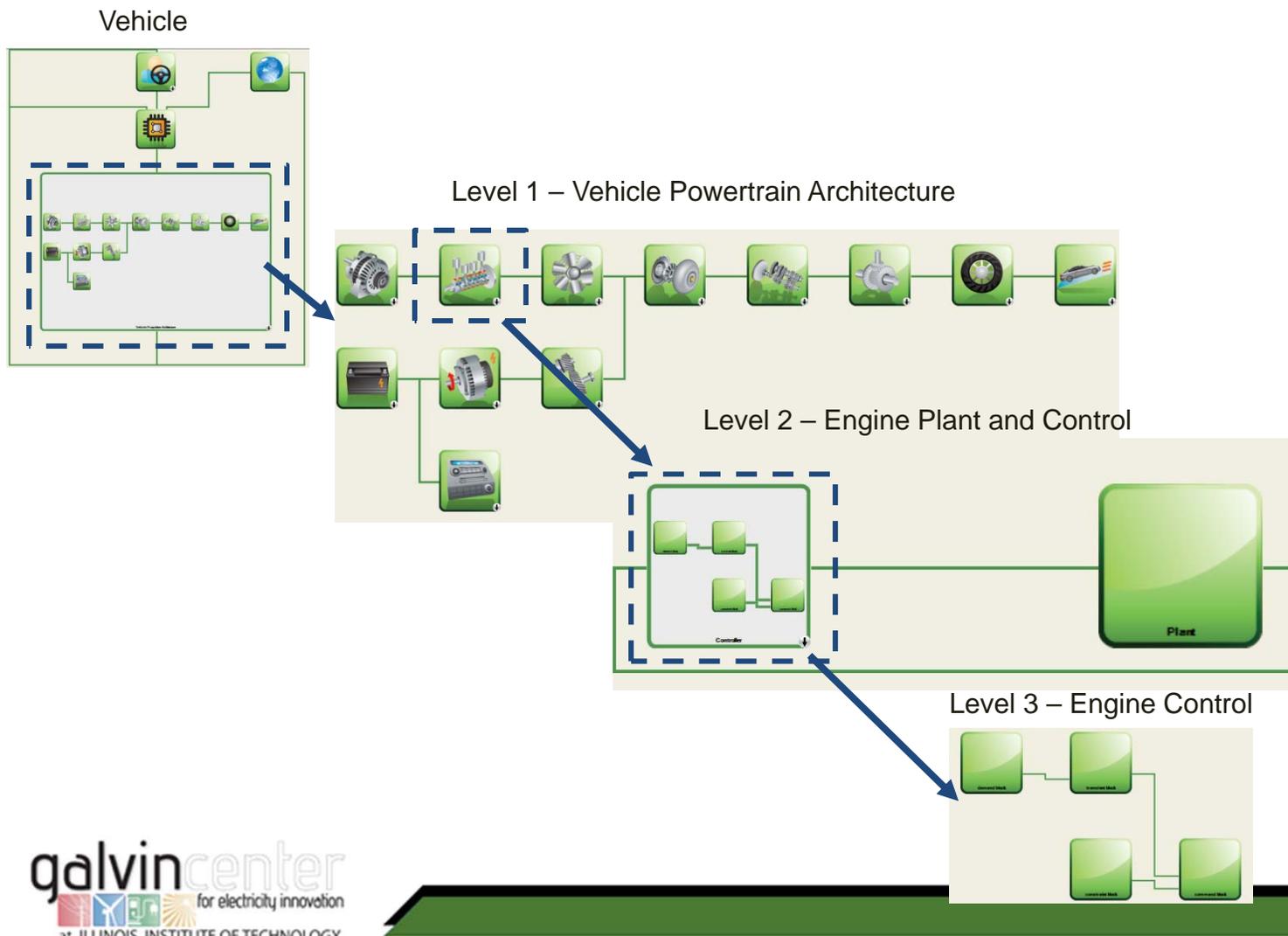


XML

Includes documentation, parameters, links to models, raw test data, calibration files...

Configurations Handled Through Hierarchical Architecture

Example of Default Conventional Automatic Vehicle

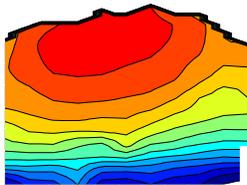




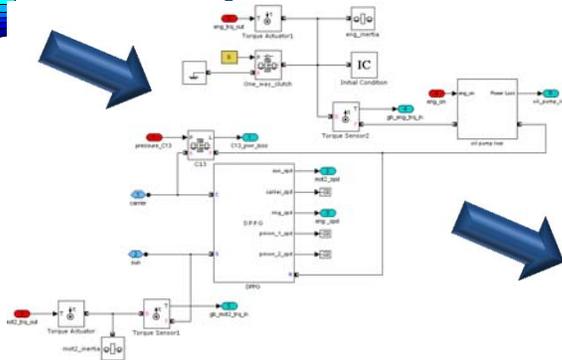
2 – MODEL & DATA SELECTION

Integrate Plant Models with Different Level of Complexity, Language

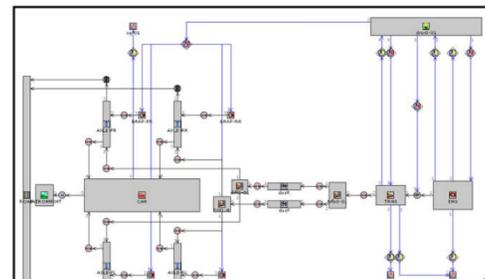
Steady State Model



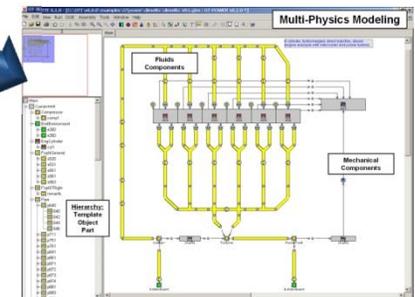
Physical Model



Mean Efficiency Model

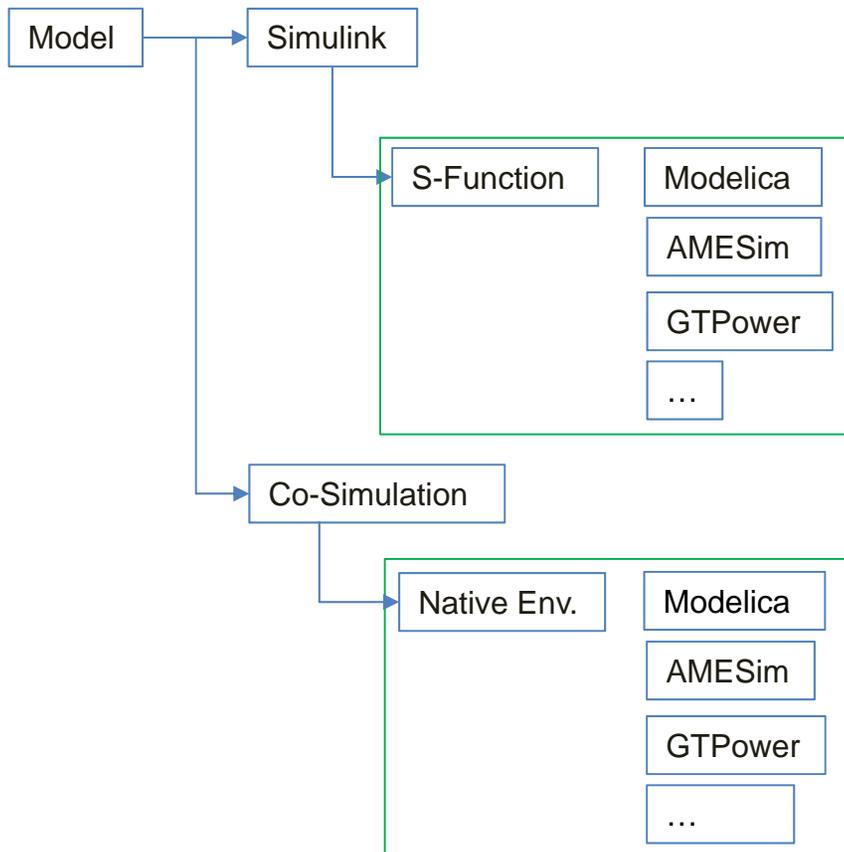


Highly Dynamic Model with Production Code



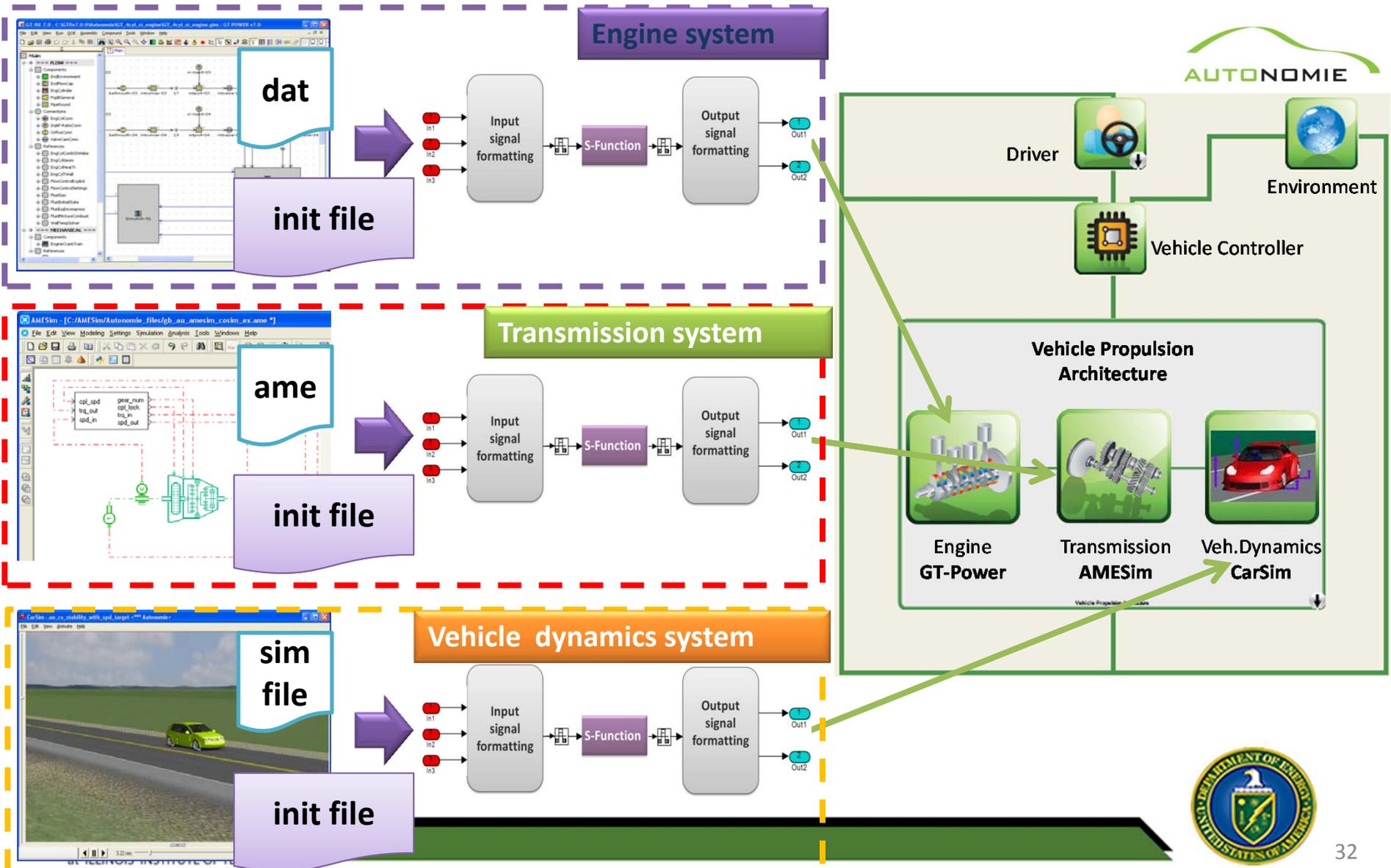
- Different models are necessary to study different phenomena
- Detailed models are required when technology cannot be tested or does not exist!

Interaction with Different Modeling Environments



- Autonomie model files link to either:
 - One Simulink model
 - One non-Simulink model via a Simulink S-Function model
 - One non-Simulink model with CosiMate I/Os for cosimulation
 - Currently working on integrating other cosimulation platforms (xMOD and ESSE)
- Compatibility / Quality control checks performed with the Autonomie files, independently of the modeling environments, and include:
 - Version matching
 - Input /Output signals connected between models
 - Parameters needed by models and other files (initialization / scaling / pre-processing / post-processing files)

Building Vehicle Using Systems Defined by Experts



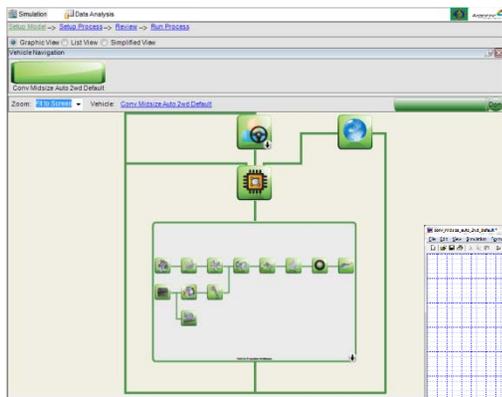


3 - CONFIGURATION COMPOSITION

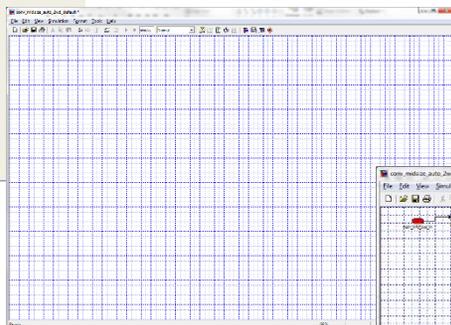
Complete Models are Built Automatically

User Select Options in GUI

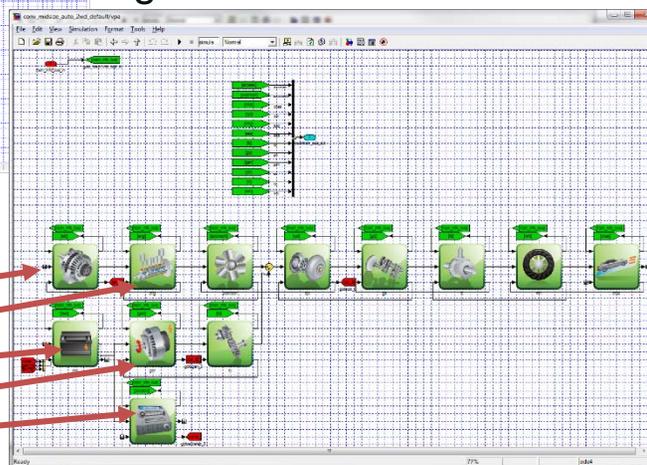
Algorithm protected by Patent



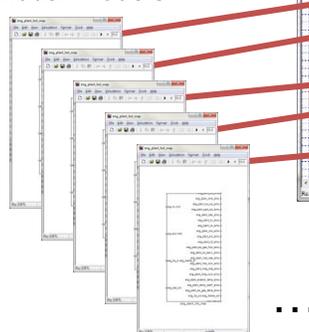
Empty Simulink is Open



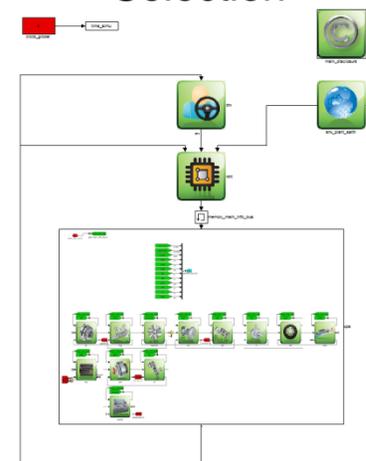
Each Model is Put in the Right Location & Connected



Individual Models

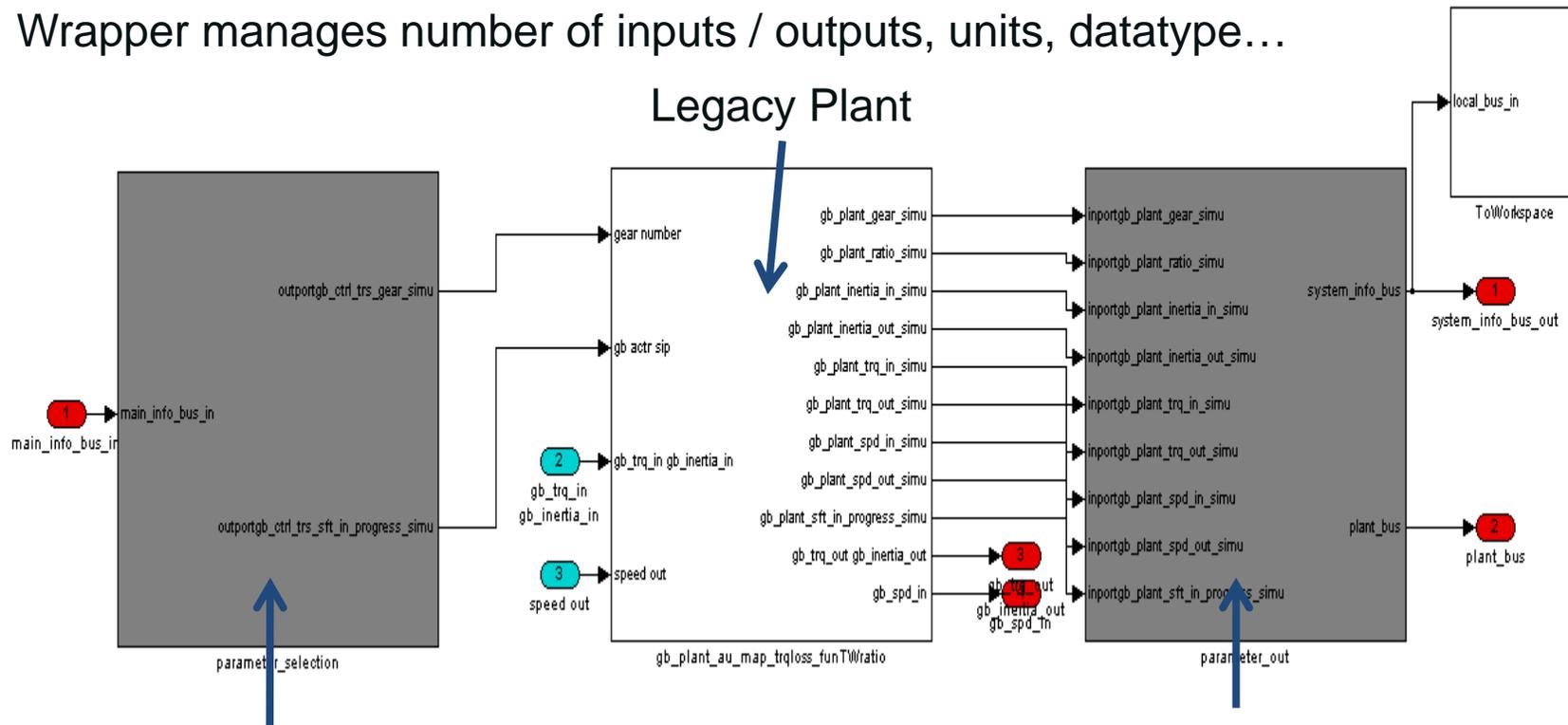


Model Build Based on Initial GUI Selection



Model Plug&Play Capability Provided by “Wrapper”

Wrapper manages number of inputs / outputs, units, datatype...



First block **automatically built** to select the required input parameters, change units and data type

Last block **automatically built** to change units (SI) and data type before sending the information to a bus to make them available to other systems

Quality Control / Check Mechanisms

- Autonomie provides a number of different checks to ensure that the final model is cohesive. Compatibilities are used to filter selections, so the user is guided to making intelligent choices:
 - Data files can be compatible with specific model files
 - Signal and parameter inputs and outputs are matched, to make sure each model has everything it needs to simulate
 - Files can be compatible with specific systems (i.e. engine models cannot be selected for transmissions)
 - Files can be compatible with other software (i.e. AMESim)
 - Files can be compatible with external libraries
 - Files can be compatible with specific versions of other files
 - Parameters are checked against the specified range
 - Signals are checked to be sure they are compatible types (i.e. a voltage signal can't be connected to a pressure signal)



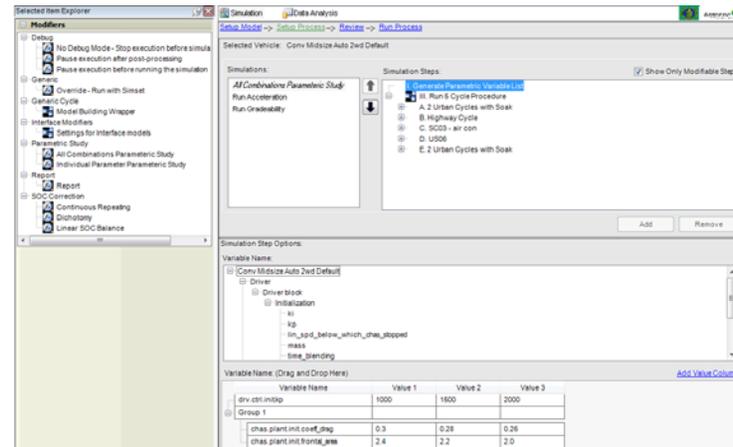
4 – PROCESS SELECTION

Complex Design Processes are Available

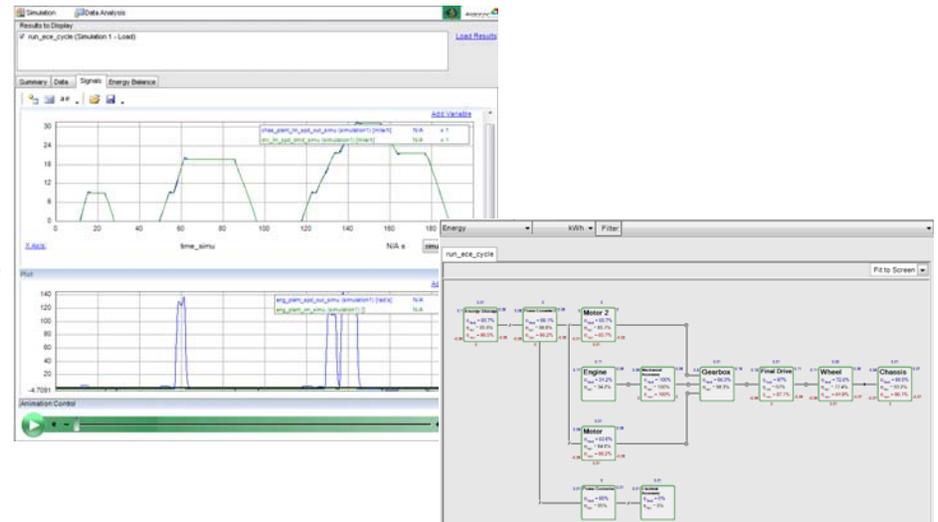
Combined and customize pre-defined processes in multiple ways



Use customizable processes to predict fuel economy (Inc. standards), performance, gradeability...



Integrate custom processes such as optimization, validation, calibration, Monte-Carlo, etc.



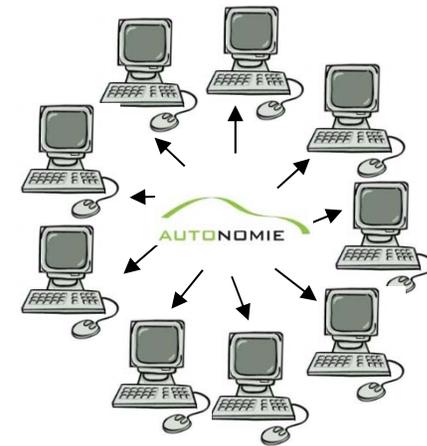
Simulation Process Adapted to the Application and the Model Complexity

- Multi-core => MPI (Message Passing Interface) ⁽¹⁾ allows Autonomie to launch multiple instances of Matlab, one for each processor core, and run a vehicle simulation on each core
- Multi-computer => Run simulations using Matlab's Distributed Computing Toolbox⁽¹⁾. It allows the user to run a separate vehicle simulation on each worker node.
- Multi-solver⁽²⁾ : co-simulation framework providing communication protocols to dynamically interact between various simulation environments

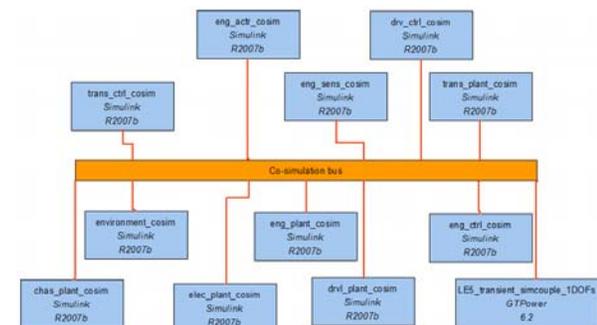
(1) Currently available outside GUI - Will be integrated into next release

(2) Linkage with CosiMate completed, linkage with EST and xMOD under development

Distributed Computing



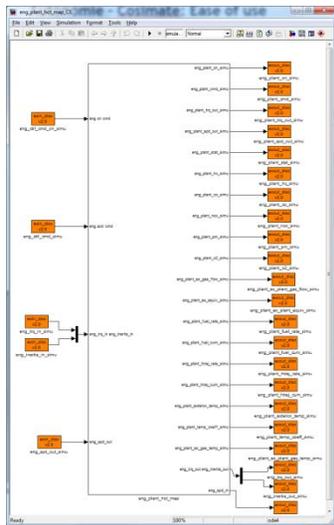
CosiMate NetList



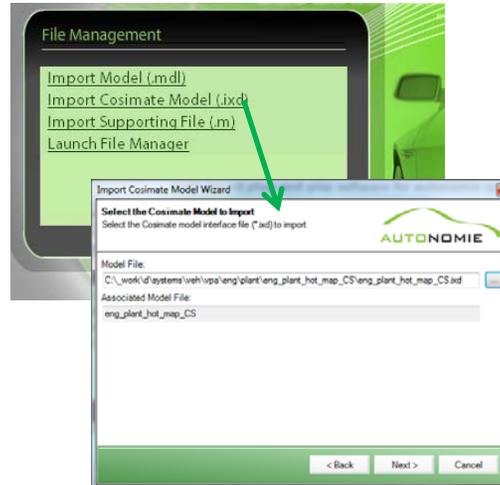
Co-Simulation Process

Cosimate Example: Model Import

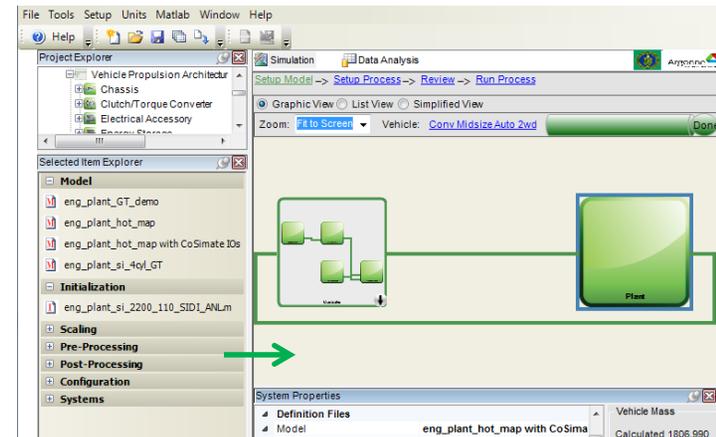
Create / Use Model(s) with Cosimate I/Os



Import Cosimate model through a Wizard



The model(s) are ready to use in Autonomie, just drag and drop them on the vehicle systems of your choice



Co-Simulation Process

Cosimate Example: Process Selection

The Co-simate process is automatically added to all the Autonomie processes (Cycle, procedure, acceleration, parametric studies, SOC Correction...)

Selected Vehicle: Conv Midsize Auto 2wd Default with Cosi

Simulations: Run Japan10 Cycle

Simulation Steps: Show Only Modifiable Steps

- I. Manage Results Directories
- II. Initialize all Systems
- III. Model Building
- IV. Initialize all Stateflow Variables
- V. Initialize Simulink Solver Options
- VI. Save System
- VII. Initialize Units
- VIII. Initialize the Cycle
- IX. Run Cosimate
- X. System Postprocessing
- XI. Property Manager
- XII. Save Results
- XIII. Save Summary
- XIV. Property Manager
- XV. Warning Handler

Generate Net List:

Cosimate Global Options

Protocol: Type: UDP

General: UDP, TCP, RCP

Spy Data: False

Excel Trace: False

Verbose: False

VCD Trace: False

Connection Waiting Time: Wait for (s)

Co-Simulation Overwrite: Simulation Mode, Time Step (s)

Simulation Overwrite: Start Mode, Start at (s), Stop at (s)

Cosimate Model Options

Path	Name	Tool	Version
veh		Simulink	7.10
veh\paleng\plant	eng_plant_hot_map_CS	Simulink	7.10

Co-Simulation Options

Co-Simulation Mode: Synchronized

Time Step: 0.01

Start Position (-1 for default): -1

Simulation Options

Starting Mode: Automatic

Start Time: 0

End Time: 135

Environment

Host: localhost

Login:

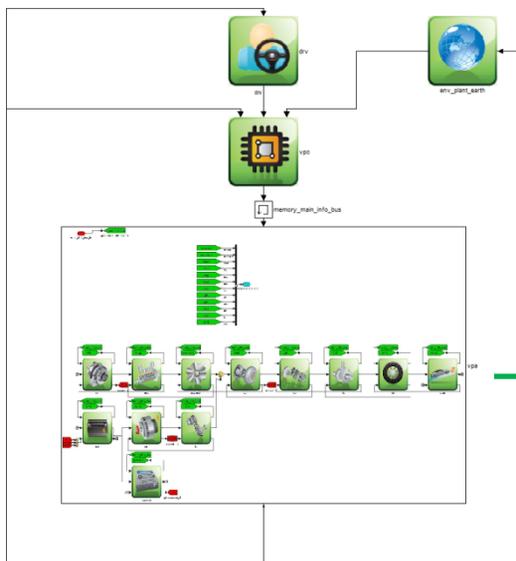
Command:

All the Cosimate Options are available to the user

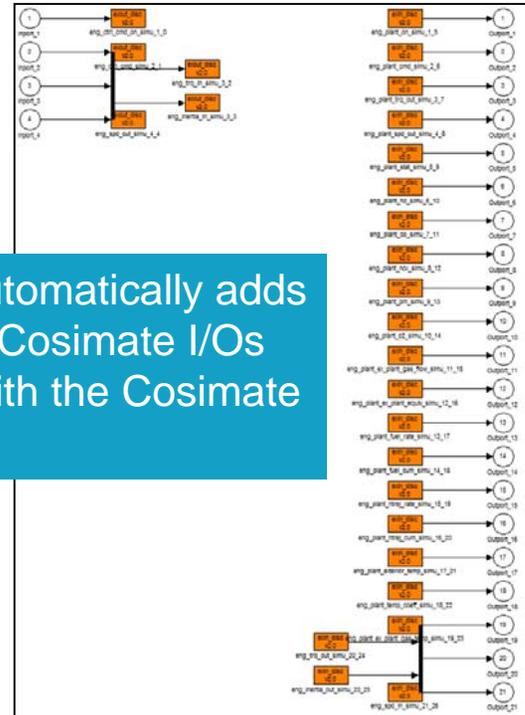
Co-Simulation Process

Cosimate Example: Model Building

Autonomie automatically build the vehicle model without the Cosimate ones



Autonomie automatically adds the matching Cosimate I/Os to interface with the Cosimate Model(s)



Autonomie automatically generates the matching netlist and launches it through Cosimate.exe

```

1 <?xml version="1.0" encoding="utf-8"?>
2 <System Name="conv_midsize_auto_2wd_default_with_cosimate_engine">
3   <Options>
4     <Protocol Name="UDP"/>
5   </Options>
6   <Model Host="localhost" Name="conv_midsize_auto_2wd_default_with_cosimate_engine" StartTime="0" Type="Simulink" TypeID="10" Version="7.10">
7     <CosimMode>
8       <Synchronized TimeStep="0.01"/>
9     </CosimMode>
10    <StartMode>
11      <Automatic EndTime="135"/>
12    </StartMode>

```



Summary

- MBSE provides benefit to companies. As a result, companies are developing plant and control models to increase its usage.
- Numerous software companies are developing tools and processes in parallel.
- Companies are still not sure where to go, but they are still moving forward!
- Numerous issues still need to be addressed to make MBSE a success, including
 - Model fidelity vs. run time
 - Growing & maintaining CAE model library(ies)
 - Interoperability of multiple entities for controls/plant representation (GT Power, AMESim, CarSim, Simulink, C-code)

Summary

- Autonomie has been developed to provide a Plug&Play Software environment to support MBSE
 - Provides ready to go plant, control and processes for fuel consumption and performance.
 - Build automatically complete simulations
 - Allow users to customize models and processes
 - Support processes from research to production (MIL, SIL, HIL, RCP, CIL)
 - Tool neutral
- But more development still need to be done as well with Autonomie to better support MBSE...