R&A CONTROLS ENGINEERING

VEHICLE CONTROLS & SYSTEMS ENGINEERING

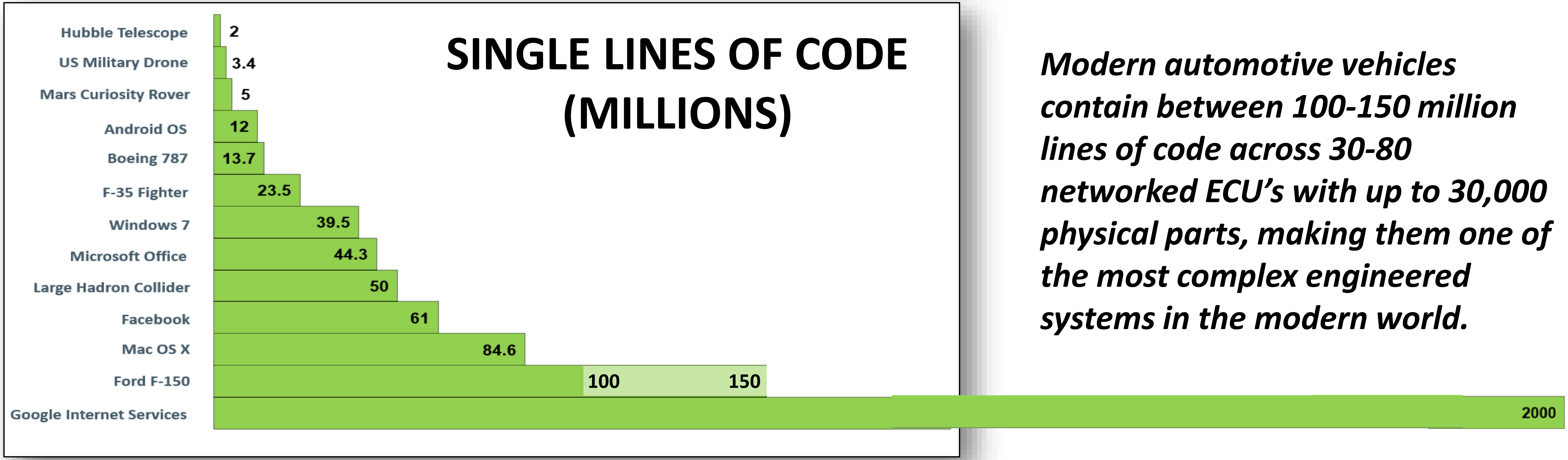
Virtual Simulation using QTronic Silver and TestWeaver

Presented by:

Robert Ter waarbeek

Ron Boudia	Claire Chen	Kaushik Kannan	Kevin Ruybal
Tim Cardanha	Jeffrey Doering	Patrick Kenny	Gary Song
Peggy Caveney	Jun Gou	Jason Meyer	Nate Rolfes

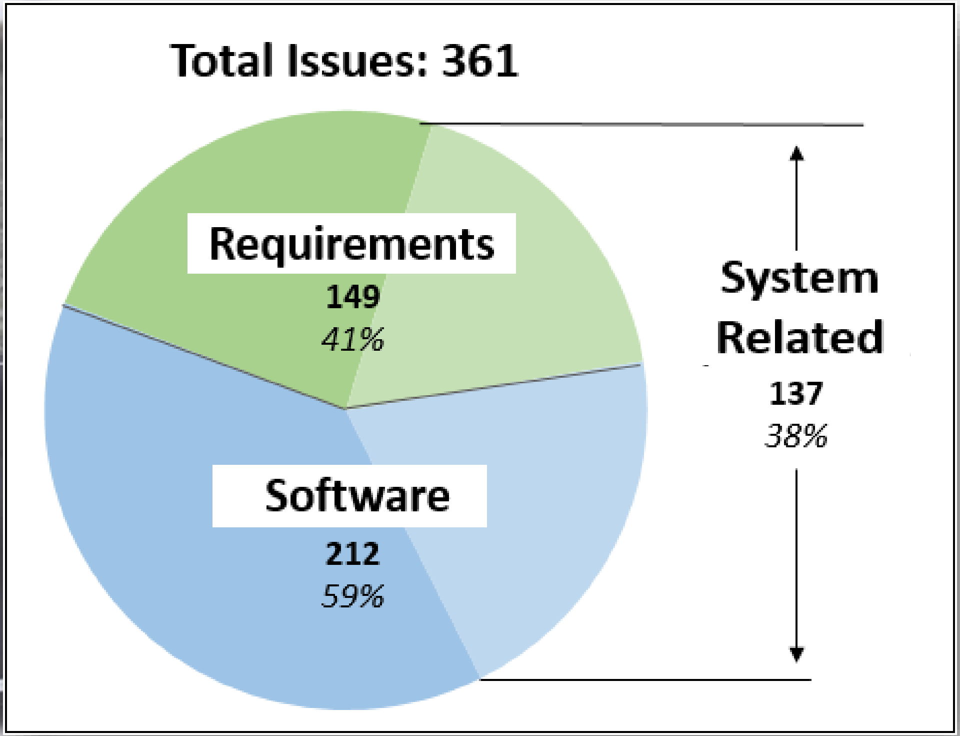
MODERN SOFTWARE COMPLEXITY



Modern automotive vehicles contain between 100-150 million lines of code across 30-80 networked ECU's with up to 30,000 physical parts, making them one of the most complex engineered systems in the modern world.

Sources:
https://informationisbeautiful.net/visualizations/million-lines-of-code/
https://www.mckinsey.com/industries/automotive-and-assembly/our-insights/rethinking-car-software-and-electronics-architecture
http://sites.ieee.org/futuredirections/2016/01/13/guess-what-requires-150-million-lines-of-code/

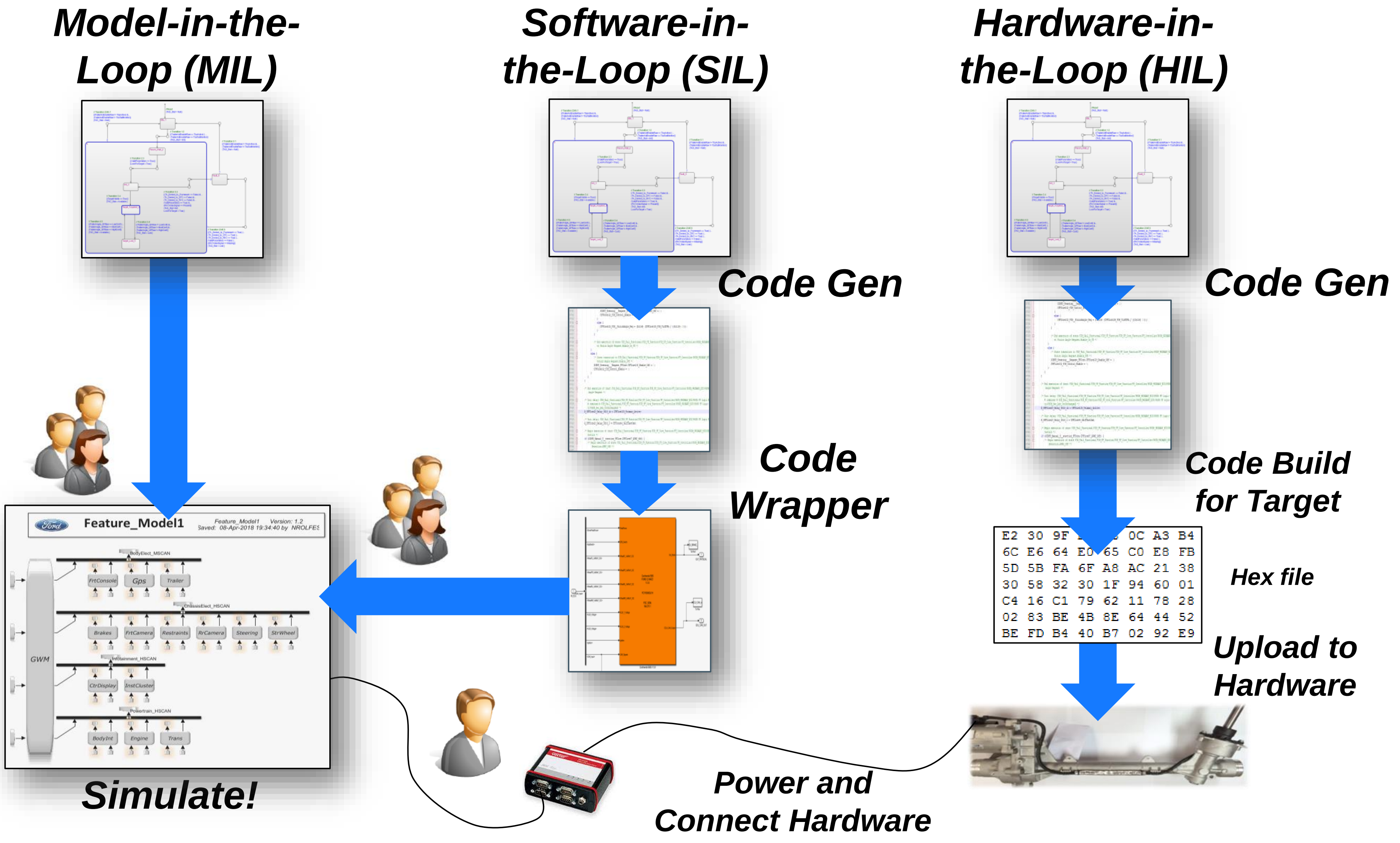
THE SOURCE OF SOFTWARE ISSUES



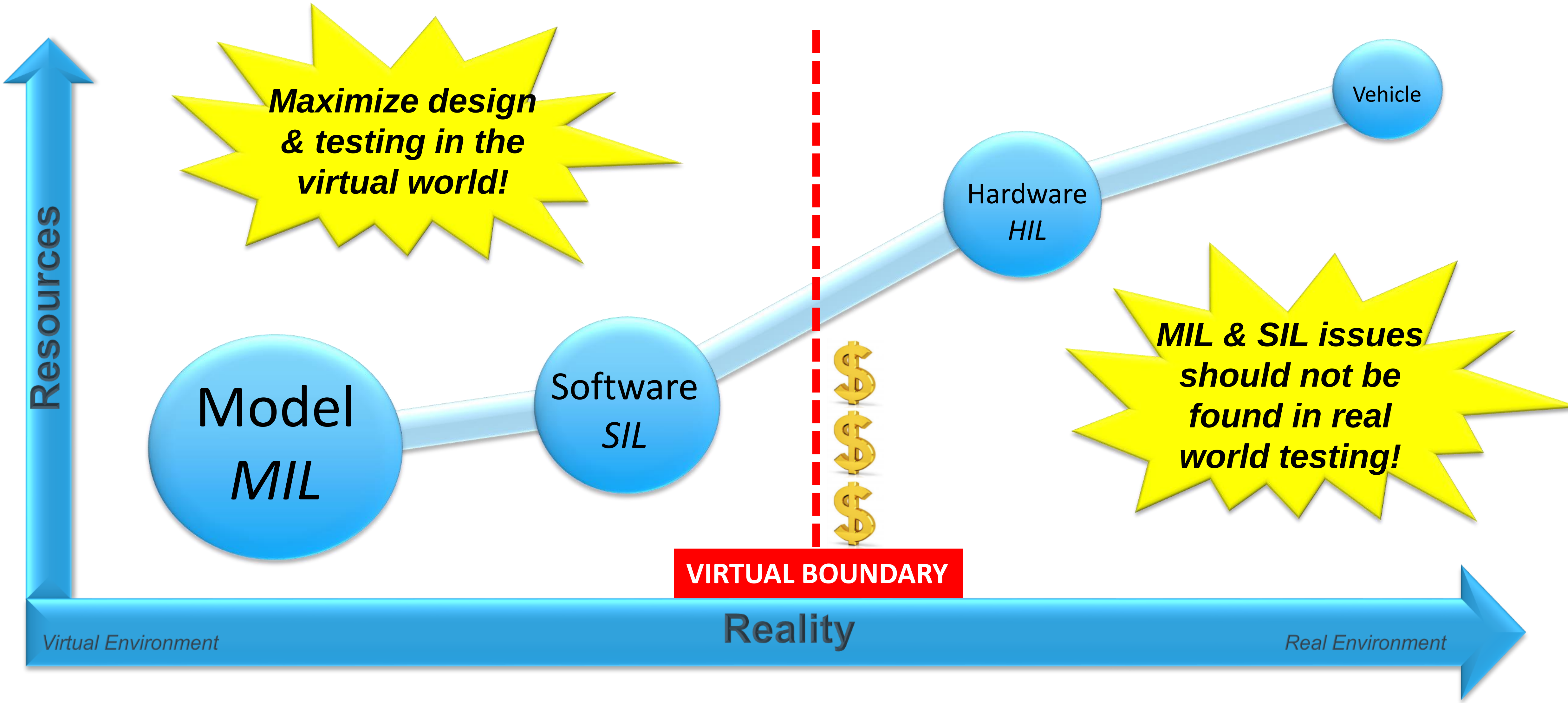
41% of Software issues found during development of the 2016MY F-150 Pro Trailer Backup Assist Feature were related to the requirements, and 38% of all software issues were system-related.

Source:
Rolfes, N., "Requirement Modeling of Pro Trailer Backup Assist™," SAE Int. J. Passeng. Cars – Electron. Electr. Syst. 10(1):2017, doi:10.4271/2017-01-0002.

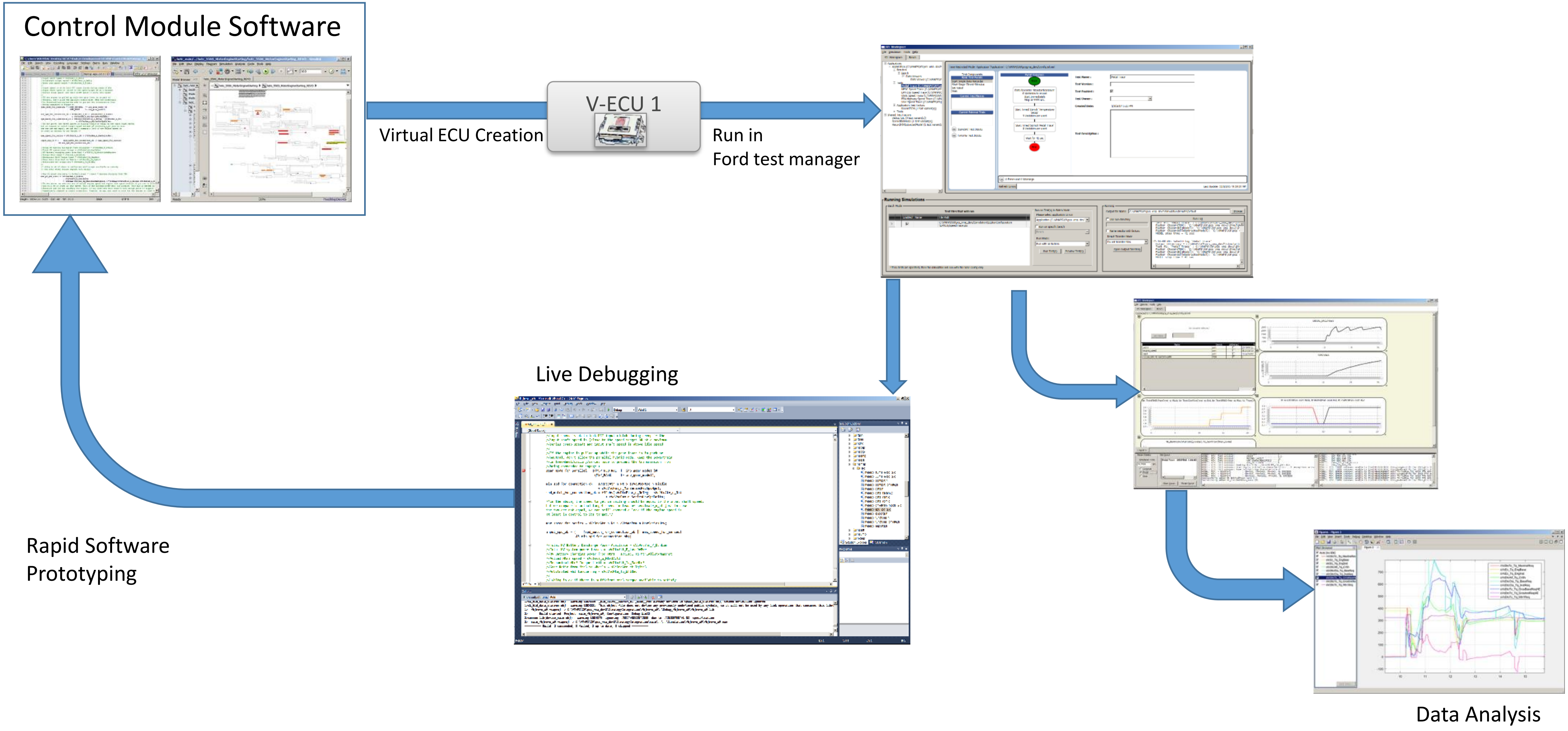
DISTRIBUTED SYSTEM MODEL TYPES



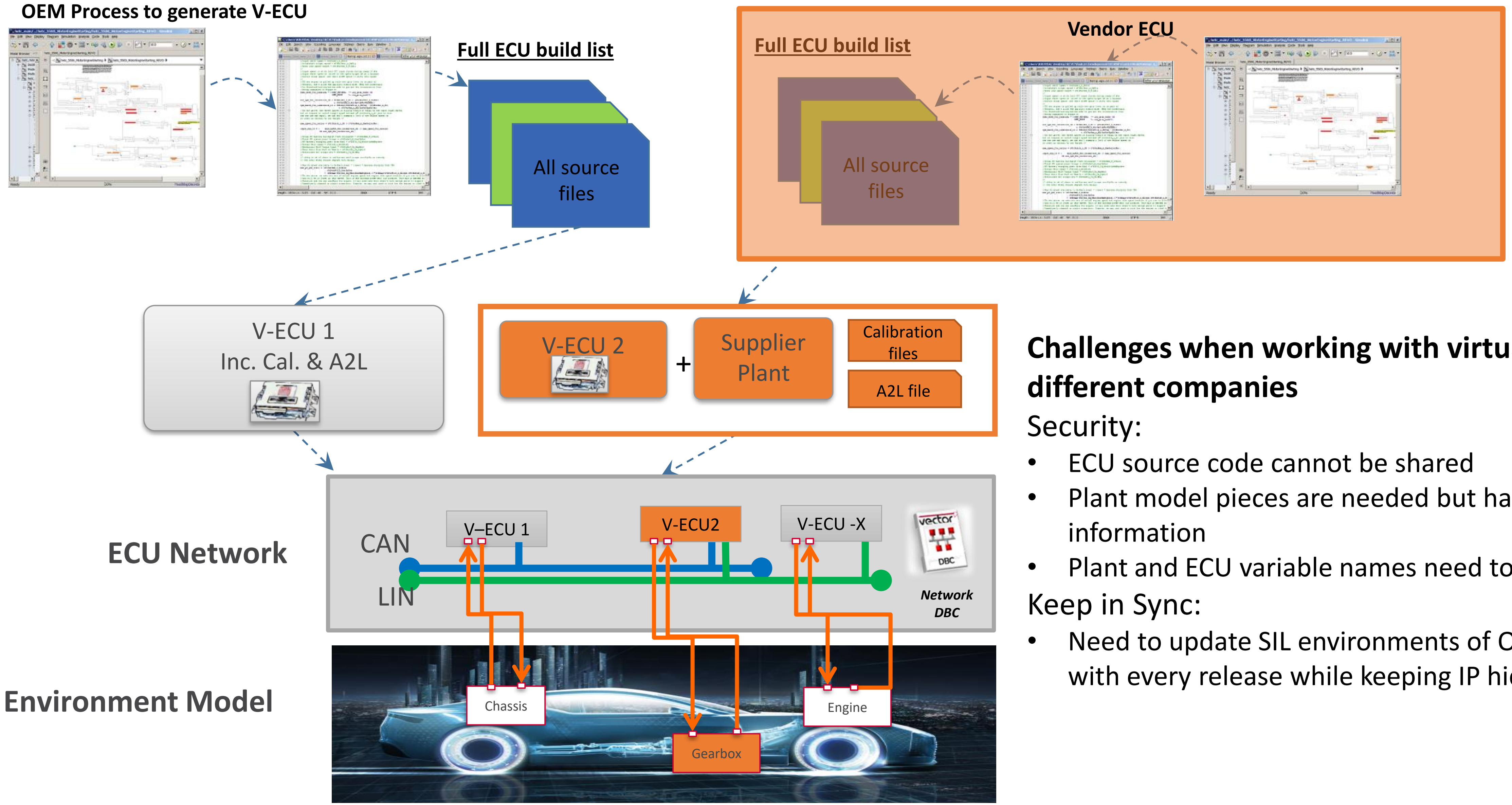
DISTRIBUTED SYSTEM MODEL TESTING



Software in the Loop (SIL) process



General Simulation environment with multiple ECUs



Challenges when working with virtual ECUs from different companies

Security:

- ECU source code cannot be shared
- Plant model pieces are needed but have proprietary information
- Plant and ECU variable names need to be hidden

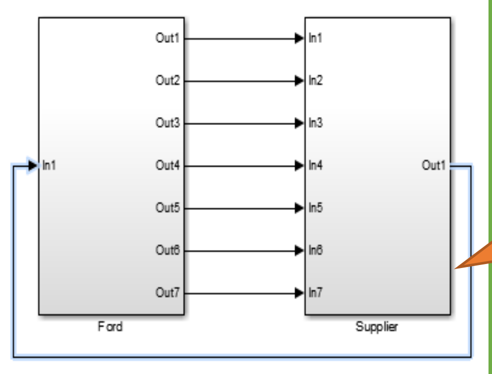
Keep in Sync:

- Need to update SIL environments of OEM and supplier with every release while keeping IP hidden

Ford SIL environment with Supplier Silver ECU

Ford Test Manager

Ford Simulink Plant



Protected plant model delivered by Supplier replaced simplified Ford internal part

Ford Simulink Controls

Ford -ECU 1

Ford -Wrapper

Supplier V-ECU

Silver DLL

* Requires Silver Runtime license

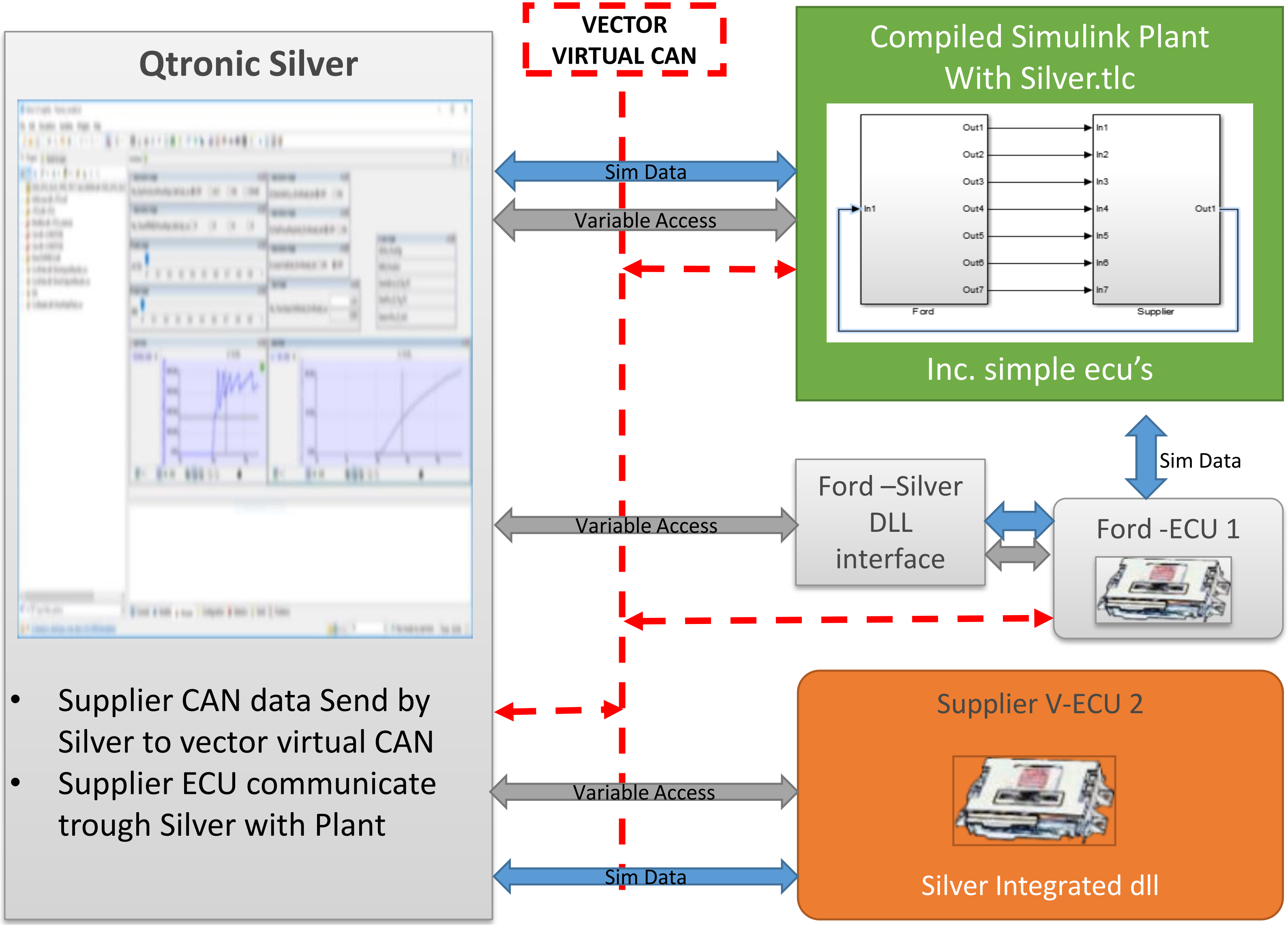
Ford Setup

- Simulink plant model Includes Simulink protected models from supplier

- Ford SIL V-ECU with all source code available
- Ford V-ECU A2L file with full variable access

- Supplier V-ECU built with Silver with (source code protected)
- Supplier V-ECU A2L access with limited variables

Joint Environment using Silver front end and V-ECU for Ford and Supplier



Supplier & exchange environment

- Ford plant model built using Silver.tlc
  - limited model variables exposed
  - inc protected supplier plant model
- Ford V-ECU without exposing source code.
  - Runs in sync with plant model
  - Ford ECU A2L file with limited variable access
- Supplier SIL integrated ECU built with Silver
  - Supplier:
    - all source code available
    - Full A2L file with full variable access
  - OEM:
    - No Source exposed
    - Reduced A2L File

Silver enabled the joint development between supplier and OEM model and enables flexible setups

Silver Experience

Features

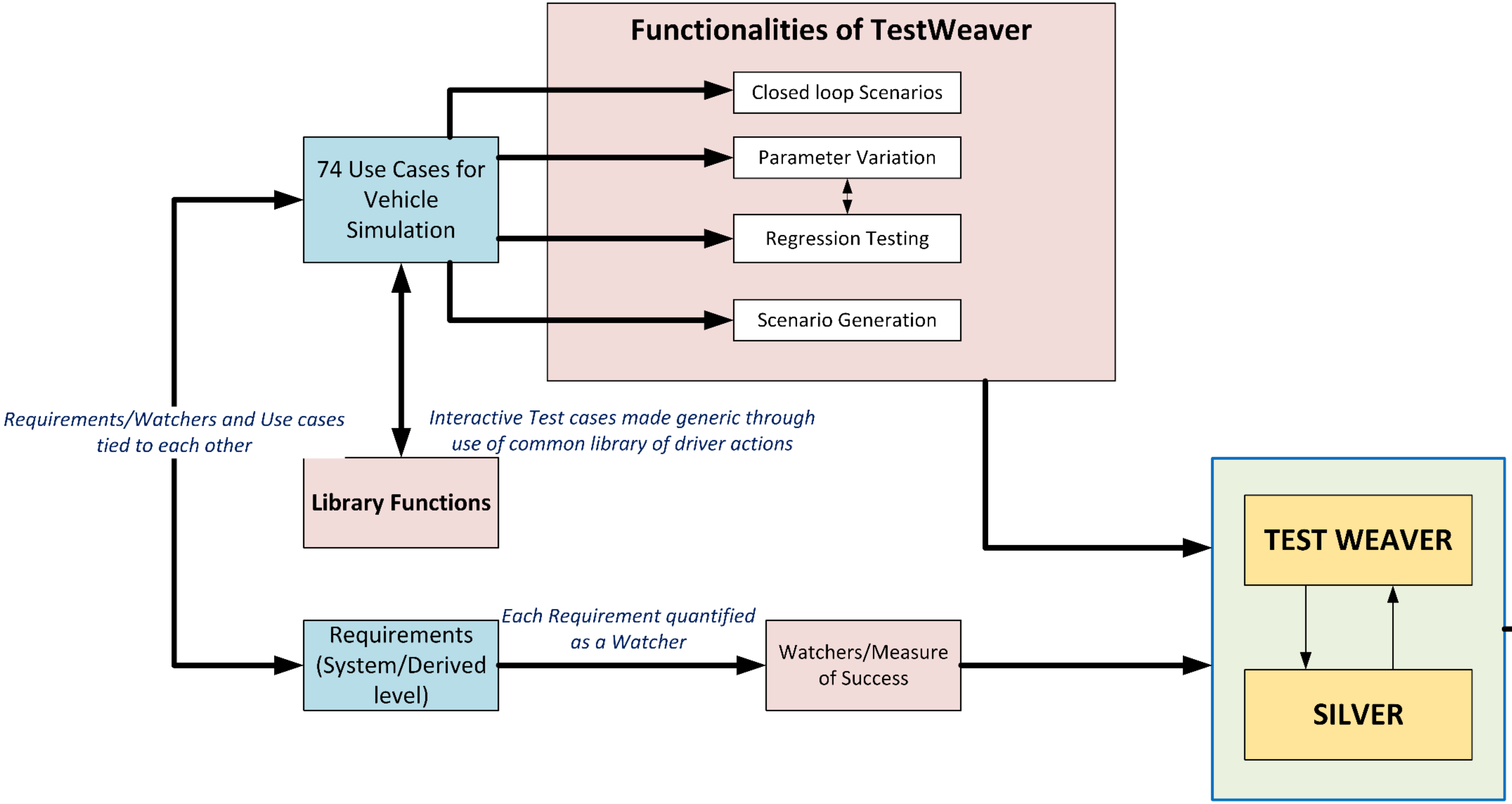
- Intuitive GUI which our users found easy to use with minimum training
- Command line supports automation which enables continuous integration
- Customizing, like making use of an additional silver_user_code.tlc file, was easy to understand and allowed us to successfully integrate our simulation pieces
- Interface with the module DLL's is very clear and allowed us to know the exact IO for DLL's from the supplier
- Found different built-in modules very useful, like Virtual CAN, A2L Access, CSV readers and writers.
- Support of Python scripting

Engineering Support

- We all had a great experience with QTronic support staff
- Everyone was very prompt on answering questions
- Received custom examples of different reporting options

Both environments are quick to setup and kept in sync easy using QTronic Silver

Qtronic Test Waver use case



Key Outcomes:

- Intelligent Automated Testing
- Automated report generation
- Pin point exact location of failures
- Status of Watchers
- MDF 3.3 format for ATI vision data analysis
- Read MDF files as inputs for data analysis
- Read/write CSV files for scenario data

Library Functions and Watchers using Test Weaver

LIBRARY FUNCTIONS

MOTIVATION:

- To make the test case as **generic** as possible
- Create Library functions of test actions that can be **re-used** across multiple applications
- Easily **sharable** and reproducible across multiple users

Examples of Library Functions:

- Power Up
- Accel Pedal Tip in Y% from Stop
- Drive to X mph from stop
- Accel Pedal Tip in Y% from X mph
- Accel Pedal Tip out to X mph
- Brake to X mph
- Power Down

OUTCOME:

- Library functions were generic and could be **re-used across several vehicle programs**
- **74 Use cases** were scripted within a very short span of time with the use of library functions
- **Shared and reproduced** by multiple test engineers

WATCHERS

WHAT ARE WATCHERS IN TESTWEAVER?

- A formulated representation of the Requirement
- An assessment with several output values including SUCCESS and FAILED

Watcher States	Value
WAITING_CONDITION	1
WAITING_EVENT	2
WAITING_DELAY	3
TOLERANCE_TIME	4
PASS_DURATION	5
SUCCESS	6
FAILED_FAILCONDITION	7
FAILED	8

- Is evaluated at specific task rate in Co-simulation mode

HOW WERE THEY USEFUL?

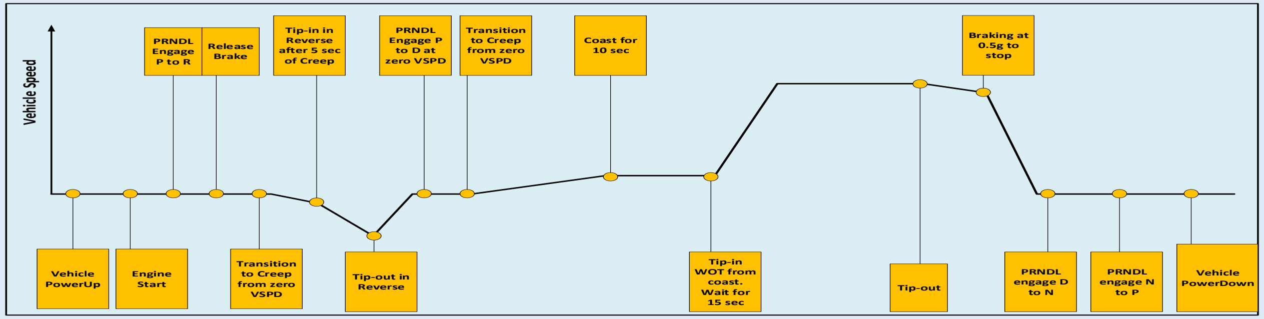
- Requirements were translated into **60 watchers**
- State of each watcher and related signals **exported to ATI** for further analysis
- Pin point the exact time and **location of failure**
- Enabled **faster debug process** and accelerated software development
- These watchers could potentially be **re-used** across several vehicle programs with minor modifications

Scenario Generation and Parameter Variation

Scenario Generation

MOTIVATION:

- ❑ **Bugs in Requirement and Software implementation leads to unintended behavior that is difficult to spot.**
- ❑ **Extend to special operation states which are not allowed or not safe to reach in vehicle.**
- ❑ Potential verification for functional safety requirements.
- ❑ There is a need to use automated testing for smart classification and generation of scenarios
- ❑ Test the vehicle behavior in a huge number of situations that are relevant
- ❑ Push the system into states and assess system behavior



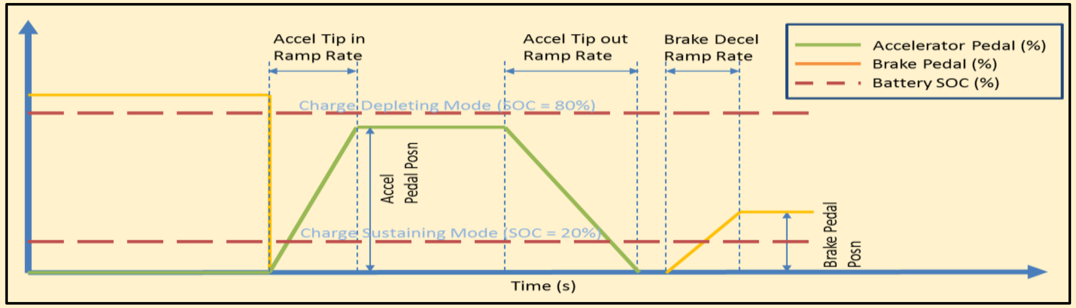
OUTCOME:

- ❑ Reactive scenario generation (Each scenario depends on history of generated scenarios)
- ❑ All cases can be reproduced
- ❑ Drive the system into states that have not reached before
- ❑ Run worst case scenarios and safety critical function evaluation
- ❑ Evaluation of Noise injection techniques with automated testing
- ❑ Detect violations in requirements

PARAMETER VARIATION

MOTIVATION:

- ❑ Use cases where two or more parameters need to be varied
- ❑ Manual scripting of each test case is time consuming and inefficient
- ❑ Programmable DOE scenario generation and testing.



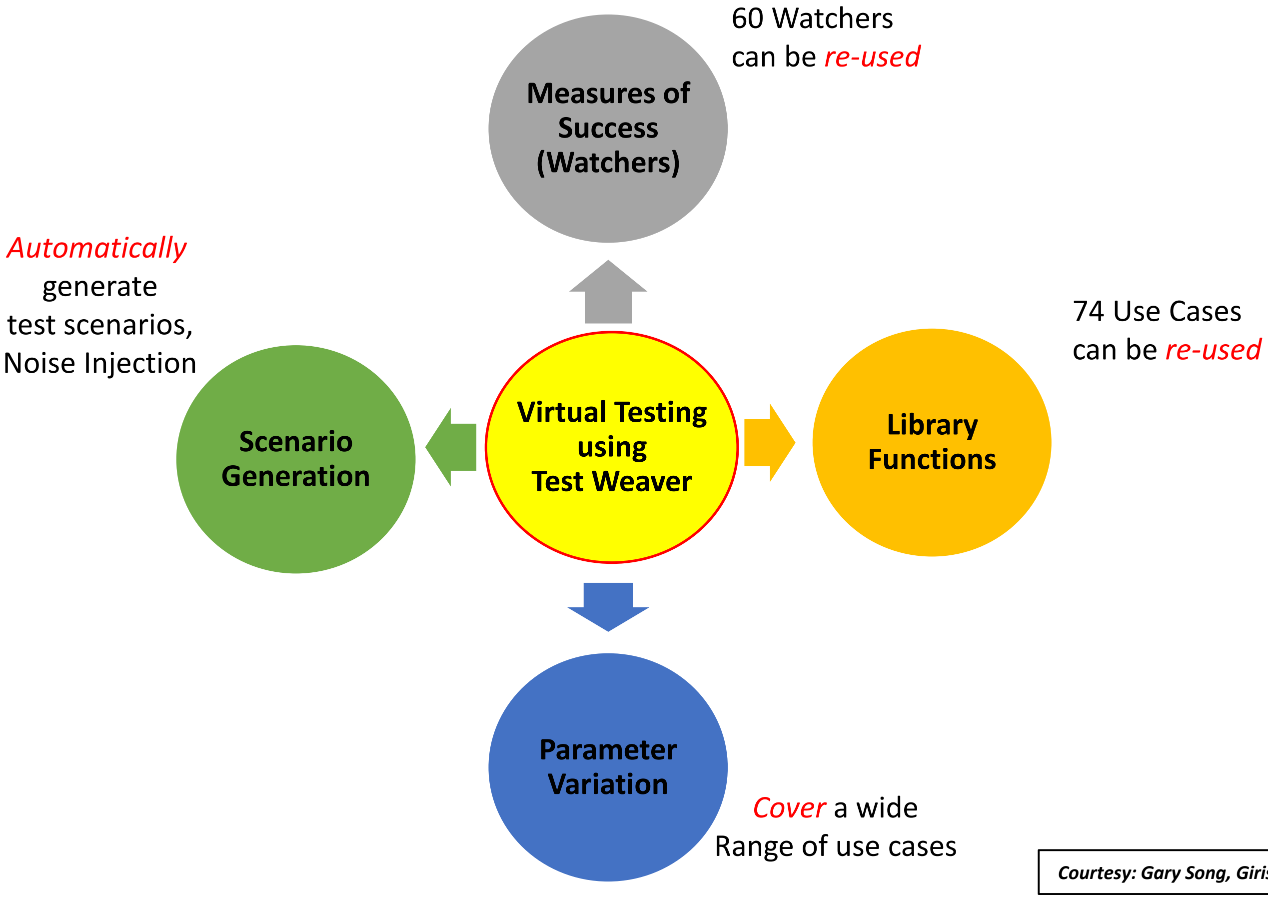
Parameter	Values	No. of Values
Accel Pedal Posn (%)	[10, 20, 30, 45, 60, 75, 100]	7
Brake Pedal Posn (g)	[0.1, 0.18, 0.4, 0.8]	4
Accel Tip in/Tip out Ramp rate (%/sec)	[15, 30, 180]	3
Braking rate (g/s)	[0.05, 0.1, 0.2, 0.4, 0.8]	5
Initial SOC (%)	[80, 20]	2

Total Number of Combinations is 840 !!

OUTCOME:

- ❑ Run all combinations of parameters and automatically generate test cases
- ❑ Run 600 test cases overnight (Over a span of 12 hours)

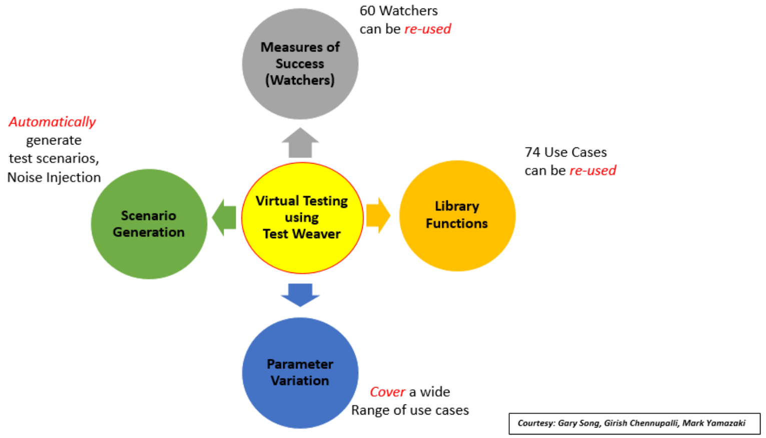
Virtual Testing using TestWeaver – An Overview



Courtesy: Gary Song, Girish Chennupalli, Mark Yamazaki

What we like the most of QTronic Silver/TestWeaver(light)/TW(Full)

1. **Powerful test assessment capability with thoughtful Measure-of-Successes (Watchers)**
   - States of Measure-of-Successes
   - "Loop-by-loop" assessment/Co-simulation
   - Clearly indication of failure location
   - Assessment results are part of test results and saved in MDF which can be viewed from ATI VISION
   - RML expands the assessment capability for chained events or behaviors.



2. **Variety of test case specification methods**
   - Python scripts and interactive sophisticated test cases in co-simulation style
   - CSV/Pre-recorded MDF
   - Python and CSV test cases can be version controlled friendly
3. **Flexible instrumentation capability for System Under Test so customers can do extensive testing.**
4. **Simulink target in TestWaver provides the capability for software component testing with the same assessment capability.**
5. **Powerful regression capability and assessed by Measure-of-Success**
6. **Powerful auto test generation with design focus (Unique to TW full)**
7. **Comprehensive** reporting capability (test database approach)


Ford
THANK YOU
181 HMA