MathWorks
AUTOMOTIVE CONFERENCE 2023
Korea

Development of Fault Detection and Reaction Software Using Model-Based Development

Song Eun Jae, Hyundai Mobis

MathWorks®

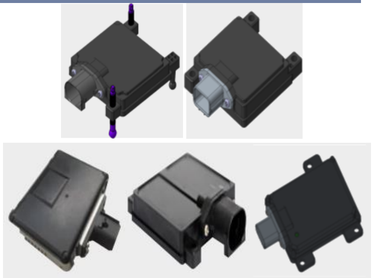
Table of Contents

- Hyundai Mobis Business Area and ADAS/AD Division
- Key Features of the Fail-Safe Software
- How Fail-Safe software is developed in Mobis
- Case in developing the Fail-Safe software: from the skeleton model to embedded code
- Case in improving work efficiency
- Concluding : Why the model-based development is applied for the Fail-Safe software
- Further Works

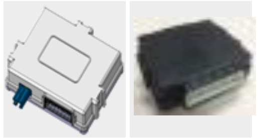
Hyundai Mobis Business Area

Core Automotive Technologies

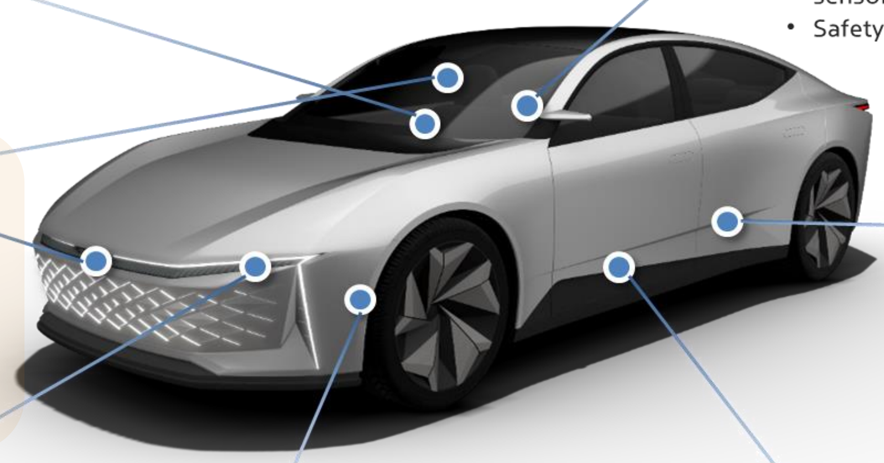
Sensor



Controller



HYUNDAI MOBIS



In-Vehicle-Infotainment

- Multimedia (Headunit)
- Head-Up Display
- Sound system
- Center Stack Display
- Switch Controller
- Telematics Unit

Safety

- Airbag Module : Front/Side/Roof Airbag
- Airbag Control Unit (incl. Remote impact sensor)
- Safety Control Module (using video data)

ADAS

- Auto Parking System (ECU, Sensor)
- Surround View Monitoring System
- Radar (Front / Corner)
- In-Cabin Sensing
  - Driver Monitoring System
  - Occupant Monitoring System

Module

- Chassis Module(Axle Assembly)
- Cockpit Module
- Front End Module

Lighting

- ADB Headlamp
- HD Headlamp
- Homogeneous Rear Lamp
- Grille Lamp

Chassis

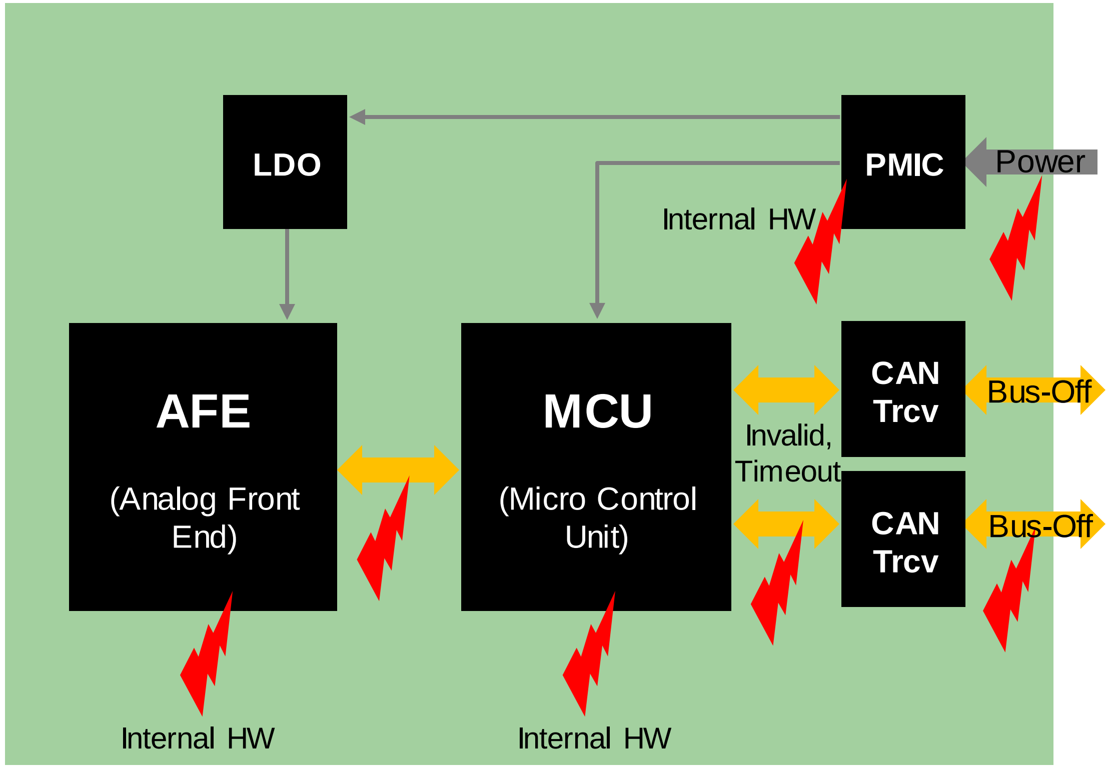
- Conventional Brake : Caliper/Drum Brake/EPB
- Slip-control, e-Boost : One box, Two box
- Steering EPS : C/DP/R type with full redundancy
- Air Suspension

E-Powertrain

- Traction Motor / EDU / Starter & Generator
- OBC / DC-DC Converter / Inverter / ICCU
- BMS & Battery System
- 48V System
- In-Wheel Motor

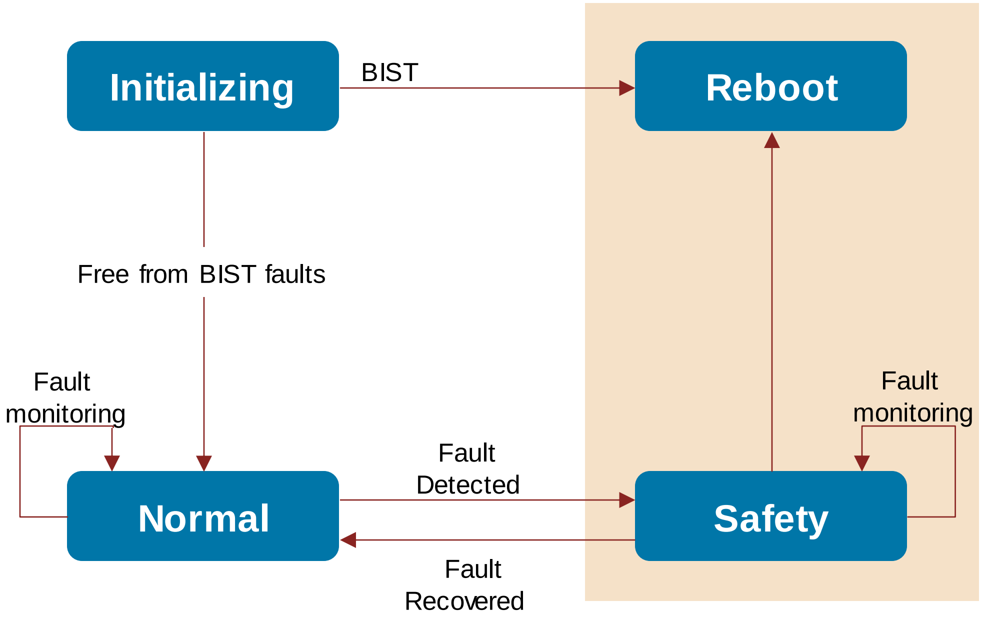
How the Fail-Safe Software ensures the system safety (1)

- It detects faults that could cause system malfunctions,
- It executes safety mechanisms to maintain the system within safe states despite the faults occurred,
  - Faults may come up anywhere: Supply Power, Bus-Off, Message Timeout, Invalid Signal, etc.
- It stores diagnostic trouble codes (DTC) in Non-Volatile Memory to facilitate self-diagnosis for fault identification and cause analysis.



How the Fail-Safe Software ensures the system safety (2)

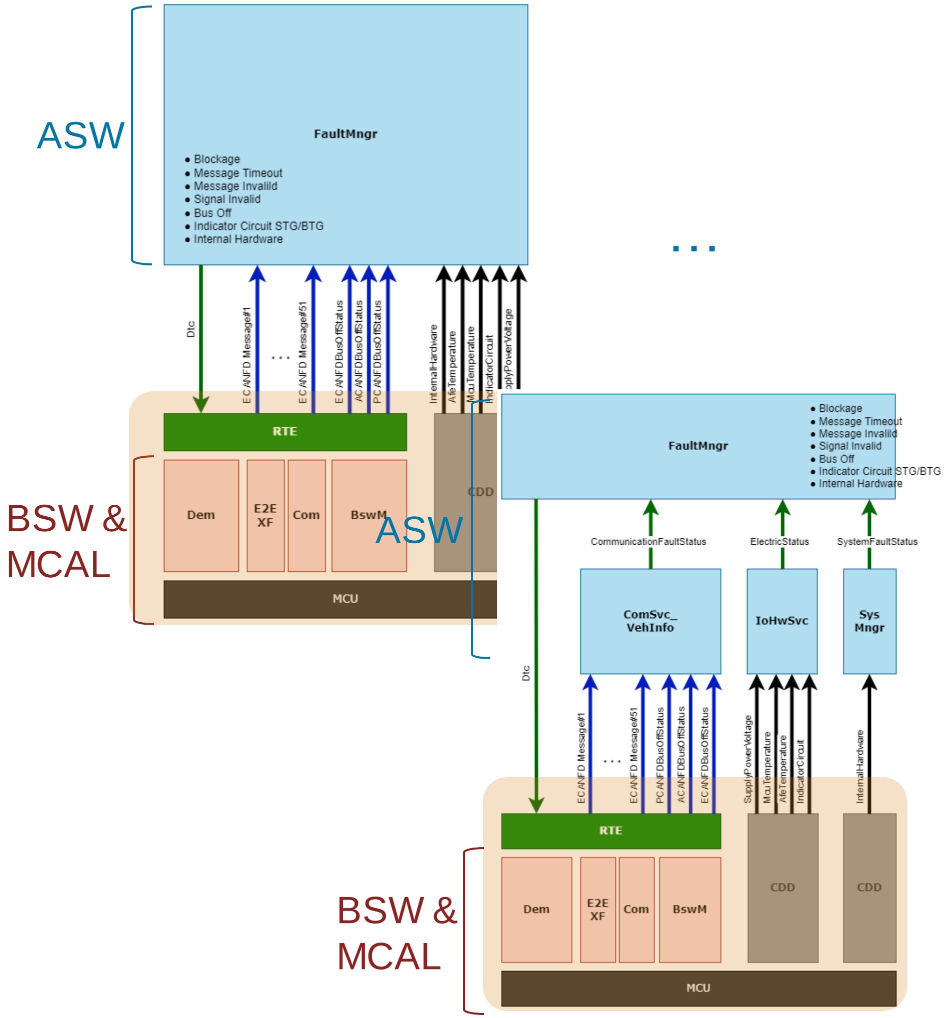
- The Fail-Safe Software transitions to Safety state and notifies other systems or driver of the status of fault status,
- It ensures that the system operates with minimal performance to prevent accidents.



What the key features of the Fail-Safe Software are

- To detect system faults (error) and react about the faults (error), uses standardized AUTOSAR interfaces,
  - BswM, Dem, E2EXf, ComXf, Complex Device driver (CDD) …
- The platform software related to the Fail-Safe notifies fault status to the application software periodically.
  - CRC error, Alive Counter error, Timeout error, Bus Off error, Out of range …
- The Fail-Safe application software qualifies the system faults and determines whether to transition the system operating mode to safe state or another state.

Software Architecture Scenario #1

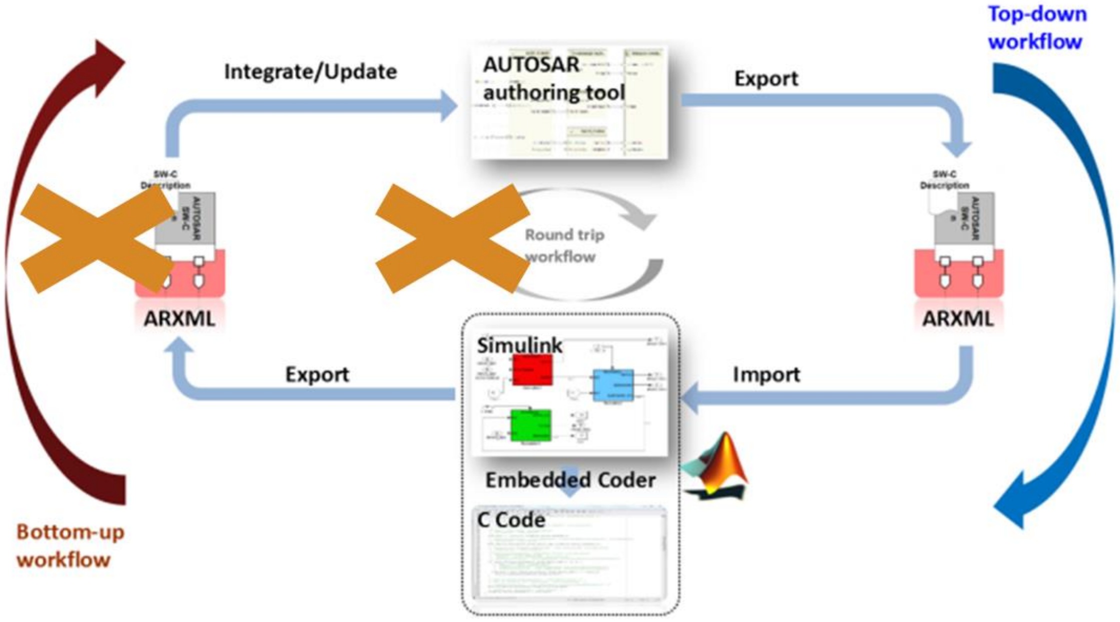


Software Architecture Scenario #n

How Mobis develops the Fail-Safe software (1)

Why not Bottom-up nor Round-Trip

- The Fail-Safe Software has high dependency on the embedded system,
  - It cannot be dependent from the hardware architecture
  - It employs AUTOSAR services in its design
- Rapid-prototyped Simulink models are not necessary,
- Simulink including AUTOSAR Toolbox is not yet 100% AUTOSAR compatible.

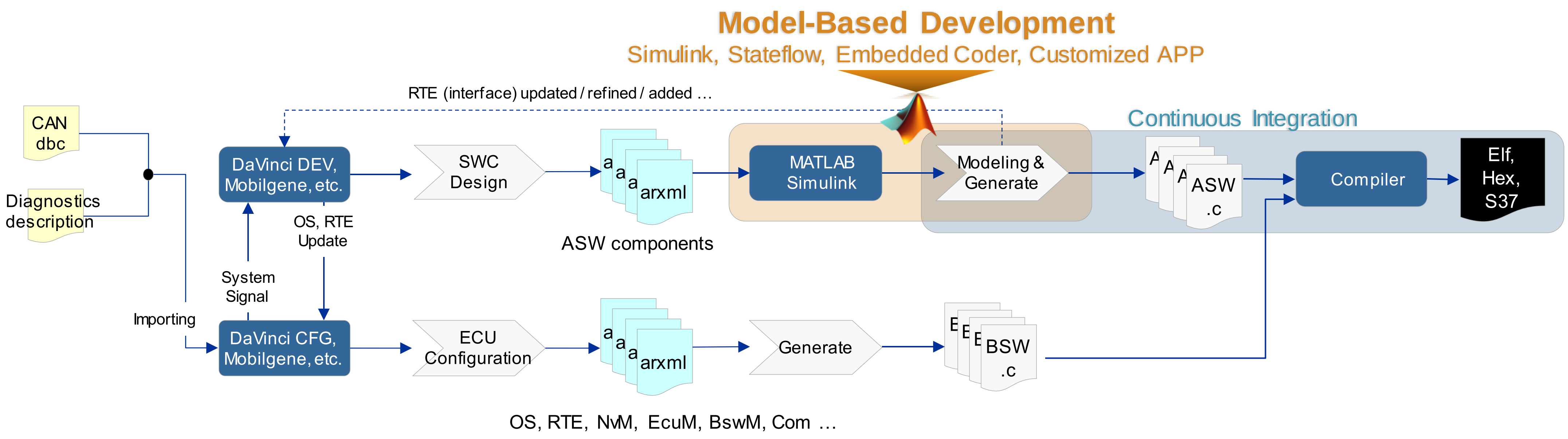


Top-Down Workflow

How Mobis develops the Fail-Safe software (2)

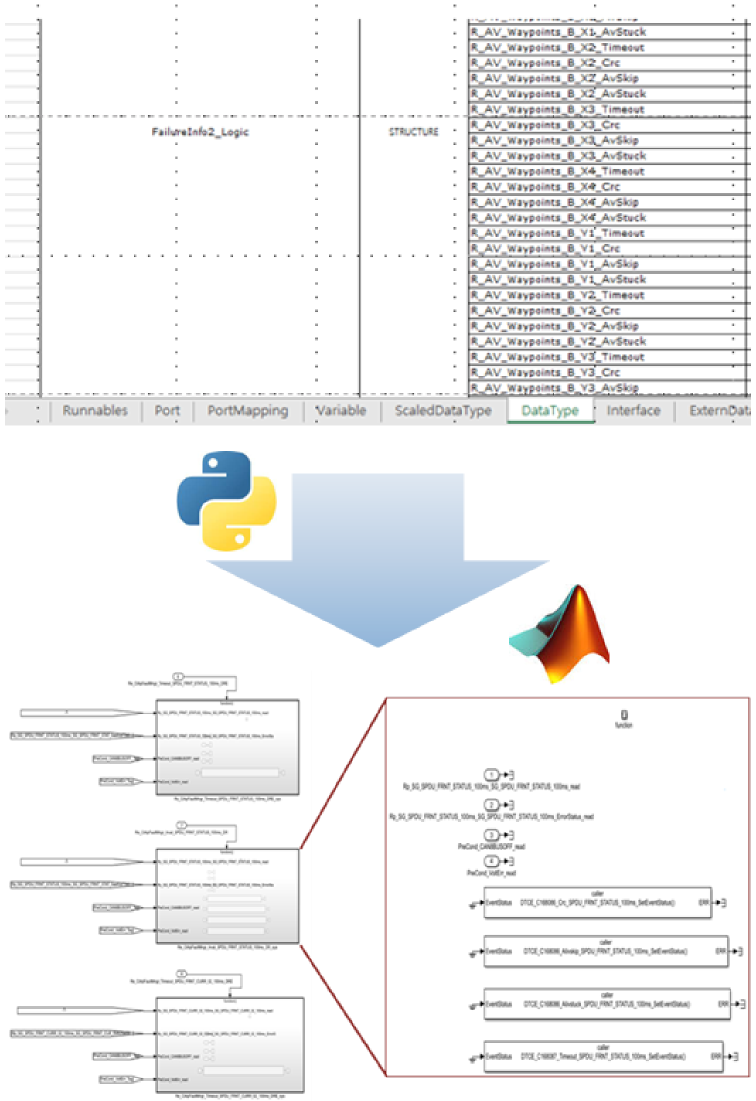
Model-Based Development

- The Fail-Safe application software is developed with Simulink, Stateflow and Embedded Coder
- The Simulink models are auto-code generated and integrated in the CI environment



Creating Skeleton model from architecture design

- Design software architectural attributes
  - runnable and trigger type, port interfaces, data types, mapping information …
- Generate ARXML file from the architecture design
  - In-house converter fills out XML metadata based on AUTOSAR shema
- Refine ARXML file with the AUTOSAR authoring tool (DaVinci, Mobilgene …)
  - VFB is realized by connecting with BSW
- Create a Simulink Skeleton model by importing the refined ARXML file with using Matlab commands.

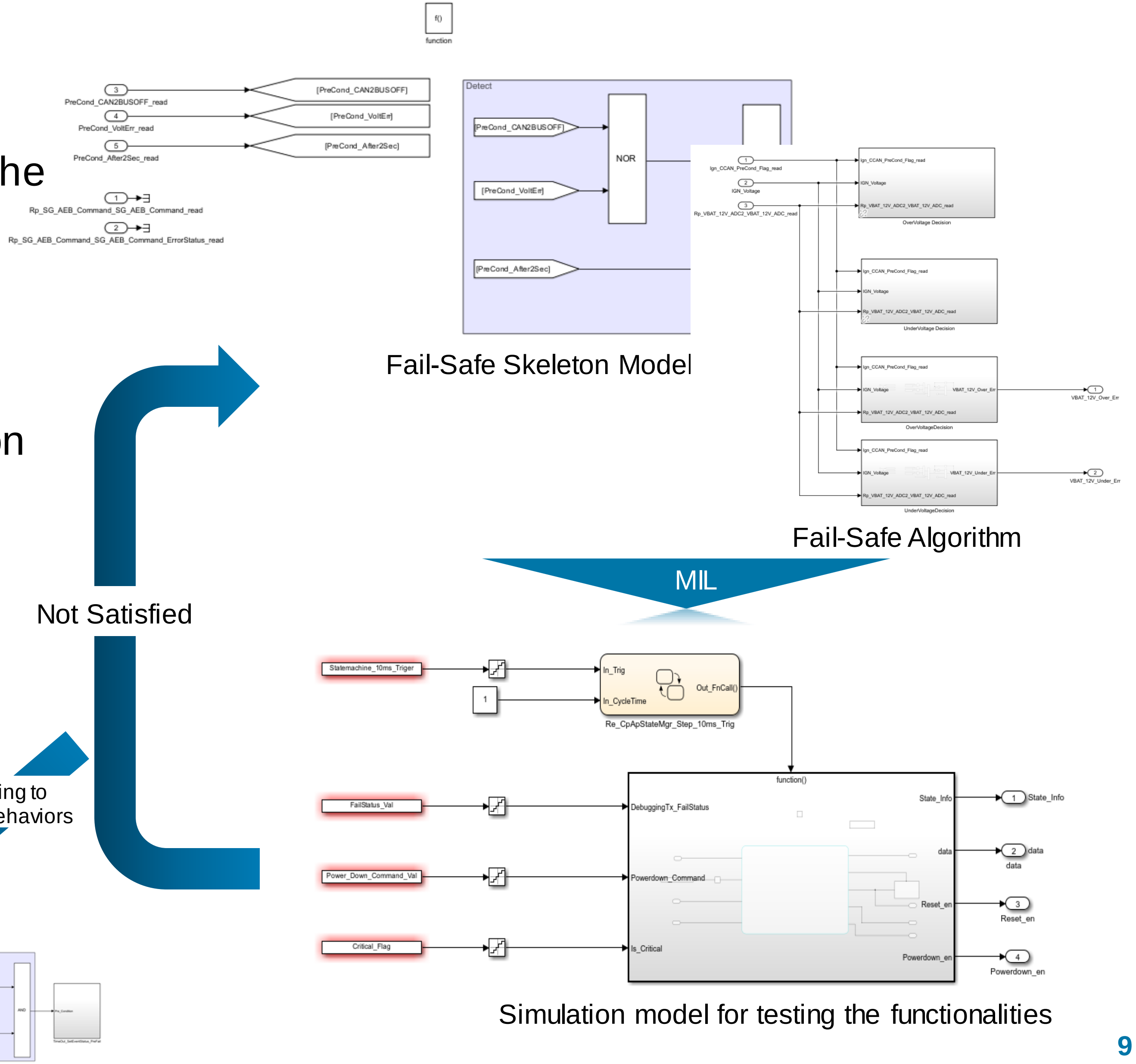


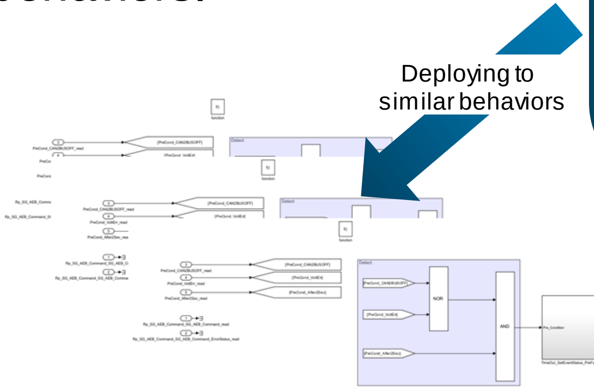
Top Level

Subsystem Level

Designing the Fail-Safe algorithm within the Skeleton (MIL)

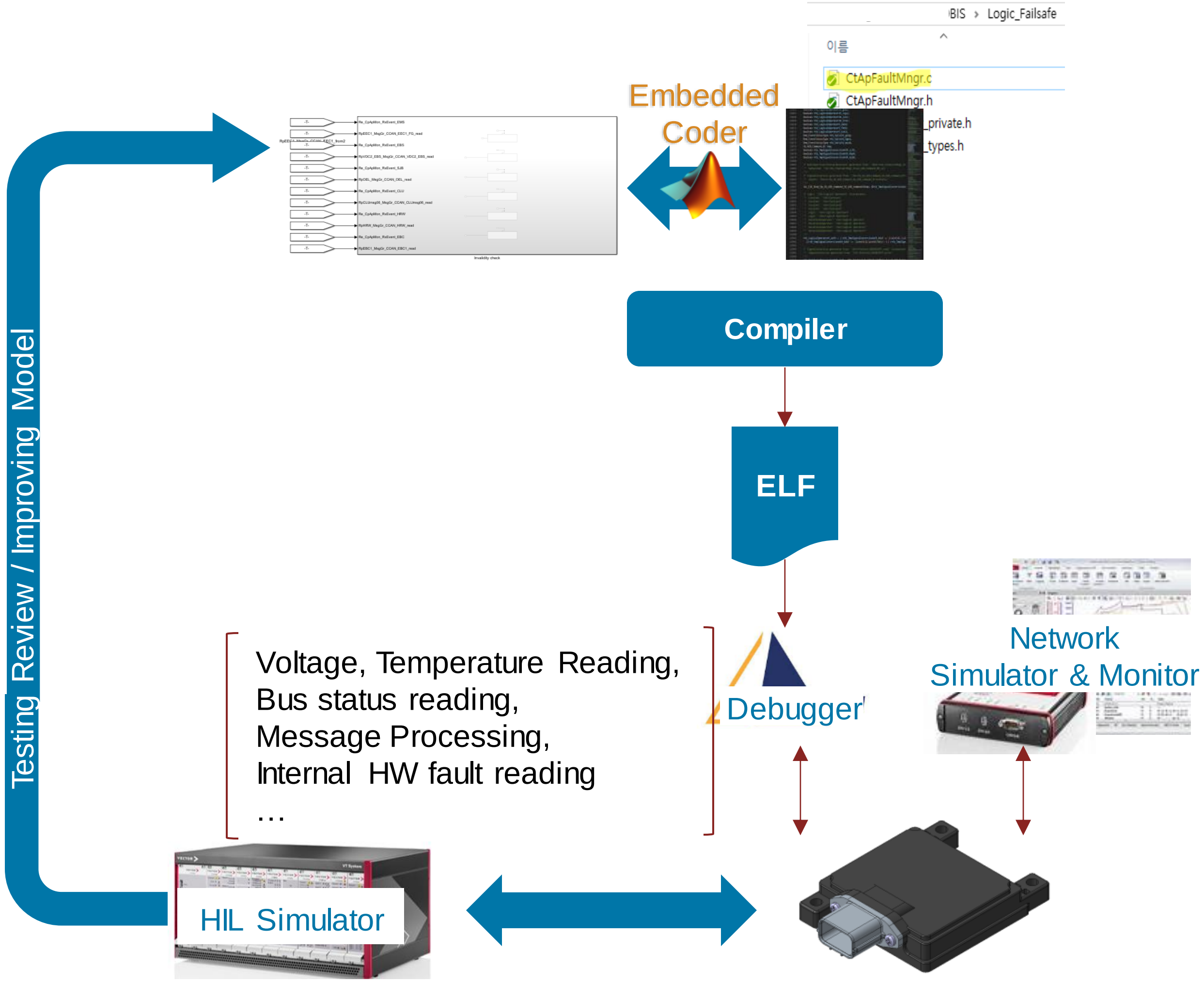
- Constructing a simulation model for verifying the goals of the software
  - Testing of increment/decrement of fault count, expectation when changing preconditions or inputs …
- Designing the internal behaviors of the function (runnable) for fault detection and reaction
  - Out of voltage range, Timeout, E2E violations, Invalidity, Bus-Off …
- Verifying the functionalities via MIL and improving the function behaviors.





Developing the Fail-Safe Software code using the embedded coder and HIL

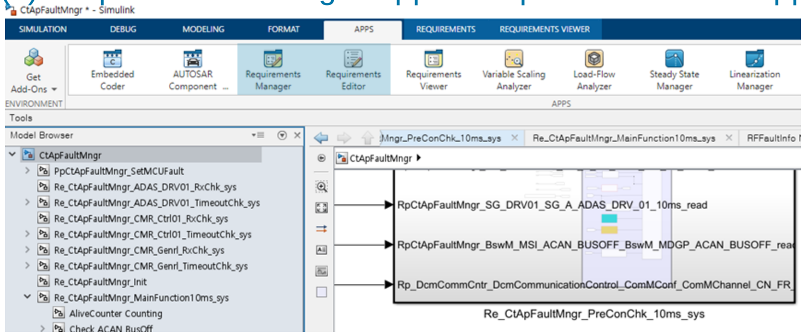
- Auto C code generation from the verified module using the embedded coder and generating executable image (ELF, Hex …) after integrating with AUTOSAR codes,
- Simulating the faults situation using a hardware simulator and monitoring the Fail-Safe Software via the debugger and network monitoring equipment,
  - Out of voltage range, Timeout, E2E violations, Invalidity, Bus-Off, DTC status …
- Verifying the functionalities and generated C code via HIL and improving the function behaviors.



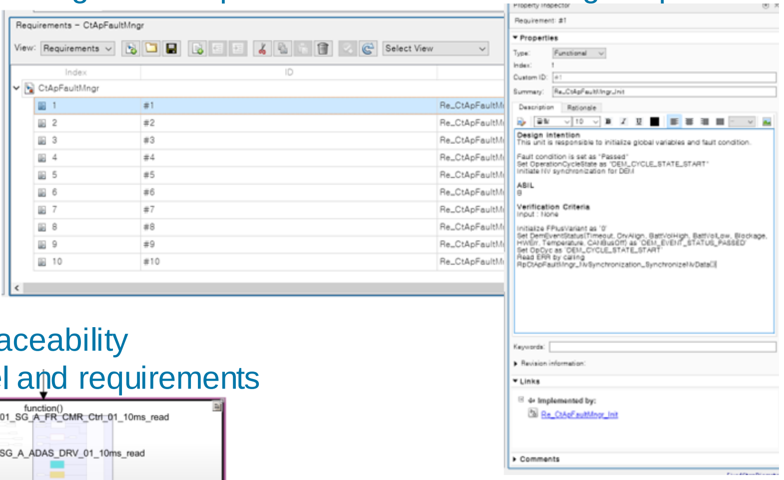
Requirement management using the Requirement Tool Box

- To define and descript the unit of Fail-Safe functionalities, use the Requirement Manager and Requirement Editor Apps,
- Using Requirement Manager App and Requirement Editor App, adding the new requirement set on the top-level and creating new requirement on the sub-system (meaning unit, runnable …) to define and descript the functionalities,
- Creating the traceability between sub-system (meaning unit, runnable …) and requirements.

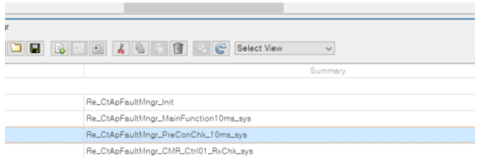
(1) Requirement Manager App & Requirement Editor App



(2) Adding New Requirement Set and Creating Requirements



(3) Creating Traceability between model and requirements

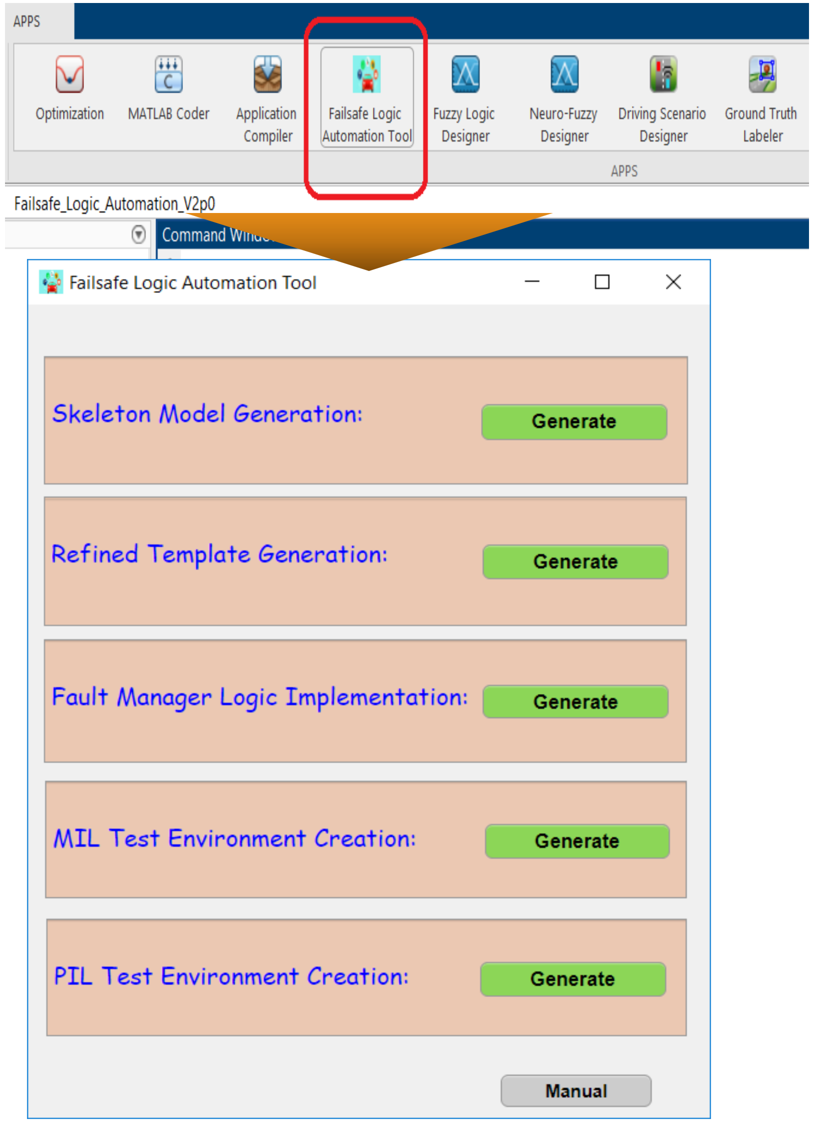


Improving work efficiency – Automated Model Creating and Updating (1)

- Following the metrics for sub-system (meaning units, runnables…) count,

S.No	Failure Mode	Runnable Count (in Model)/DTC Count
1	CAN Bus Off	9
2	Message Time-out	218
3	Signal Invalidity	218
4	Battery Voltage Failures	5
5	Others	20

- As part of this automation, the following Fail-Safe modes are considered (100% automated)
  - CAN Bus Off, Message Time Out, Signal Invalidity

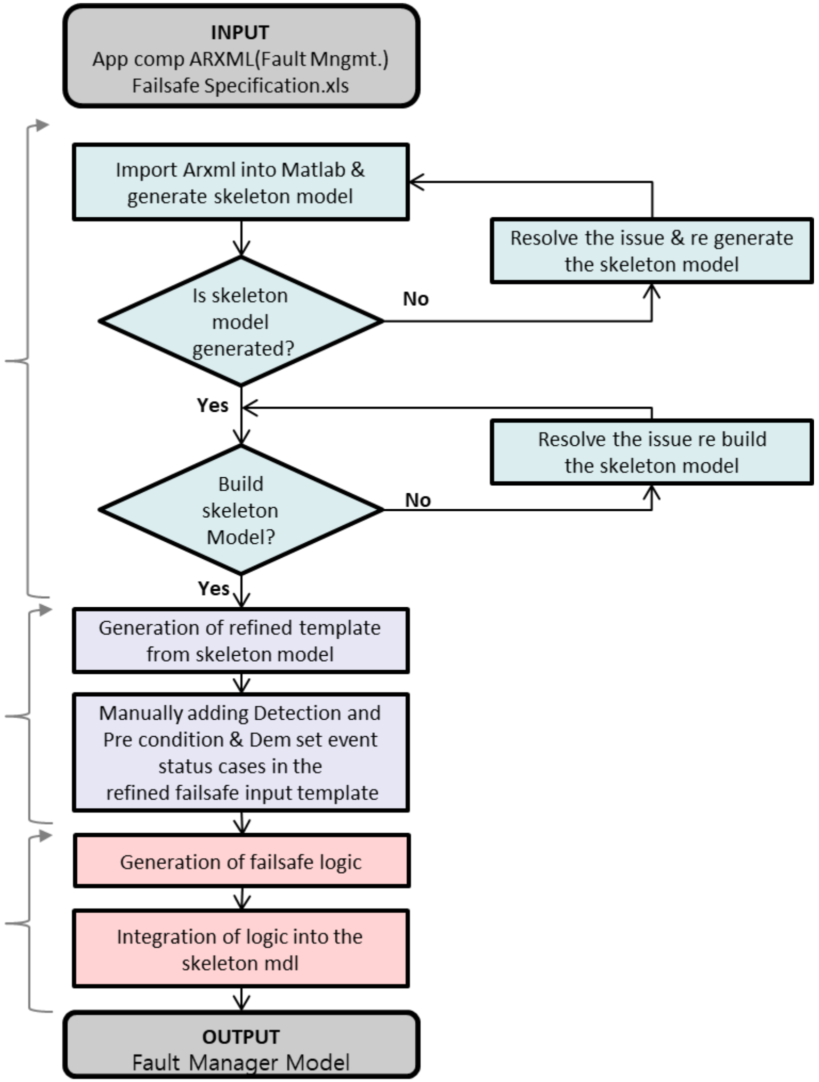


Improving work efficiency – Automated Model Creating and Updating (2)

Phase-1:
Creation of Fault Manager Skeleton Model

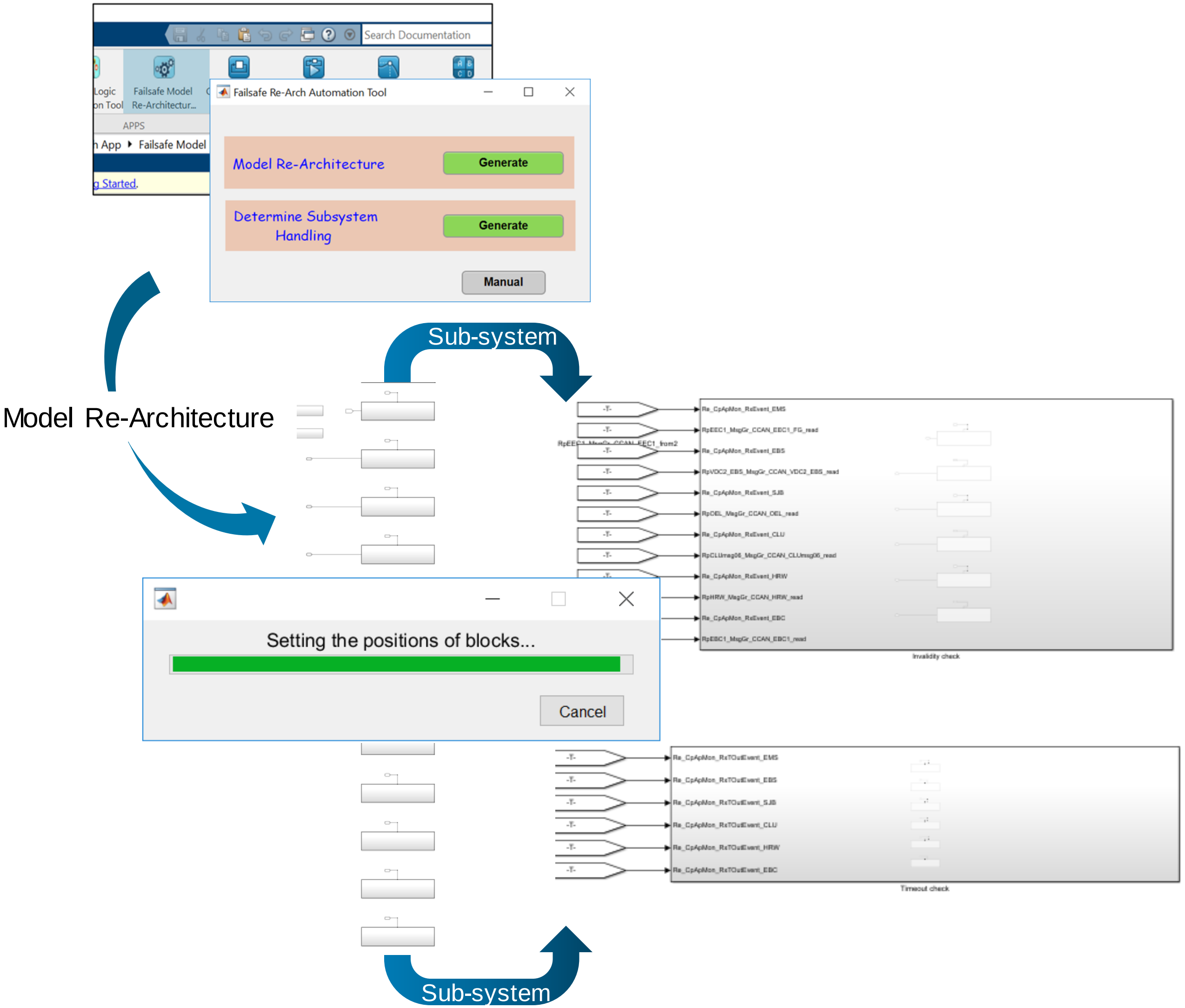
Phase-2:
Generation & Updation of refined failsafe input template

Phase-3:
Detction logic automation and Auto code generation



Improving work efficiency – Automated Model Re-Architecting

- All runnables are created on the Top-Level in the skeleton model,
- Necessary to group similar runnables together to improve readability,
  - Timeout, Invalidity …
- Creating sub-system and move the runnables to corresponding sub-system automatically.

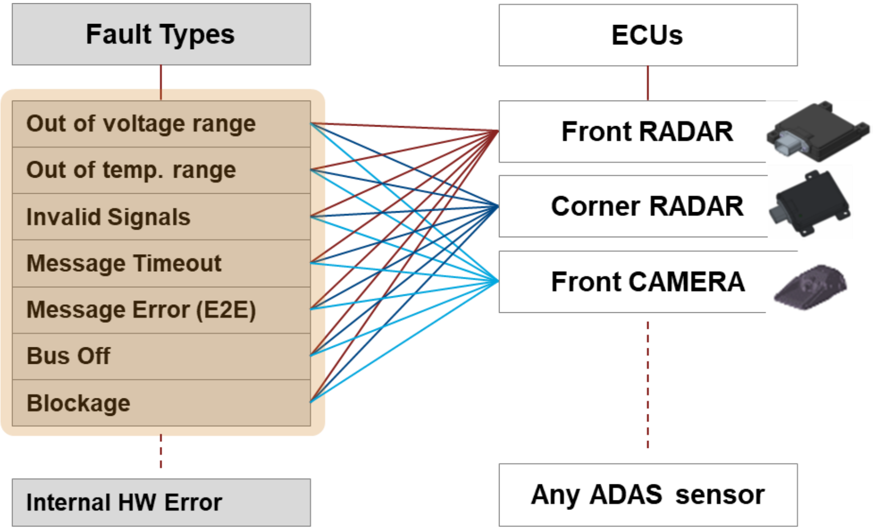


Concluding Remarks : ‘*Why Mobis applies Model-Based Development to the Fail-Safe Software?*’

<u>Reusability</u>

- Different Sensors/ECUs but similar fault detection algorithms.

How can we make it easy to reuse?



<u>Quality Maintenance</u>

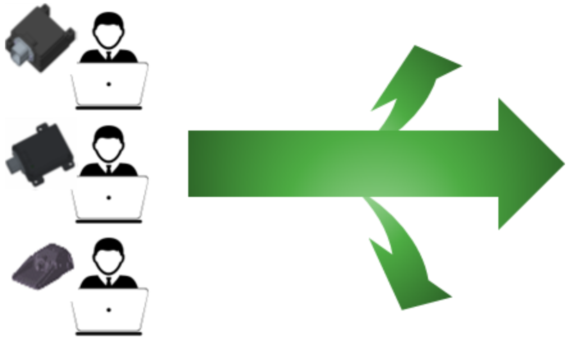
- Developers’ different coding capabilities. ASPICE consistency hard to achieve. Coding rule compliance.

How can we make it easy to maintain?

<u>Collaboration</u>

- Ambiguity in natural language (especially in international collaboration)

How can we make it easy to share design intention precisely?



Future Works

System Composer and Requirement Management

- Designing the Fail-Safe Software Architecture in Simulink and managing software requirement & software architecture requirement …

Advanced usage of Autosar Tool Set

- Improving the simulation environment using Dem, Nvram Tool box

Enhanced Fail-Safe automation

- Automatically adding the Fail-Safe mechanism to the software component ports, interfaces … and validating whether it has been applied as intended.

MathWorks
AUTOMOTIVE CONFERENCE 2023
Korea

Thank you