SIEMENS
Ingenuity for life

Siemens Digital Industries Software

Consistent test reuse across MIL → SIL → HIL in a model-driven development workflow

Embedded system

Executive summary

Deploying safe and secure automotive-grade embedded software is inherently complex. It requires a massive amount of verification and validation (V&V) effort that is ever-increasing with the advent of advanced technologies for advanced driver assistance systems (ADAS), automated driving systems (ADS) and autonomous vehicles (AV). To address these challenges, the automotive industry is being bombarded with an expanding number of different specialized software tools from different vendors that must somehow work together. To be successful in terms of on-time, on-budget and on-quality, it is essential that V&V suites developed throughout and across car projects are reusable.

Lance Brooks

Contents

Workflow and verification

Development methodologies and V&V tools have advanced over the past few decades. Today, the MDD methodology and XIL verification are well-established as an effective means to develop safe and secure vehicle E/E subsystems.

In the MDD systems engineering process, or generative workflow (figure 1), a model of the ECU is tested against a model of the vehicle system's communication networks, sensors, actuators and plant environment that surround the ECU: the so-called "MIL" or model-in-the-loop level of abstraction. Once the model is validated, C/C++ code is auto-generated from it and retested: the so-called "SIL" or software-in-the-loop level of abstraction. Eventually, the generated code is integrated into ECU hardware and platform software (a.k.a. firmware) and again retested: the so-called "HIL" or Hardware-In-the-Loop level of abstraction. HIL-level testing can also be performed utilizing models of the ECU hardware: the so-called "vHIL" or virtual Hardware-In-the-Loop level of abstraction.

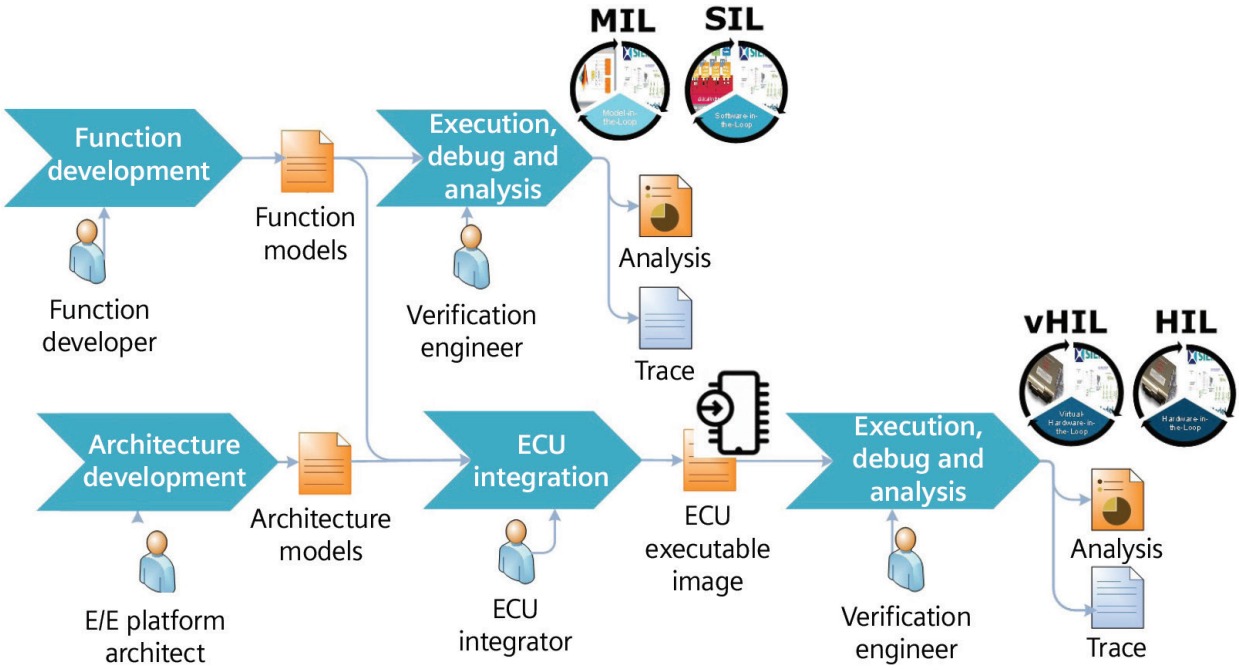

Figure 1: The generative MDD/XIL workflow produces correct systems using automation.

The different abstraction levels of XIL configurations found across the MDD/XIL workflow are summarized in (figure 2).

The range of accuracy (or fidelity) of the various XIL configurations can be quite wide. The vHIL configuration that leverages virtual ECU simulation technology covers the broadest range. In the vHIL configuration, the accuracy of the ECU hardware model can scale whereas the platform and application software remains the actual code that deploys in the final vehicle (similar to HIL). This facilitates testing final production software on a platform that is optimal for the verification intent.

With advanced MDD tools, integration can also be generative within the workflow.

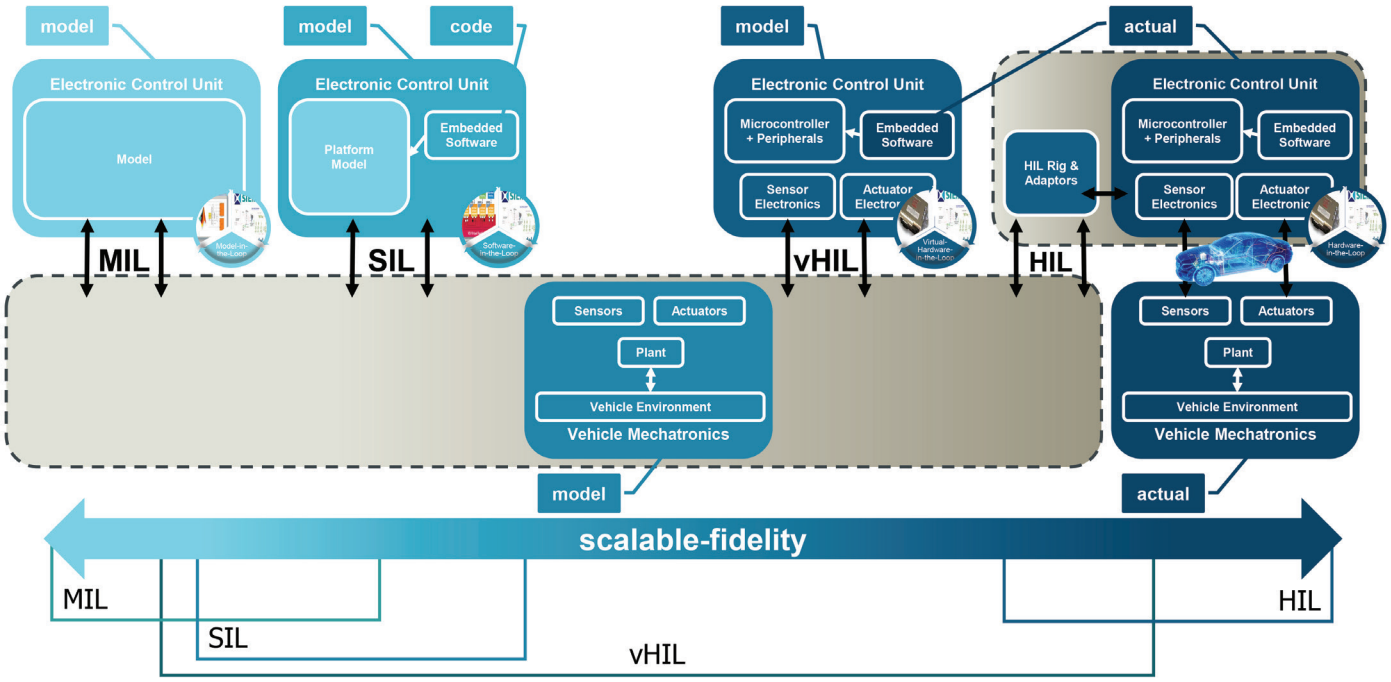

Figure 2: An overview of XIL test bench configurations.

Test benches and test automation

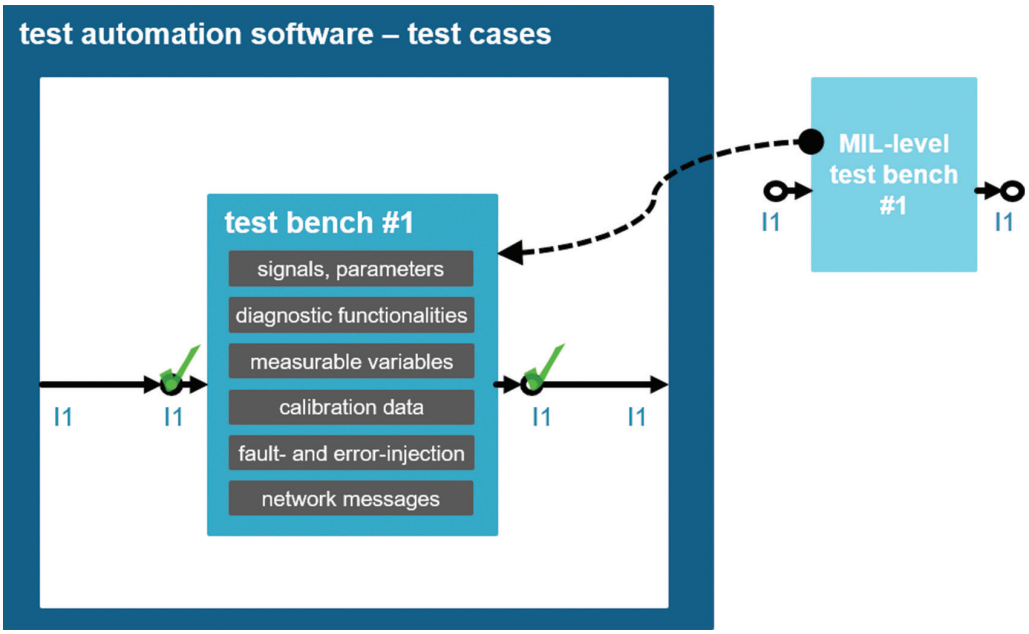

Figure 3: Test automation software sequences the SUT and accesses data within it using interfaces exposed by test benches.

There are two major elements in the MDD/XIL testing architecture: the test bench and the test automation software that executes test cases (figure 3).

Test bench

Testing an XIL system begins at the test bench. The test bench is an abstraction that provides the ability to execute a system under test (SUT) and access the data within it. The SUT includes ECUs, networks, mechatronic hardware and other simulated parts of the system. Each type of data or access requires a different interface and the inform-ation is configured and managed by differ-ent specialized tools from different vendors.

For automotive E/E subsystems, the primary data and accesses of interest include:

- Signals and parameters in embedded software and simulation models
- Data from the diagnostic functionalities of ECUs
- Measurable variables and calibration data within automotive software
- Data controlling fault- and error-injection within the SUT
- Signals encoded into messages communicated over an automotive network

Test automation software

Supporting the testing sequence – or test case – test automation software utilizes the interfaces exposed by the test bench to access the data and exercise the sys-tem's functionality. The test cases must have enough coverage to V&V all aspects of the system that are appli-cable at each phase of the MDD/XIL workflow.

Testing challenges

My most important (and rewarding) responsibility is to regularly work with major automotive OEMs and suppliers – in particular within the function development departments. By doing this, I learn about the challenges on both the design and the verification sides of the so-called "V" development process, as well as down in its implementation valleys.

A major observation I continuously make is how well teams utilize the MDD/XIL workflow yet surprisingly how little test reuse exists across their development processes.

On one hand I am continuously impressed by how well MDD/XIL is successfully deployed within the automotive industry. Really, it's a major technological success story! The idea of using computers to automatically transform modeled functions into production quality embedded C/C++ code was once considered impractical, yet is now well-established as the most effective way to develop software.

However, because of several factors such as the tiered ecosystem where several teams in different companies collaborate over large process gaps, the sheer complexities involved in developing modern E/E systems, and because of the relatively slower pace of standardization relative to the pace of new methodology adoption, there remains large disconnects between V&V at the model level and V&V at the implementation level.

It is clear this lack of test reuse is a significant issue. A huge business ramification is a reduced amount of V&V coverage if every test case is not converted or back-adapted to each level of abstraction. This causes issues in both directions: production code is not exercised against all important functional scenarios that are validated at the model level; and an overreliance on expensive HIL equipment that is not always available to every software engineer.

Perhaps a larger business issue are the problems not found early in a project when they are less expensive to fix. Even worse, issues uncovered in the field can be disastrous to the business as a whole.

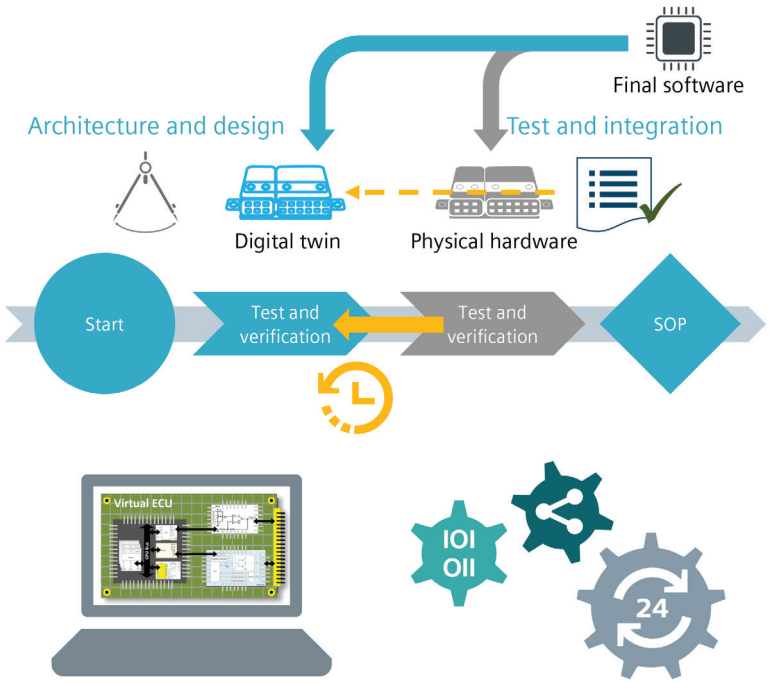

Figure 4: Shift-left test and verification using a digital twin to find problems earlier.

Factors that effect test reuse

The main factors that contribute to the lack of test reuse are seen in figure 5, these include:

- Levels of abstraction of the data and signals within the test benches
- Modeling and programming languages
- Sequencing and control over the various specialized tools
- Means to stimulate and trace the SUT
- Lack of standardization

If these factors are not addressed adequately, equipment- and tool-dependent tests are developed multiple times, often by different people with varying skill sets.

Different levels of data abstraction

A major challenge is when signals are represented, encoded and interfaced at different levels of abstraction as the system is re-integrated moving from MIL to SIL to HIL. As an example, a signal might be encoded as a double precision software variable at the MIL-level, yet it ultimately will be realized within a CAN network message at the HIL-level or as a calibrated 32-bit integer at the SIL-level. How does a test case consistently access data that changes in its data structure throughout a project without any modifications?

Different languages

Throughout the systems engineering process, different languages are used. A modeling language such as Simulink, Amesim, the UML, or Modelica might be used at the MIL-level whereas a programming language like C/C++ is typically used at the HIL-level. How does a test case read and write values in a SUT consistently throughout development as the languages change?

Different tools and simulators

Many different tools and simulators are needed to validate a modern ECU. Siemens Digital Industries Software Amesim might be used in a MIL-level testbench to describe a plant model and Capital® VSTAR Virtualizer could be used at the vHIL-level. Automated driving tests require environment scenario tools like Siemens TASS

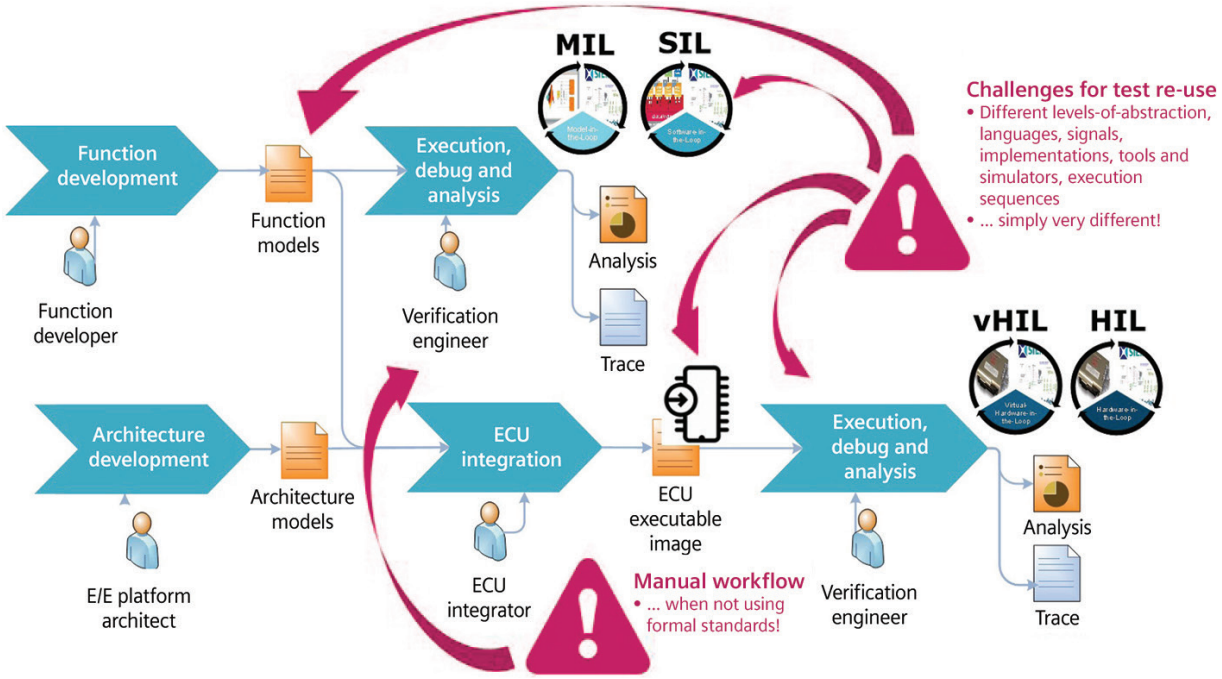

Figure 5: Several factors contribute to the challenges within the MDD/XIL workflow.

PreScan™. Starting up a HIL rig is very different from launching a virtual ECU. These are just a few examples. How does a test case know about configure, initialize and how to control the various tools used as a project progresses?

Different means to simulate and trace

Each tool has a different way to extract signals and parameters; obtain data from diagnostic functionalities; measure variables and access calibration data; provide a means to control fault- and error-injection; and parse and extract signals packed within automotive network messages. How does a test case stimulate and trace data consistently as test benches change?

Lack of standards

Without standardization, test case reuse is simply unattainable because otherwise there is no standard way to:

- Communicate between test cases and test benches
- Describe, configure and initialize test benches
- Map and access data within test benches
- Generate test artifacts automatically using MDD model transformations

The business problems related to a lack of standardization are:

- No ability to efficiently exchange test cases and test benches between OEMs and suppliers
- Increased training costs because know-how cannot be easily transferred
- Prohibitive costs to switch between testing technologies and products
- Verification engineers cannot combine the best test automation software with the best test bench products

Issues with direct-coupled test architectures

If the test automation software is directly coupled to the test bench using a rigid interface, whether it's standard or not, test reuse is not possible across the MDD/XIL workflow.

A further major drawback of direct-coupled test architectures is the verification apparatus does not readily support a heterogeneous configuration where MIL-level test benches can be mixed with HIL- and/or vHIL-level test benches. These heterogeneous configurations are needed to efficiently test multi-ECU systems, which are typically required to validate modern E/E systems for ADAS, ADS and AV.

MIL-level workflow with direct-coupling

The MIL-level workflow that uses direct-coupling starts with the development of test cases that are rigidly connected to the first MIL-level test bench. Everything appears good enough. However, when you consider that even at the MIL-level different representations of the system are possible, you quickly discover rigid interfaces do not support test reuse because the interfaces simply do not match. As an example, if a MIL-level system #1 is modeled using language #L1, data type #D1 and tool #T1, its test bench interface #I1 will not match the interface #I2 for a MIL-level system #2 that uses language #L2, data type #D2 and tool #T2.

XIL-level workflow with direct-coupling

By examining the rest of the MDD/XIL workflow with direct-coupled test benches and test automation software, you see the problems expand significantly because of the many different languages, levels of data abstraction and the number of different specialized tools that must be controlled (figure 6). It is most certain that none of the downstream test benches will expose interfaces that are compatible with the original MIL-level test benches.

In a direct-coupled test architecture, test development must be performed multiple times, most likely by different groups. This means function development is more isolated from ECU integration.

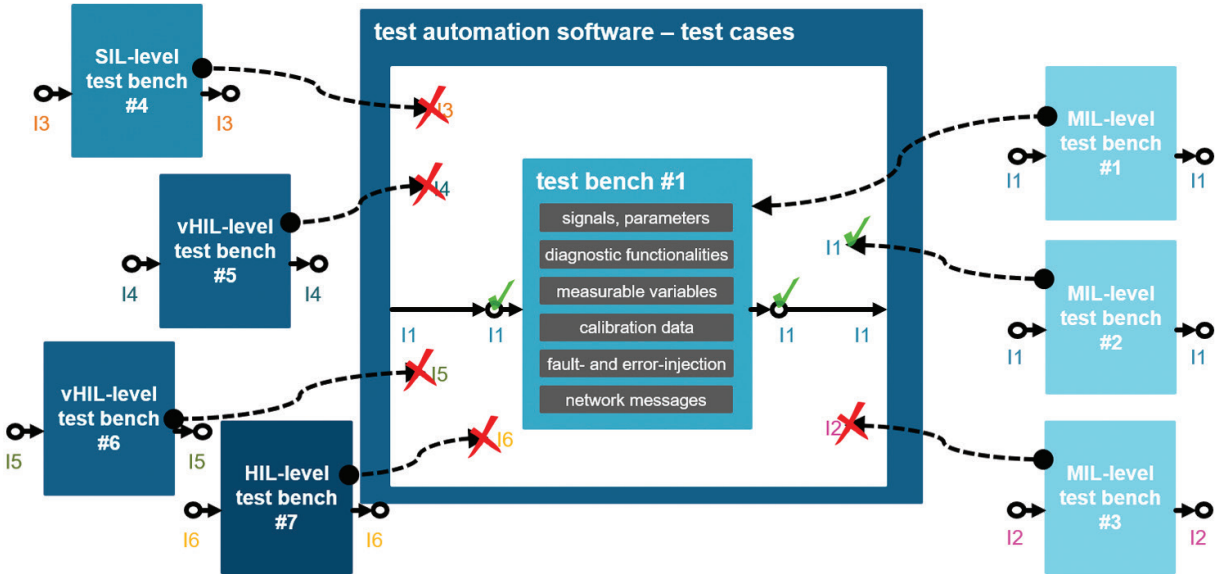

Figure 6: Interfaces across the XIL-levels do not match when data, languages, or tools change.

Standards that support test reuse

Open standards such as ASAM XIL and functional mockup interface have emerged to support testing artifact reuse and provide a common test automation architecture that promotes interoperability between testing tools from different vendors. These standards are very effective at reducing development and training costs.

The standards are more effective when combined with an MDD/XIL workflow methodology that automatically produces testing artifacts as normal by-products of systems integration efforts using institutional rules and model transformations.

When used within the MDD/XIL generative workflow as so-called "tool profiles", architecture standards such as AUTOSAR can further maximize effectiveness and quality. Several professional-grade AUTOSAR runtime software and associated development tool solutions exist from different vendors.

ASAM XIL

The standard most applicable to the topic of this paper is ASAM XIL, an API standard for the communication between test automation tools and test benches. The standard started in 2009, as an effort to standardize on HIL and has since advanced to cover the other test bench configurations used within the MDD/XIL workflow (MIL, SIL, et al). The notation "XIL" indicates the standard can be used for all "In-the-Loop" systems. The most recent release is ASAM XIL v 2.1.0 .

A beneficial contribution of ASAM XIL in terms of business value is the ability to develop tests early in the development process when issues are least expensive to rectify. Other ASAM XIL values include:

- Support for test case reuse by decoupling test automation software from test hardware
- Interoperability between different vendors of test automation software and test benches
- Significant reduction of testing efforts
- Longer term protection of testing investments
- Common way to setup and initialize a test bench
- Collection of time-aligned trace data and a specification of its data format
- Ability to inject faults and errors into the SUT
- Support for test event and triggering

Functional mockup interface (FMI)

Functional mock-up interface (FMI) is an open tool independent standard broadly supported by many tool vendors to support both model exchange and co-simulation of dynamic models using a combination of XML files and compiled C-code. The XML and C-code is bundled into a so-called functional mockup unit (FMU). These FMUs are used during V&V to represent the sensor, actuator and plant environment surrounding a distributed E/E subsystem.

FMI has been driven by an EU ITEA2-funded project, called MODELISAR.

ASAM XIL-MA

The FMI initiative cooperated with the ASAM XIL initiative within the MODELISAR project. This cooperation helped standardize on the way simulations are setup and executed in order to accomplish the coupling of simulation tools with testing tools across the MDD/XIL workflow. The results were embodied into an internally named result called "FMI for applications". ASAM and MODELISAR decided to join this part of their activities in order to develop a single standard, resulting in ASAM XIL-MA.

The major goal of ASAM XIL-MA is to allow test automation tools to control simulation tools and access simulation models executing within them. ASAM XIL-MA is extracted from ASAM XIL and further abstracts test automation tools from test benches specifically for this purpose.

AUTOSAR

The AUTOSAR partnership is an alliance of OEM manufacturers, Tier 1 automotive suppliers, semiconductor manufacturers, software suppliers and tool suppliers. Considering the different automotive E/E architectures in current and future markets, the partnership establishes a de-facto open industry standard for an automotive software architecture.

A contribution of AUTOSAR that facilitates test reuse is the ability to describe architectures and their configurations within automotive E/E systems to provide the required formal model transformation targets within the MDD/XIL generative workflow.

A better test architecture

To combat the inherent testing challenges described herein, a test architecture centered on standards and augmented with automated model transformation within an MDD/XIL generative workflow is key.

Mapping and configuration framework

ASAM XIL is architected into two major parts (figure 7):

1. A test bench for the simulation models, ECU data (e.g. parameters, variables, diagnostics), electrical error simulation and the automotive networks.
2. A mapping and configuration framework that supports the mapping of units, data types or variable identifiers, triggering, actions, measurement, logging and other management facilities and services related to communication to the test bench such as initialization, ordering and sequencing.

The test bench part of the architecture provides standardized interfaces to the different types of tools by means of specific test bench ports. These ports offer standardized access to ECUs, including interfaces for:

- Calibration
- Measurement
- Model access

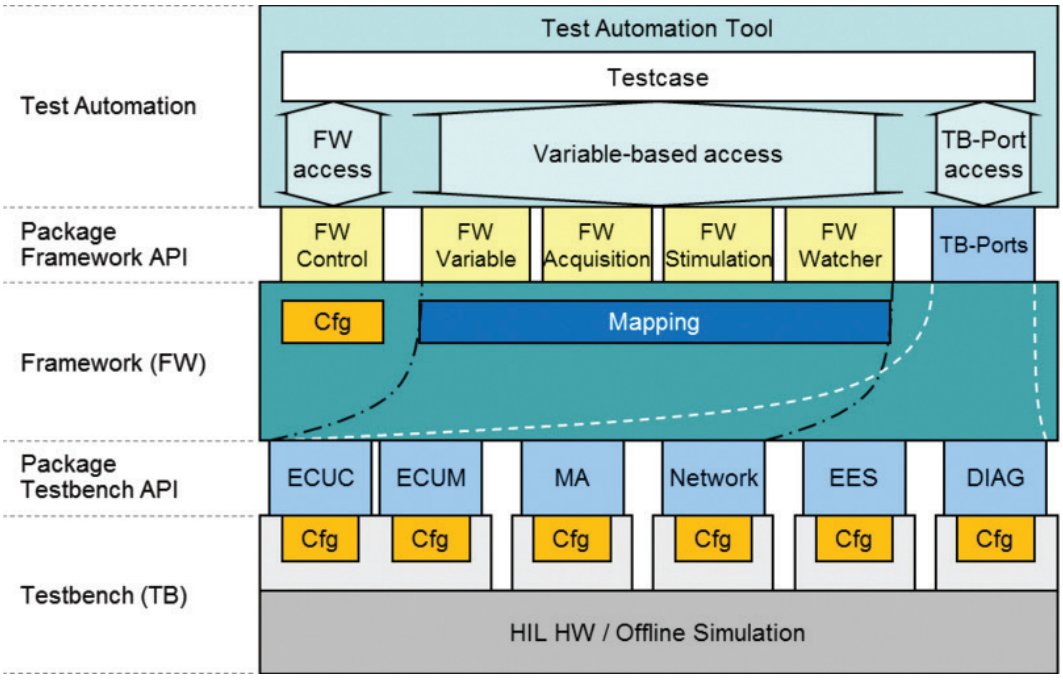

Figure 7: ASAM XIL provides a flexible standard test architecture.

- Diagnostics
- Electric error simulation
- Automotive networks

The most challenging task of decoupling test automation software from test benches is facilitated by the ASAM XIL mapping framework. It allows data within a test bench to differ in value and type during the overall MDD/XIL workflow. The framework isolates the differences to different mappings of values between the test cases and the test benches. Examples include abstract identifiers to concrete identifiers, physical units to physical units and data types to data types. By providing a new mapping for each different test bench, the test case remains unchanged and can be reused directly across engineering phases.

It should be noted that the test bench ports are supported in ASAM XIL 2.0 only for compatibility with ASAM HIL and to provide functionality that is currently not provided by the framework layer. The major change in this area of ASAM XIL 2.0 is to introduce port independence to test cases by using an object-oriented access to variables on the test bench and an abstraction of ports within the framework layer. Based on these variable objects, the framework also provides objects for port-independent signal recording, signal generation and event watching and triggering.

ASAM XIL thus effectively provides an adaption layer between the test cases and the different XIL test bench configurations within the MDD/XIL workflow. This allows the verification engineer to test any MIL, SIL, vHIL, or HIL test bench configuration using the same test case by inserting its corresponding mapping and configuration counterpart between the test case and the test bench (figure 8).

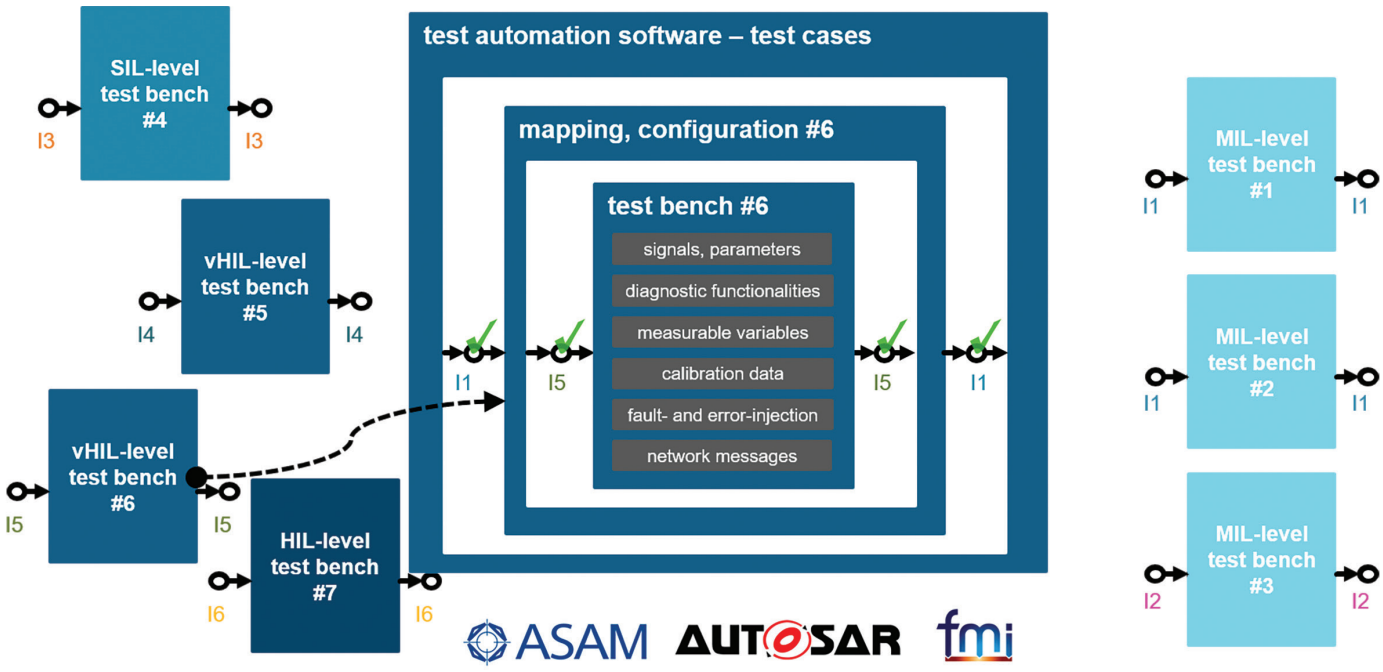

Figure 8: Test cases are reusable across MDD/XIL configurations using ASAM XIL mapping and configuration layers to adapt test bench interfaces.

Generative MDD/XIL workflow

ASAM XIL, FMI and AUTOSAR combine to provide all of the formal ingredients required for automated model transformation to create a generative MDD/XIL workflow.

With the addition of a set of institutional rules, it is possible for automated tooling to take test case descriptions, function models, environment models and architecture models that are produced during normal system design efforts, and generate the test benches, mappings and configurations required for automated V&V regression testing.

The workflow starts at the MIL-level of abstraction and generates test artifacts (figure 9). It is important to note that all of the models conform to some formal domain-specific meta-model, and the map and configuration descriptions correspond to the function and environment model test benches for MIL-level testing.

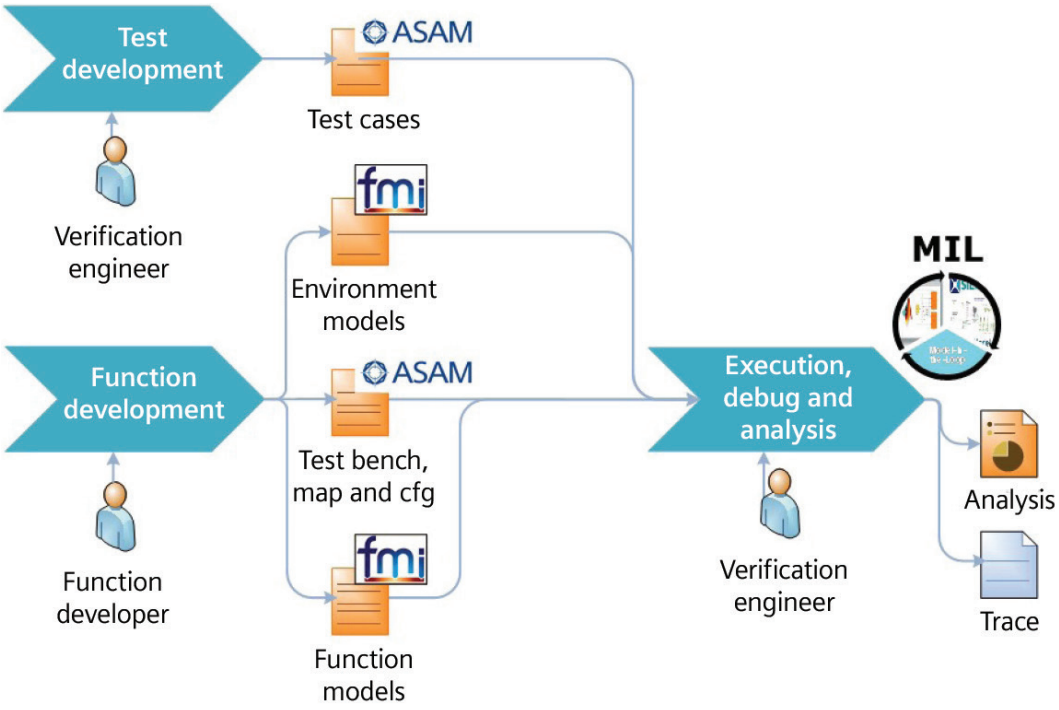

Figure 9: The MDD/XIL MIL-level workflow using ASAM and FMI standards is straight-forward.

After the function model is validated at the MIL-level, it is used as input for automatic integration into an AUTOSAR-based platform using another set of rules (figure 10). Note again that all of the models conform to some formal meta-model. The additional map and configuration descriptions correspond to the ECU test bench. The test bench, map and configurations for the environment model are reused directly from the MIL-level test bench. Notice also FMI co-simulation is used to connect and execute the environment model and ECU simulation (for the case of vHIL).

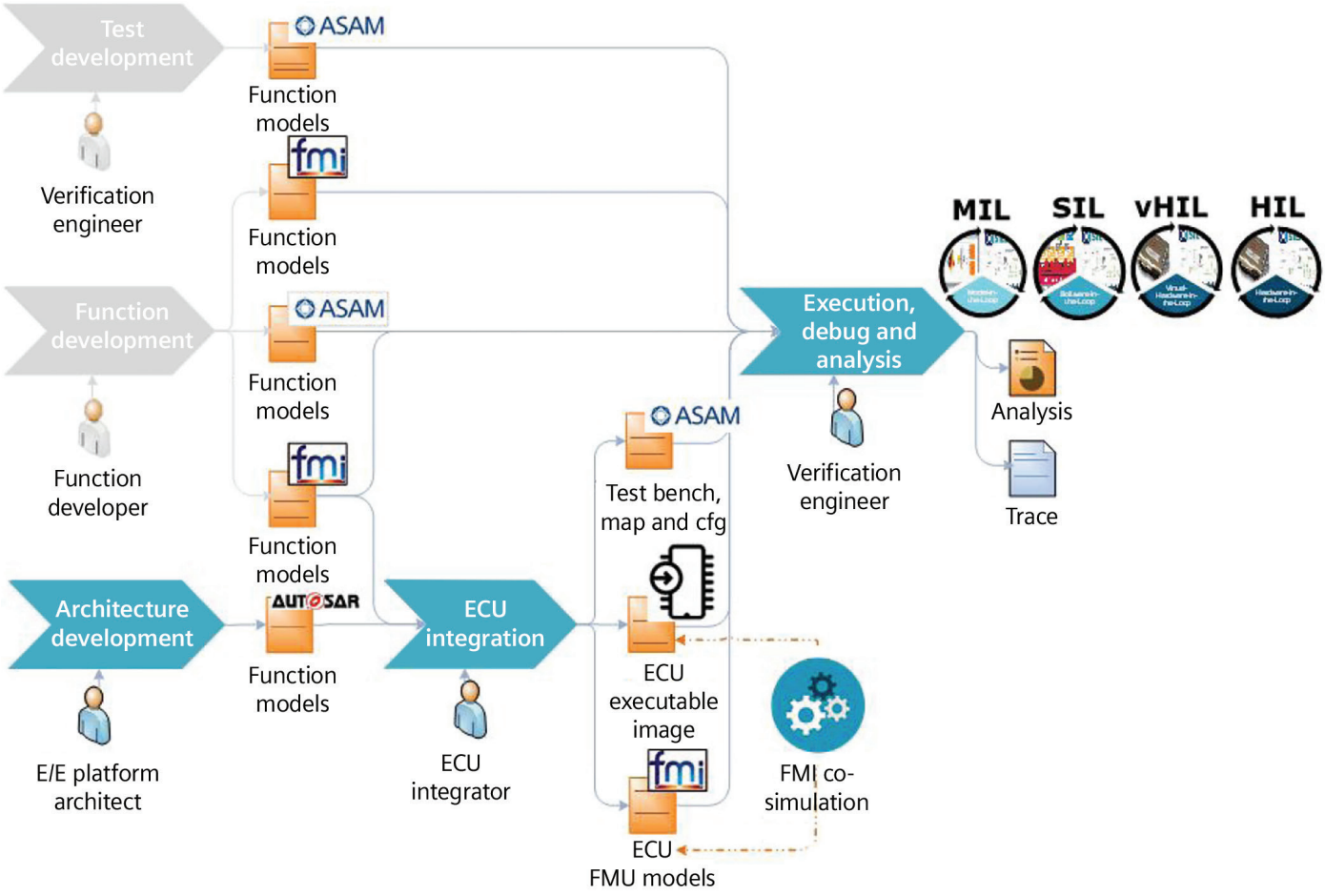

Figure 10: The MDD/XIL workflow reuses tests across XIL-levels using formal standards.

Results and conclusion

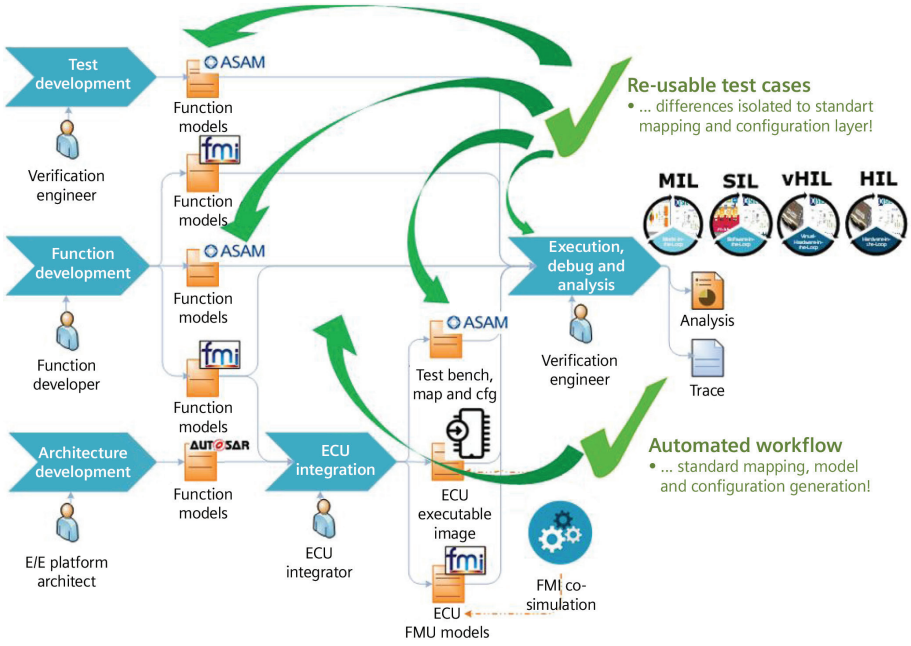

Figure 11: A MDD/XIL workflow that uses a standard test architecture supports test case reuse.

The paper presented a standards-based, systematic and automated generative MDD/XIL workflow that helps automotive developers develop their production ECU V&V suites early during software modeling, and reuse them throughout the overall systems engineering project. The test cases developed during design can be reused through to production ECU testing and ultimately for automated regression V&V of future software updates deployed into the field (figure 11).

There are several valuable business benefits this testing architecture and generative MDD/XIL workflow affords:

- Problems are found earlier in car projects when they are least expensive to fix
- Test cases are reused across the entire MDD/XIL workflow, preserving testing investments
- Increased V&V coverage increases safety, enhances security and otherwise finds problems before they are deployed into the field
- An automated and generative MDD/XIL workflow increases quality by using model transformations that are correct-by-design, and it also significantly reduces development which shortens time to market
- OEMs and suppliers can efficiently exchange test cases and test benches which saves both cost and time
- Decreased training costs because standards-based know-how is more easily transferred
- Verification engineers can combine the best test automation software with the best test bench products because of standard interoperability – and switching between testing solutions is much less expensive
- Tests can include multiple levels of abstraction in a heterogeneous configuration – this is particularly useful when validating the distributed multi-core, multi-ECU E/E systems found in modern ADAS, ADS and AV vehicle systems
- Multiple, cross-discipline interface requirements are satisfied with standardized access: calibration, measurement, diagnostics, networking, tracing, stimulus and fault-injection

Future work

The following sections contain short discussions of areas for future work and consideration.

ACOSAR

The main goal of ACOSAR is to develop a non-proprietary real-time (RT) co-simulation interface. ACOSAR is an ongoing project. Real-time testing, in particular mixing actual and virtual testing components together, is an important use case (co-simulation of RT and non-RT systems). Because of the inherent need for network protocols in such a configuration, ACOSAR could potentially fill the gap of the missing network protocol specifications within FMI. It is considered that ACOSAR will provide a so-called "FMI for RT co-simulation".

System structure and parameterization

System structure and parameterization (SSP) is a developing standard that should be usable in all stages of the MDD/XIL workflow. The standard's effort is coordinated with the FMI and ASAM standards. The goal of SSP is to provide a standardized format to represent a connected set of components.

These components for example, can be FMI-based simulation models and ASAM testing artifacts. This would provide for a more complete automation and a standard way to describe and parameterize test configurations if the MDD/XIL workflow utilizes SSP as a standard description format.

For more information on automotive software and networking system solutions, please visit this website.

Siemens Digital Industries Software

Headquarters
Granite Park One
5800 Granite Parkway
Suite 600
Plano, TX 75024
USA
+1 972 987 3000

Americas
Granite Park One
5800 Granite Parkway
Suite 600
Plano, TX 75024
USA
+1 314 264 8499

Europe
Stephenson House
Sir William Siemens Square
Frimley, Camberley
Surrey, GU16 8QD
+44 (0) 1276 413200

Asia-Pacific
Unit 901-902, 9/F
Tower B, Manulife Financial Centre
223-231 Wai Yip Street, Kwun Tong
Kowloon, Hong Kong
+852 2230 3333

About Siemens Digital Industries Software

Siemens Digital Industries Software is driving transformation to enable a digital enterprise where engineering, manufacturing and electronics design meet tomorrow. The Xcelerator portfolio helps companies of all sizes create and leverage digital twins that provide organizations with new insights, opportunities and levels of automation to drive innovation. For more information on Siemens Digital Industries Software products and services, visit www.sw.siemens.com or follow us on LinkedIn, Twitter, Facebook and Instagram. Siemens Digital Industries Software – Where today meets tomorrow.

About the author

Lance Brooks is part of the Siemens embedded automotive product team. Lance has over thirty years of experience creating embedded systems and related development tools with over fifteen years dedicated to automotive software development, with a specific focus on hardware/software modeling and simulation.

siemens.com/software

81432-C4 6/20 N