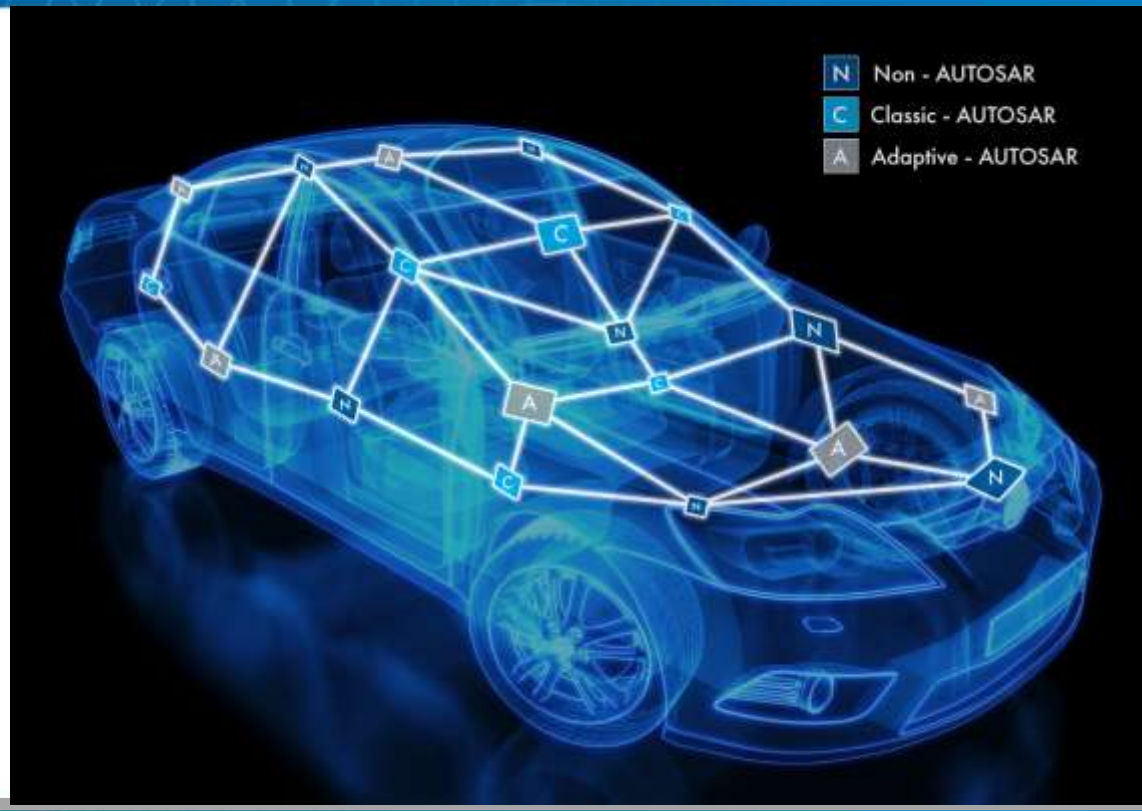
Comprehensive workflow for AUTOSAR Classic & Adaptive using Model-based Design



Durvesh Kulkarni
Senior Application Engineer
MathWorks India

Agenda

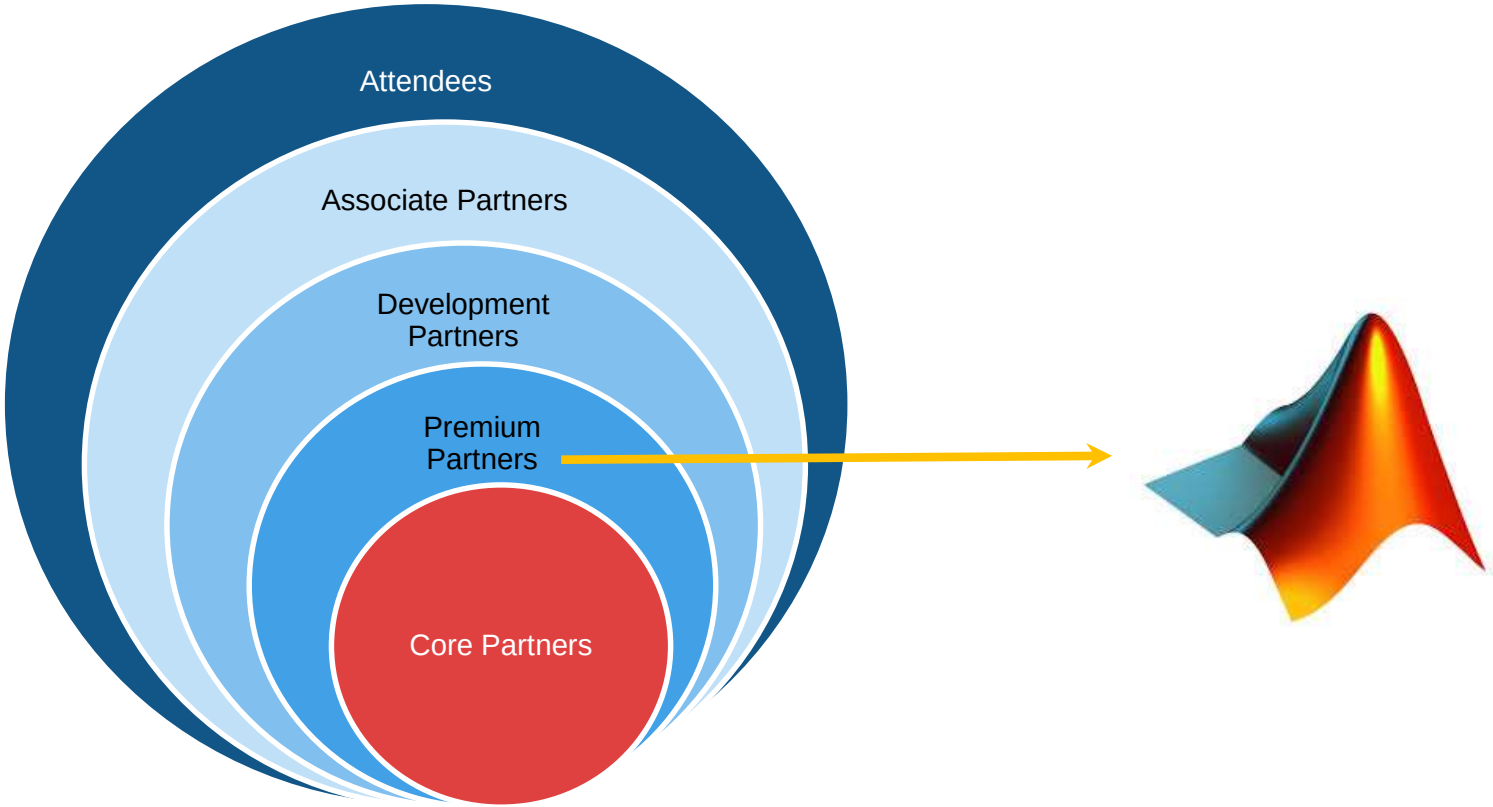
- Introduction to AUTOSAR
- Simulink for Classic Platform
  - Automatic modeling and code generation
  - Simulation of AUTOSAR ECU software
  - Blocks for AUTOSAR Library routines
  - Importing and exporting AUTOSAR descriptions artifacts (ARXML files)
- Simulink for Adaptive Platform
  - A closer look at the Adaptive layers
  - Motivation for Simulink to support Adaptive
  - Mapping Adaptive platform to Simulink
  - Code Generation for Adaptive components
- Polyspace for AUTOSAR
- Additional Resources

Agenda

- Introduction to AUTOSAR
- Simulink for Classic Platform
  - Automatic modeling and code generation
  - Simulation of AUTOSAR ECU software
  - Blocks for AUTOSAR Library routines
  - Importing and exporting AUTOSAR descriptions artifacts (ARXML files)
- Simulink for Adaptive Platform
  - A closer look at the Adaptive layers
  - Motivation for Simulink to support Adaptive
  - Mapping Adaptive platform to Simulink
  - Code Generation for Adaptive components
- Polyspace for AUTOSAR
- Additional Resources

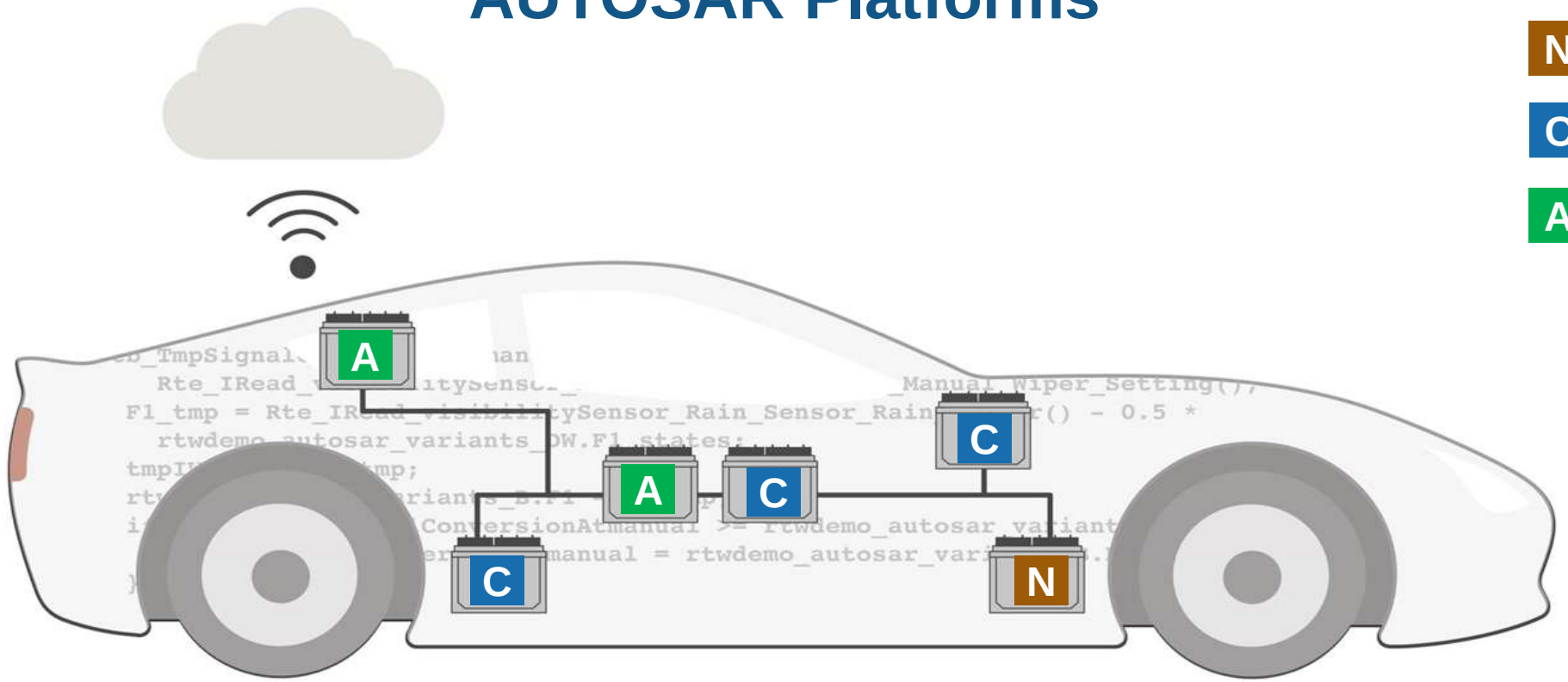
Introduction to AUTOSAR

- AUTOSAR – **AUT**omotive **O**pen **S**ystems **Ar**chitecture
  - Middleware standard, jointly developed by automobile manufacturers, electronics and software suppliers and tool vendors.
  - Motto: “cooperate on standards, compete on implementations”



AUTOSAR Platforms

OVER THE AIR UPDATE



N Non - AUTOSAR

C Classic - AUTOSAR

A Adaptive - AUTOSAR

Non- AUTOSAR

Software

Hardware

Classic AUTOSAR

Application Software

RTE

Basic Software

Hardware

Adaptive AUTOSAR

Adaptive Application Software

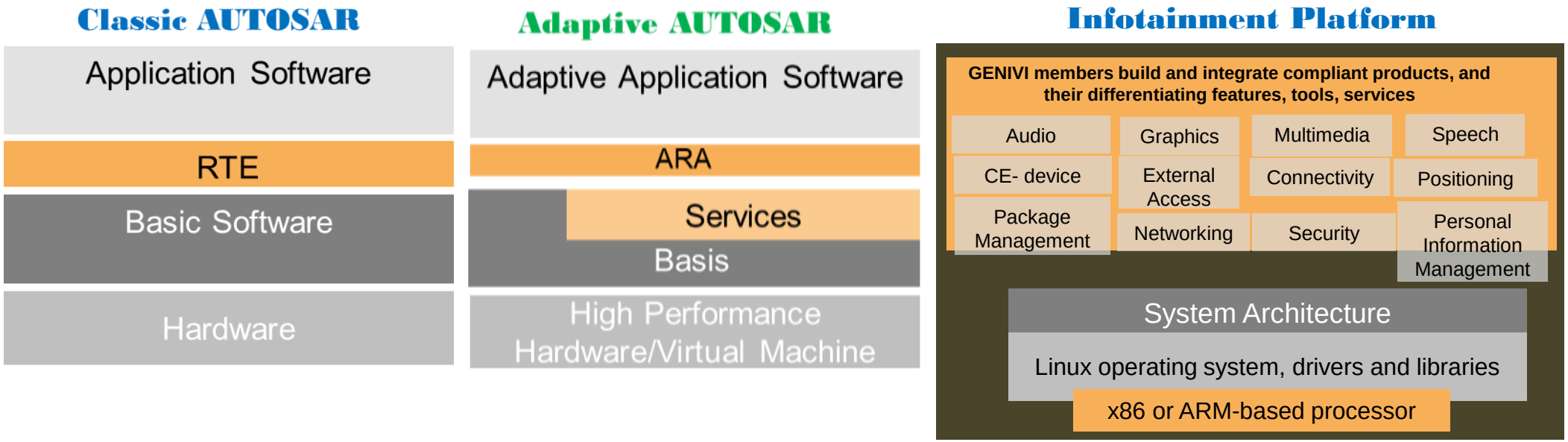
ARA

Services

Basis

High Performance Hardware/Virtual Machine

A new platform for compute intensive applications



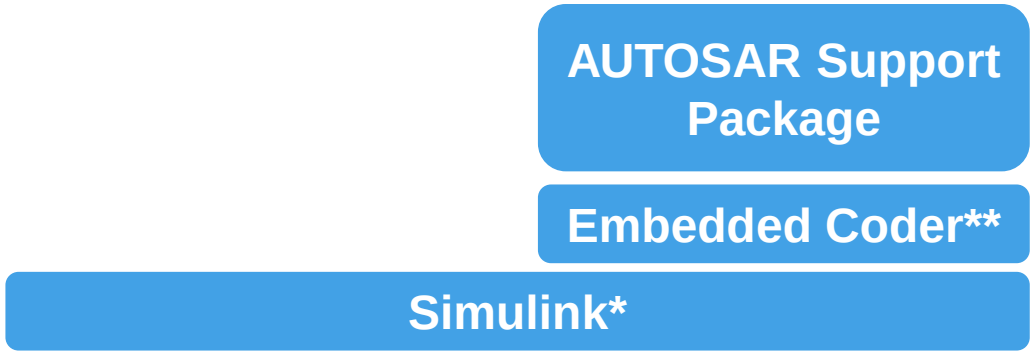
Real time Requirements	**High,** **in the range of micro-sec**	**Mid,** **in the range of milli-sec**	**Low,** **in the range of sec**
Computing power	Low, ~ 1000 MIPs	High, > 20,000 MIPs	High, ~ 10,000 MIPs

MIPs: Million Instructions per Second

Source: AUTOSAR Adaptive Platform Joint Meeting May 2017

AUTOSAR Support Transition

- R2018b and earlier



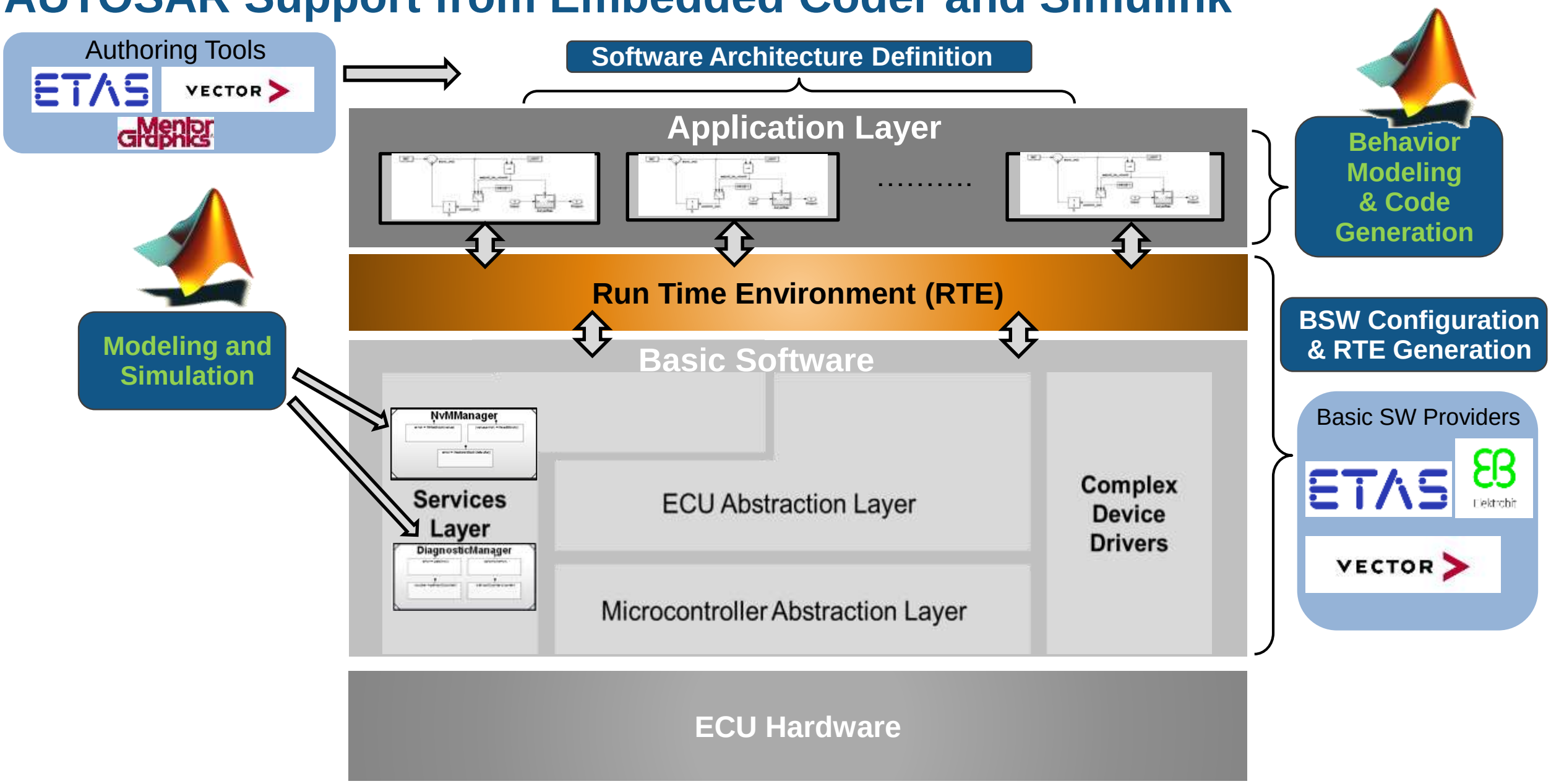
Required

*Requires MATLAB
**Requires MATLAB Coder and Simulink Coder

Agenda

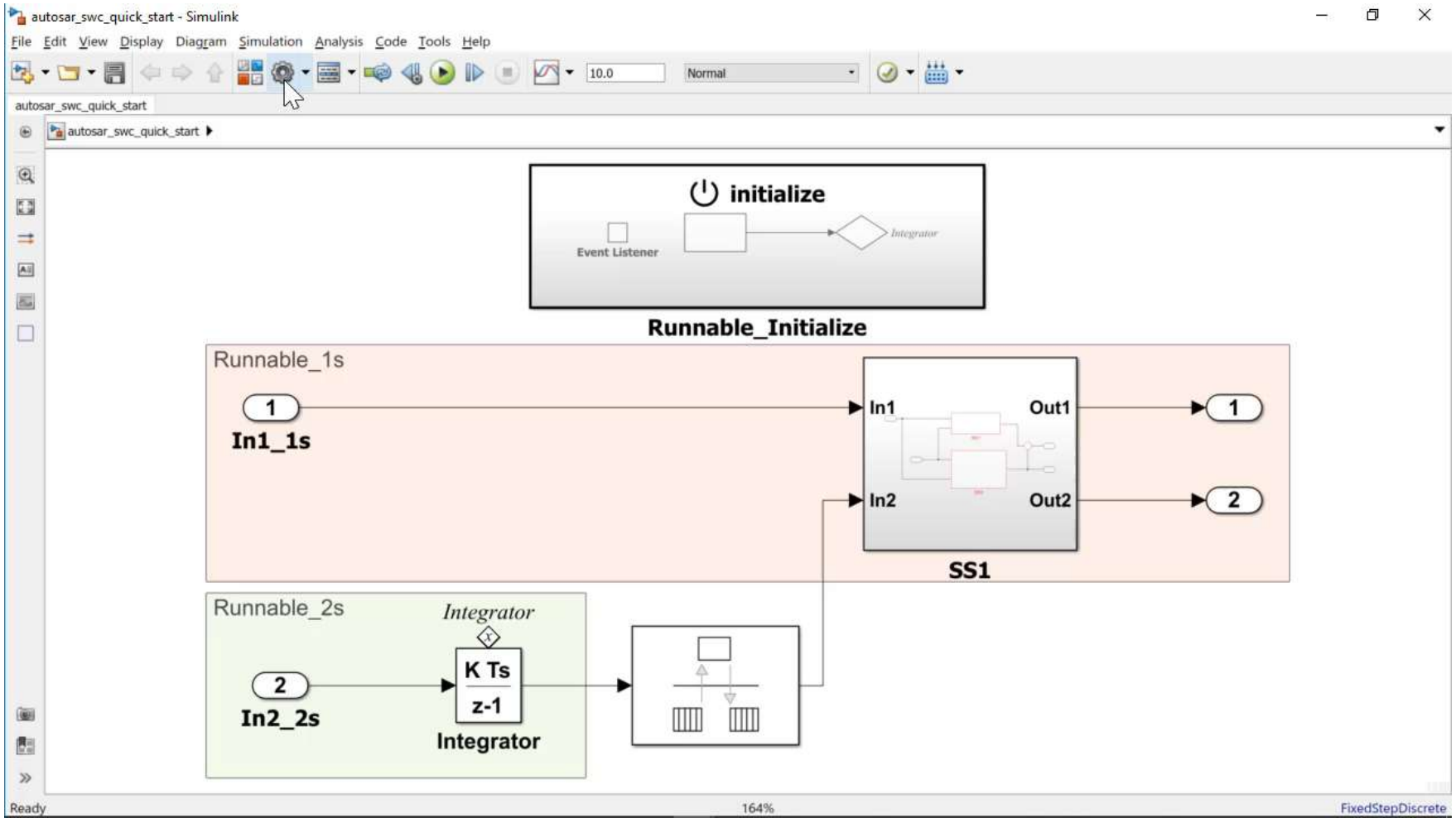
- Introduction to AUTOSAR
- Simulink for Classic Platform
  - Automatic modeling and code generation
  - Simulation of AUTOSAR ECU software
  - Blocks for AUTOSAR Library routines
  - Importing and exporting AUTOSAR descriptions artifacts (ARXML files)
- Simulink for Adaptive Platform
  - A closer look at the Adaptive layers
  - Motivation for Simulink to support Adaptive
  - Mapping Adaptive platform to Simulink
  - Code Generation for Adaptive components
- Additional Resources

AUTOSAR Support from Embedded Coder and Simulink



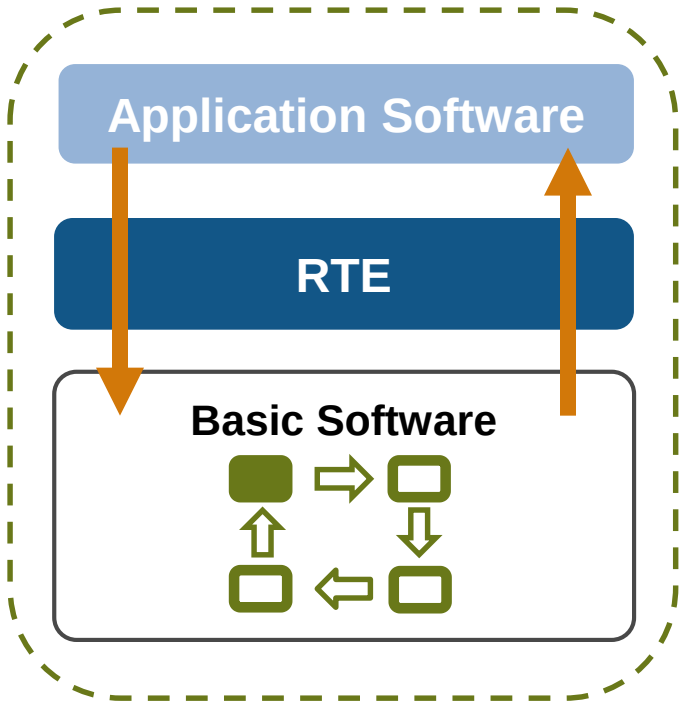
Automatic modeling and code generation

- Show quick start demo, edit in code perspective UI and AUTOSAR dict, code gen



Functional simulation of AUTOSAR basic software is critical for AUTOSAR ECU development

AUTOSAR ECU layered architecture

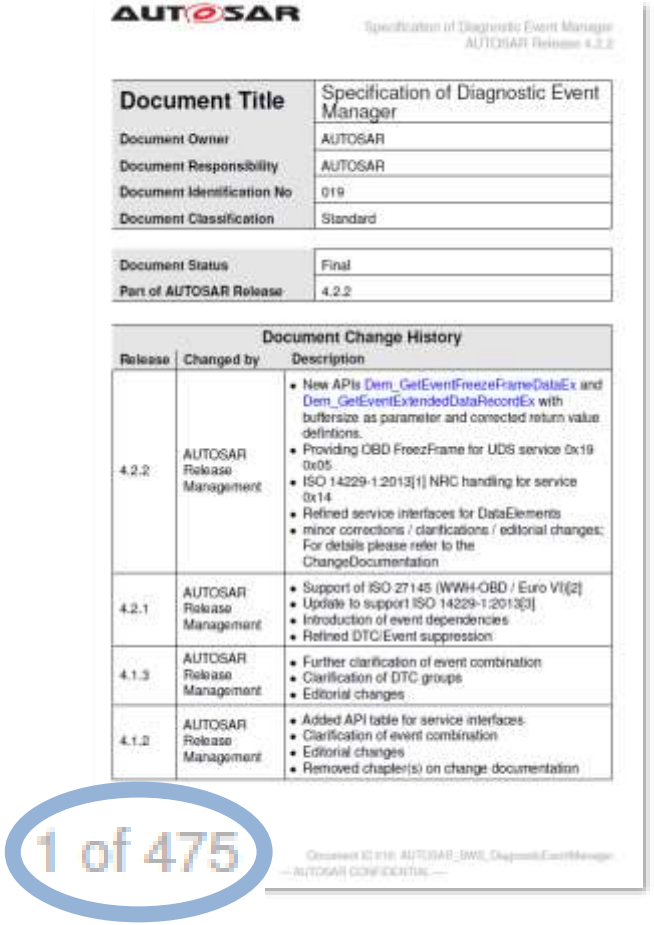


Many calls between application software and basic software

Basic software functionality is highly dynamic

Simulation of basic software reduces development time and improves software quality

Basic software library makes functional simulation of AUTOSAR basic software as easy as pressing the play button



AUTOSAR

Document Title	Specification of Diagnostic Event Manager
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	019
Document Classification	Standard

Document Status	Final
Part of AUTOSAR Release	4.2.2

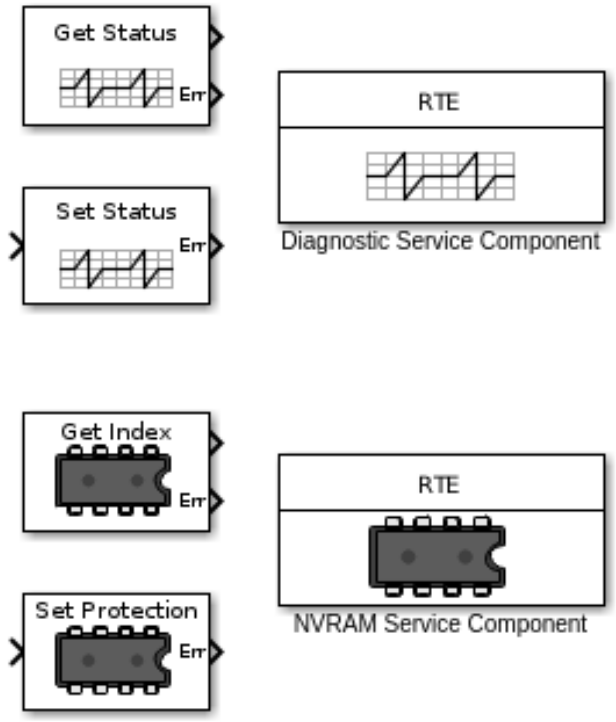
Document Change History

Release	Changed by	Description
4.2.2	AUTOSAR Release Management	• New APIs Dem_GetEventFreezeFrameDataEx and Dem_GetEventExtendedDataRecordEx with buffersize as parameter and corrected return value definitions. • Providing OBD FreezFrame for UDS service 0x19 0x05 • ISO 14229-1:2013[1] NRC handling for service 0x14 • Refined service interfaces for DataElements • minor corrections / clarifications / editorial changes; For details please refer to the ChangeDocumentation
4.2.1	AUTOSAR Release Management	• Support of ISO 27145 (WWH-OBD / Euro VI)[2] • Update to support ISO 14229-1:2013[3] • Introduction of event dependencies • Refined DTC/Event suppression
4.1.3	AUTOSAR Release Management	• Further clarification of event combination • Clarification of DTC groups • Editorial changes
4.1.2	AUTOSAR Release Management	• Added API table for service interfaces • Clarification of event combination • Editorial changes • Removed chapter(s) on change documentation

1 of 475

Detailed Specifications

Encapsulated in



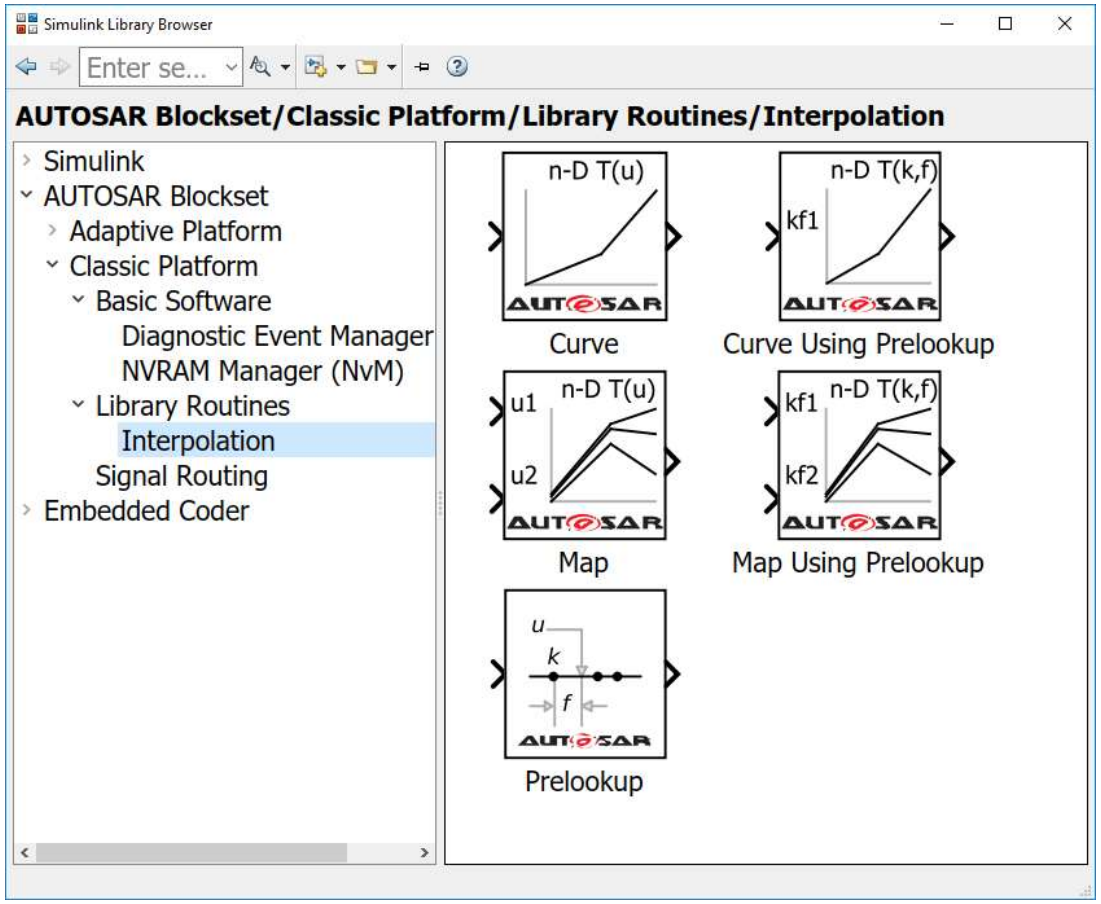
R2017b

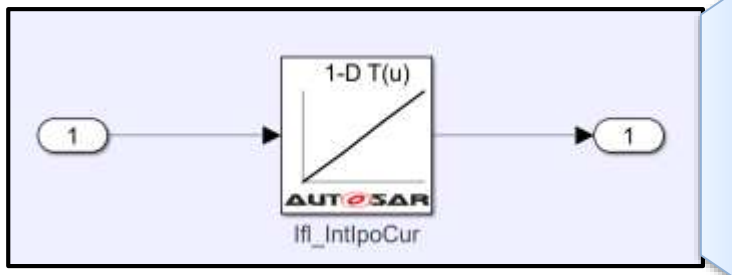
Basic Software Library

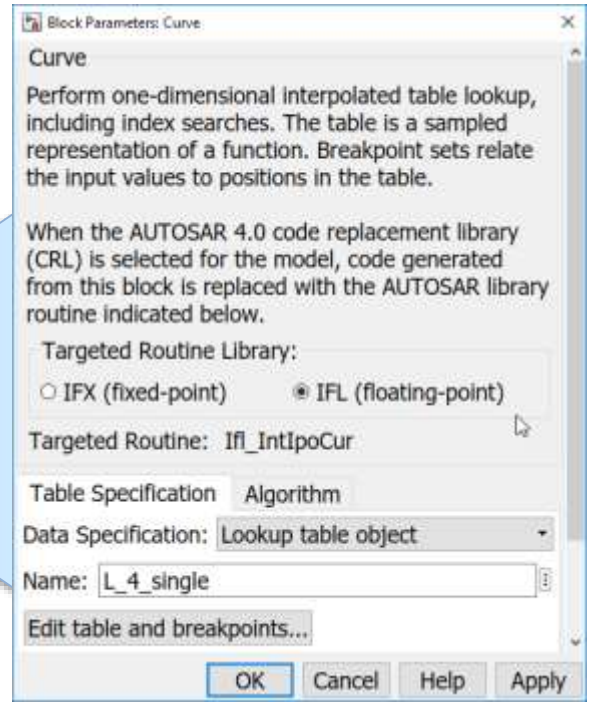
Simulation of AUTOSAR ECU software

- Seat Belt Reminder demo

AUTOSAR Library Routines

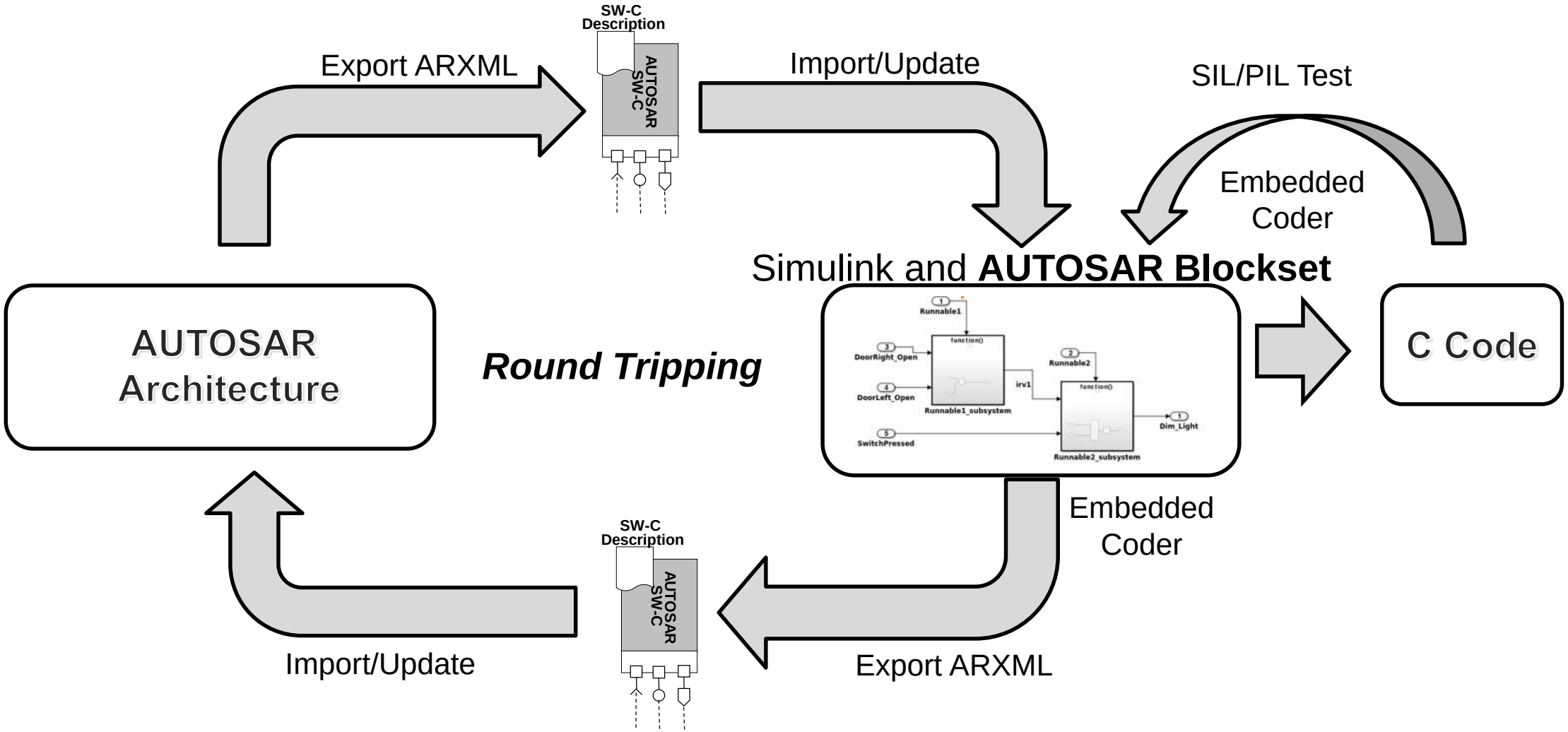







```
Rte_IWrite_Runnable_Step_Out1_Out1(Ifl_IntIpoCur_f32_f32
  (Rte_IRead_Runnable_Step_In1_In1(), L_4_single.Nx, L_4_single.Bp1,
   L_4_single.Table));
```

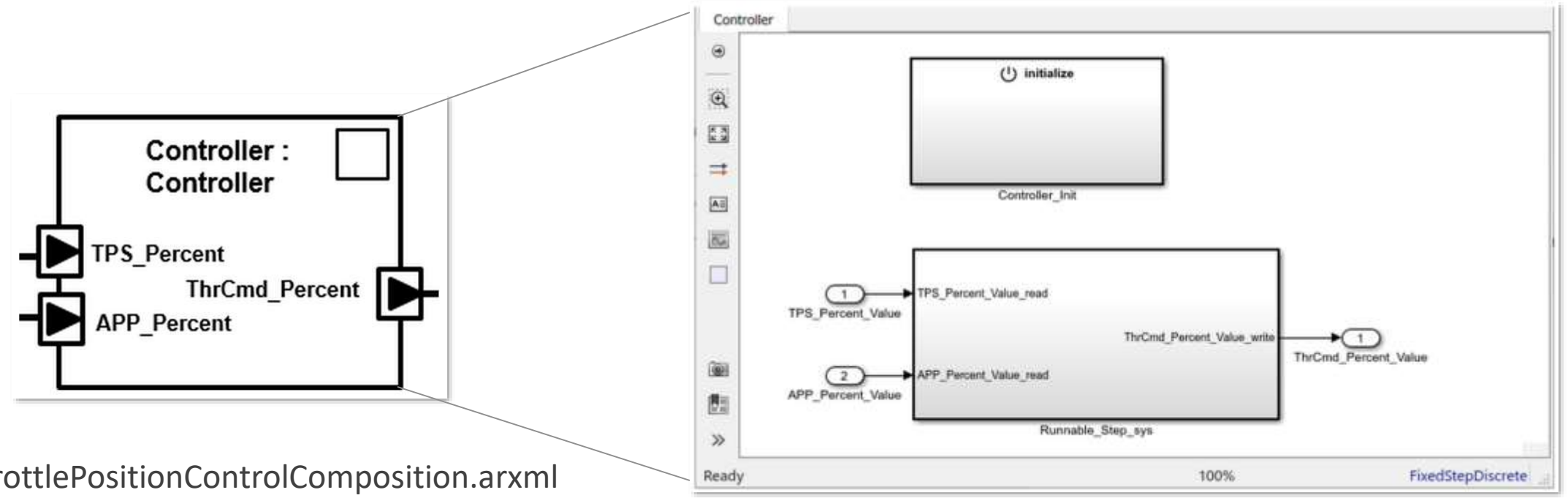
Importing and exporting AUTOSAR descriptions (ARXML files)



Import AUTOSAR XML to Simulink

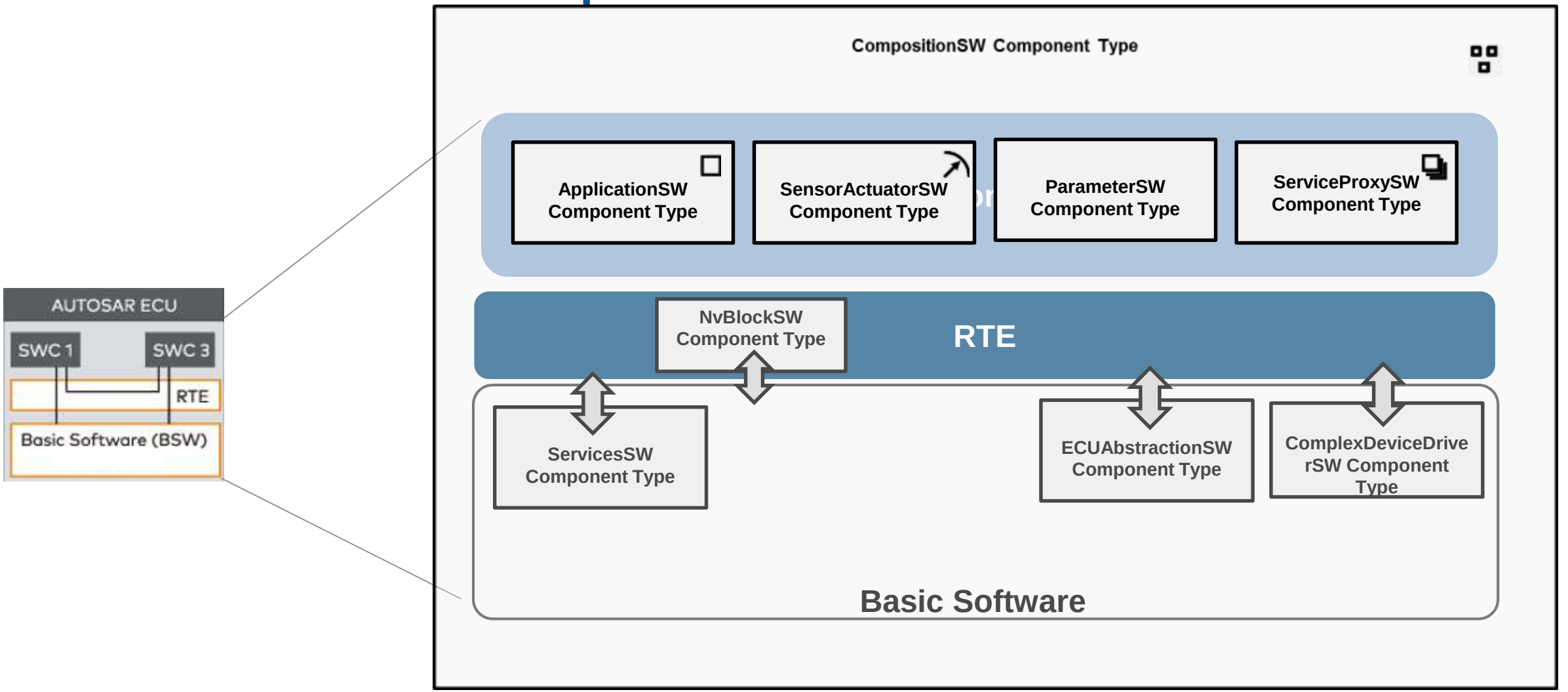
Import AUTOSAR Component to Simulink

```
ar = arxml.importer('ThrottlePositionControlComposition.arxml');
createComponentAsModel(ar,'/Company/Components/Controller',...
    'ModelPeriodicRunnablesAs','AtomicSubsystem');
```



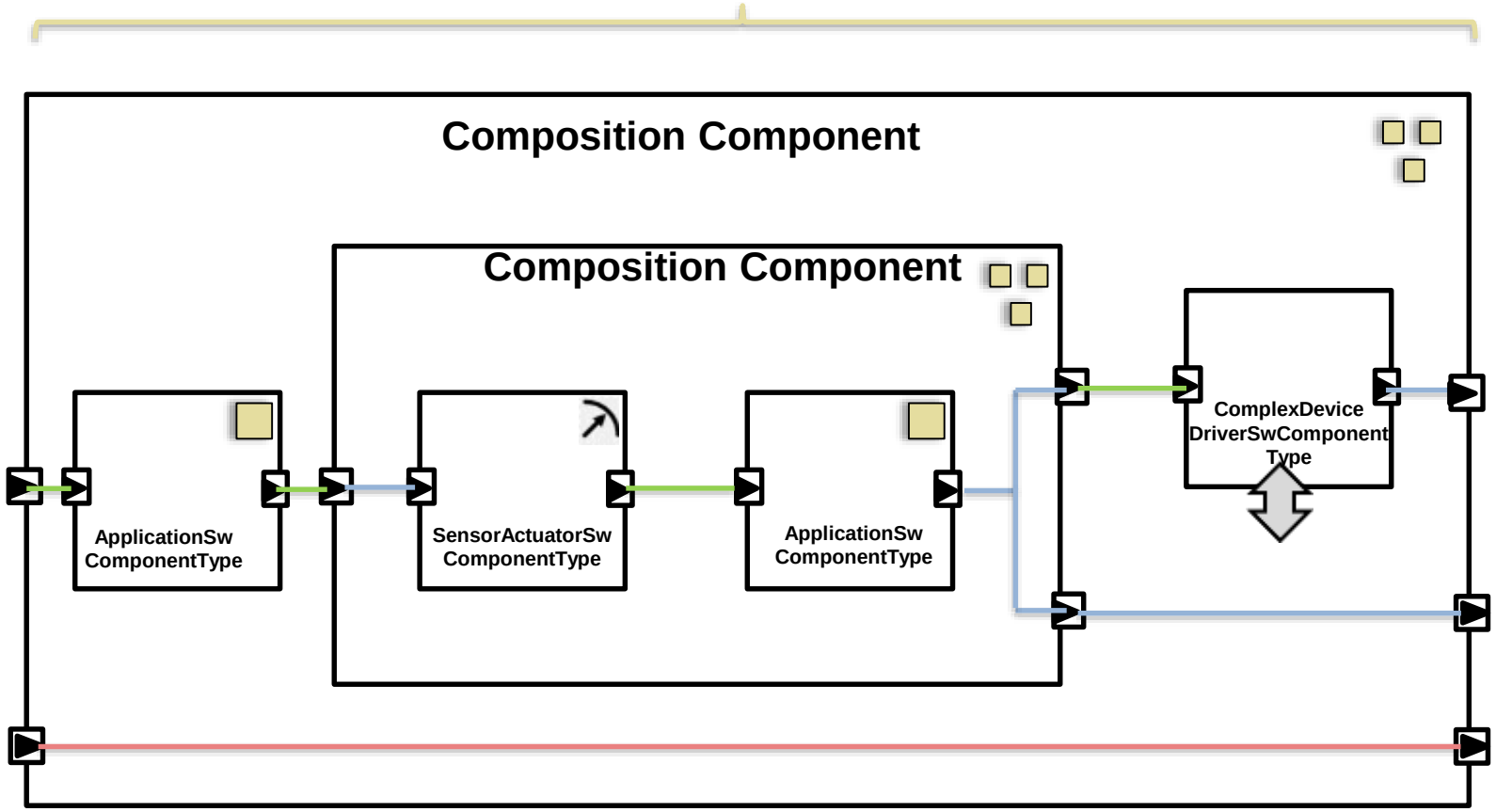
ThrottlePositionControlComposition.arxml

AUTOSAR Software Components



AUTOSAR Composition-Software-Component

- Compositions purely architectural element
  - Do not impact how components interact with RTE, and code



Composition component ➔ Hierarchical aggregation of software components

Live Editor - \\fs-58-ah\vmgr$\home08\sshwetha\Documents\MATLAB\examples\autosarstandard\ImportAUTOSARCompositionToSimulinkSpkgExample\ImportAUTOSARCompositionToSimulinkSpkgExample....

LIVE EDITOR | INSERT | VIEW

New | Open | Save | Find Files | Compare | Print | Go To | Find | Text | Title | Code | Section Break | Run Section | Run and Advance | Run to End | Run | Step | Stop

FILE | NAVIGATE | TEXT | CODE | SECTION | RUN

ImportAUTOSARCompositionToSimulinkSpkgExample.mlx

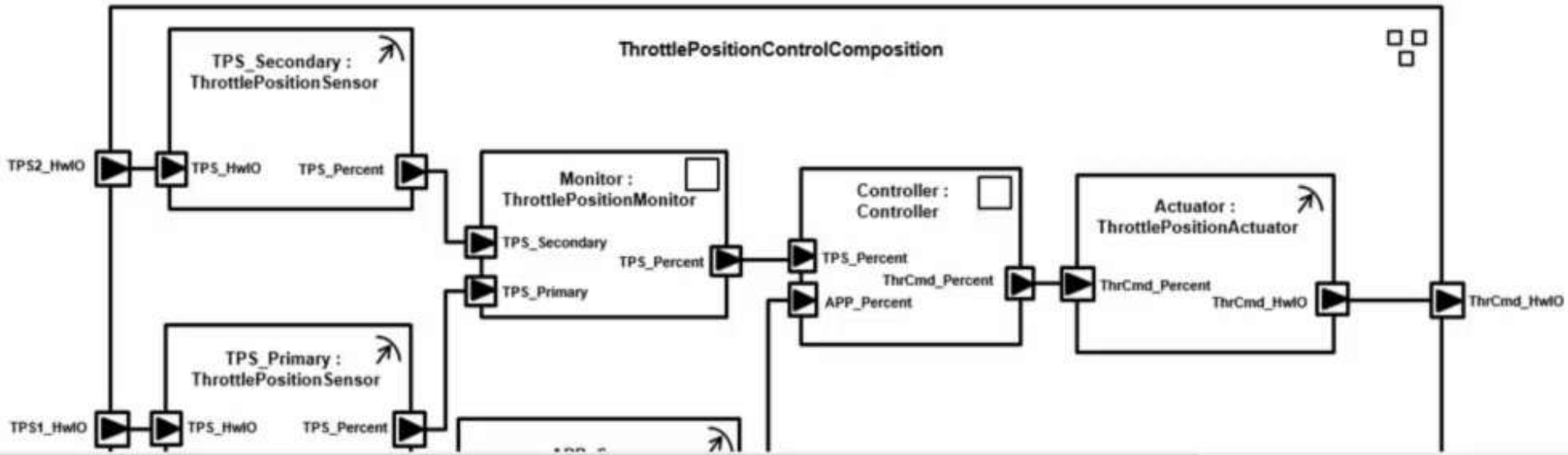
Import AUTOSAR Composition to Simulink

Create Simulink® representation of AUTOSAR composition imported from AUTOSAR authoring tool `arxml` file

Import AUTOSAR Composition from `arxml` File to Simulink

Here is an AUTOSAR software composition that implements a throttle position control system. The composition contains six interconnected AUTOSAR software component prototypes -- four sensor/actuator components and two application components.

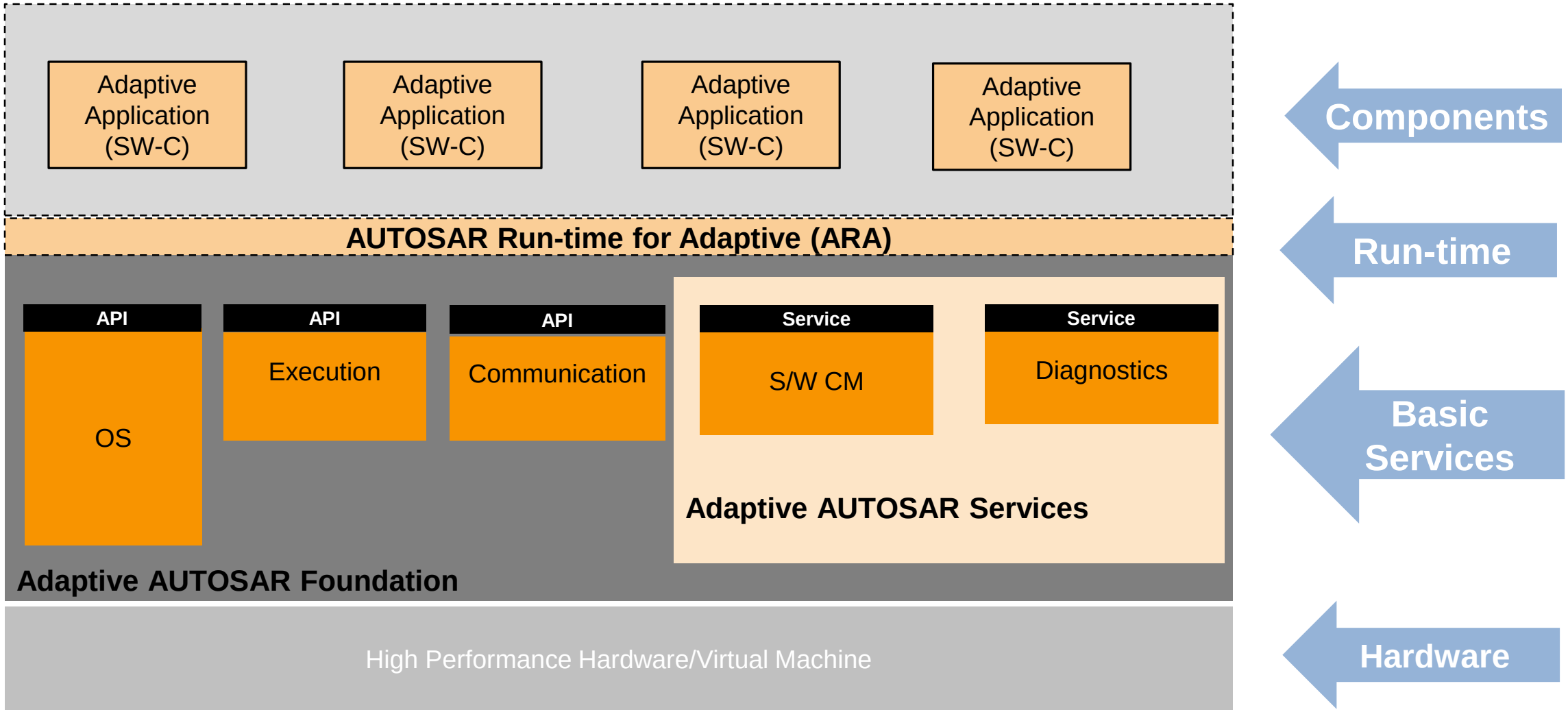
The composition was created in an AUTOSAR authoring tool and exported to the file `ThrottlePositionControlComposition.arxml`. (You can access the `arxml` files used in this example in the installed AUTOSAR support package tree, at `autosarroot/autosar_examples/ThrottlePositionControlSystem/arxml`.)



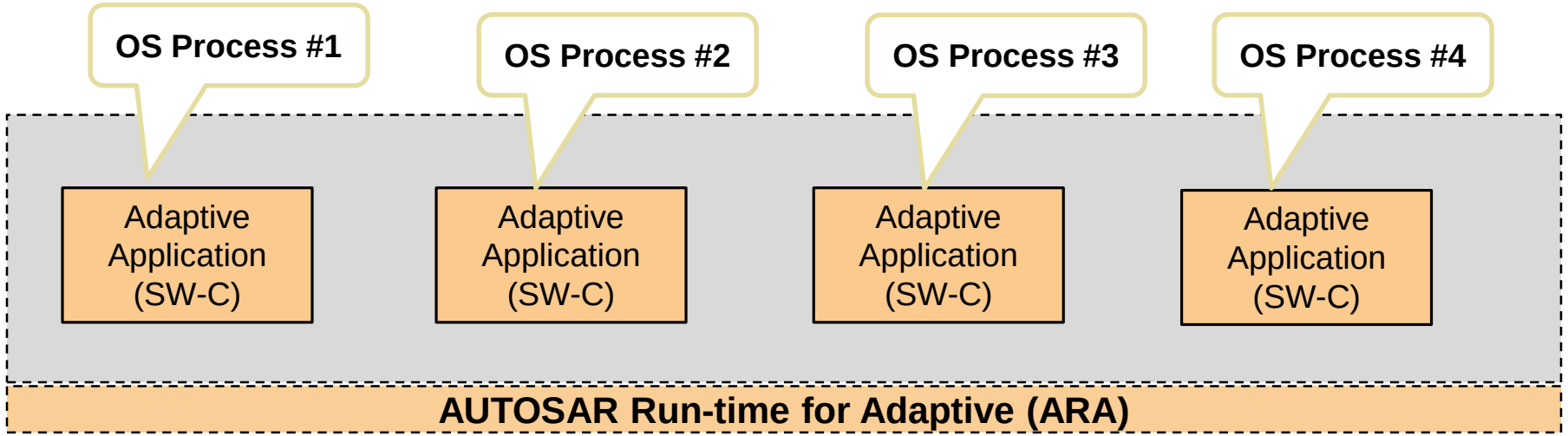
Agenda

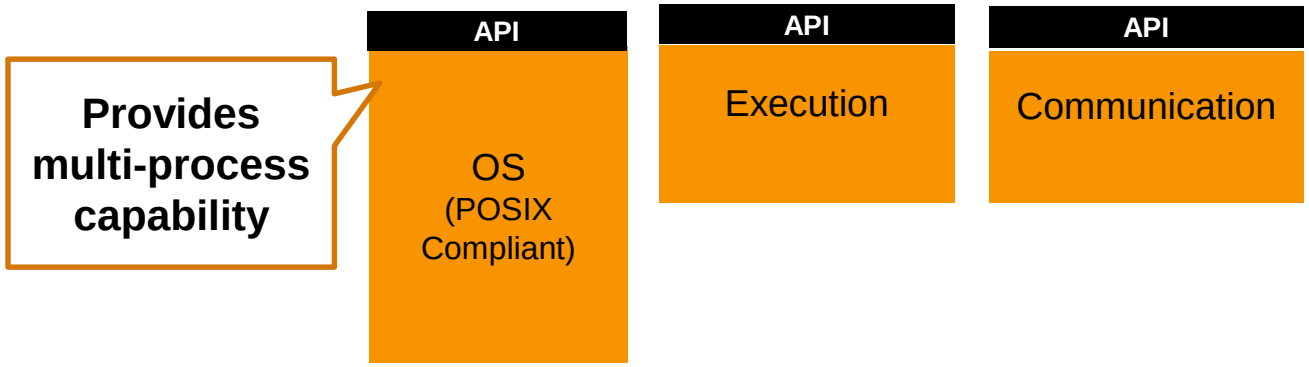
- Introduction to AUTOSAR
- Simulink for Classic Platform
  - Automatic modeling and code generation
  - Simulation of AUTOSAR ECU software
  - Blocks for AUTOSAR Library routines
  - Importing and exporting AUTOSAR descriptions artifacts (ARXML files)
- Simulink for Adaptive Platform
  - A closer look at the Adaptive layers
  - Motivation for Simulink to support Adaptive
  - Mapping Adaptive platform to Simulink
  - Code Generation for Adaptive components
- Polyspace for AUTOSAR
- Additional Resources

AUTOSAR Layered software architecture



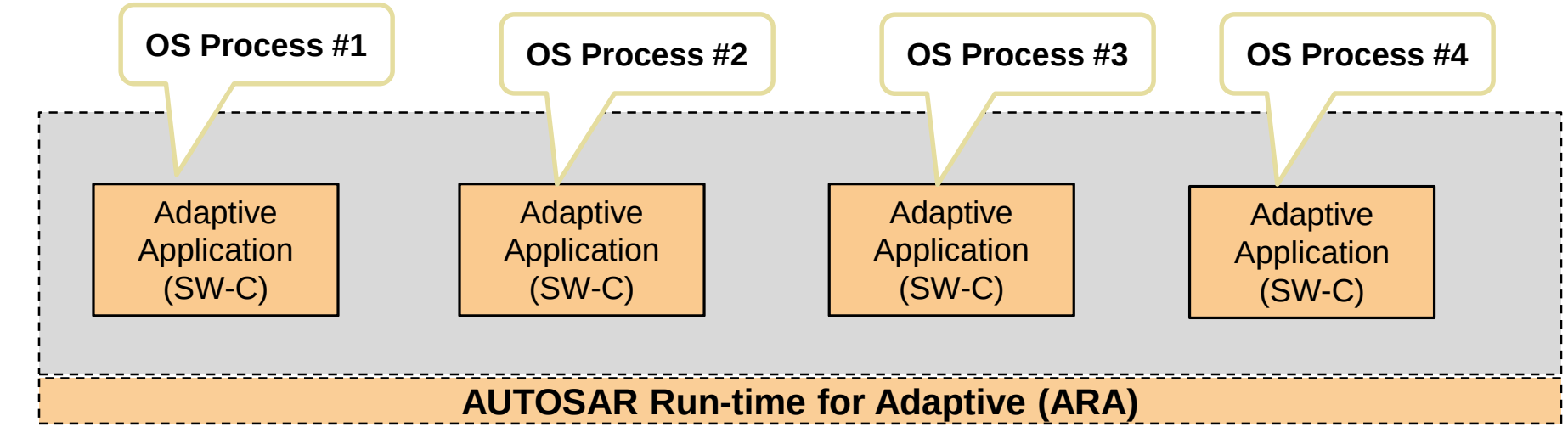
Key Concept #1
Everything is a process .. as in “OS process”

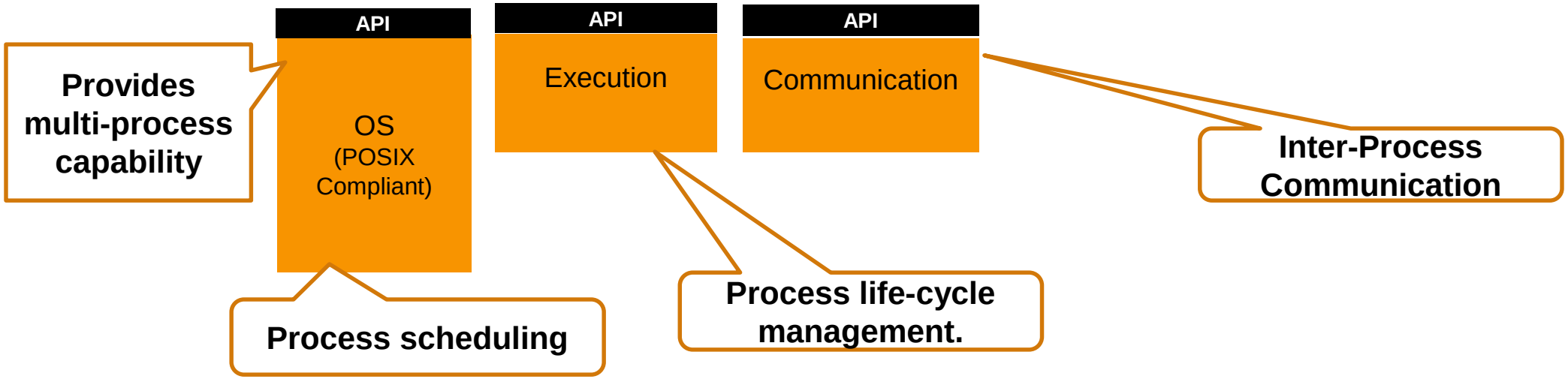




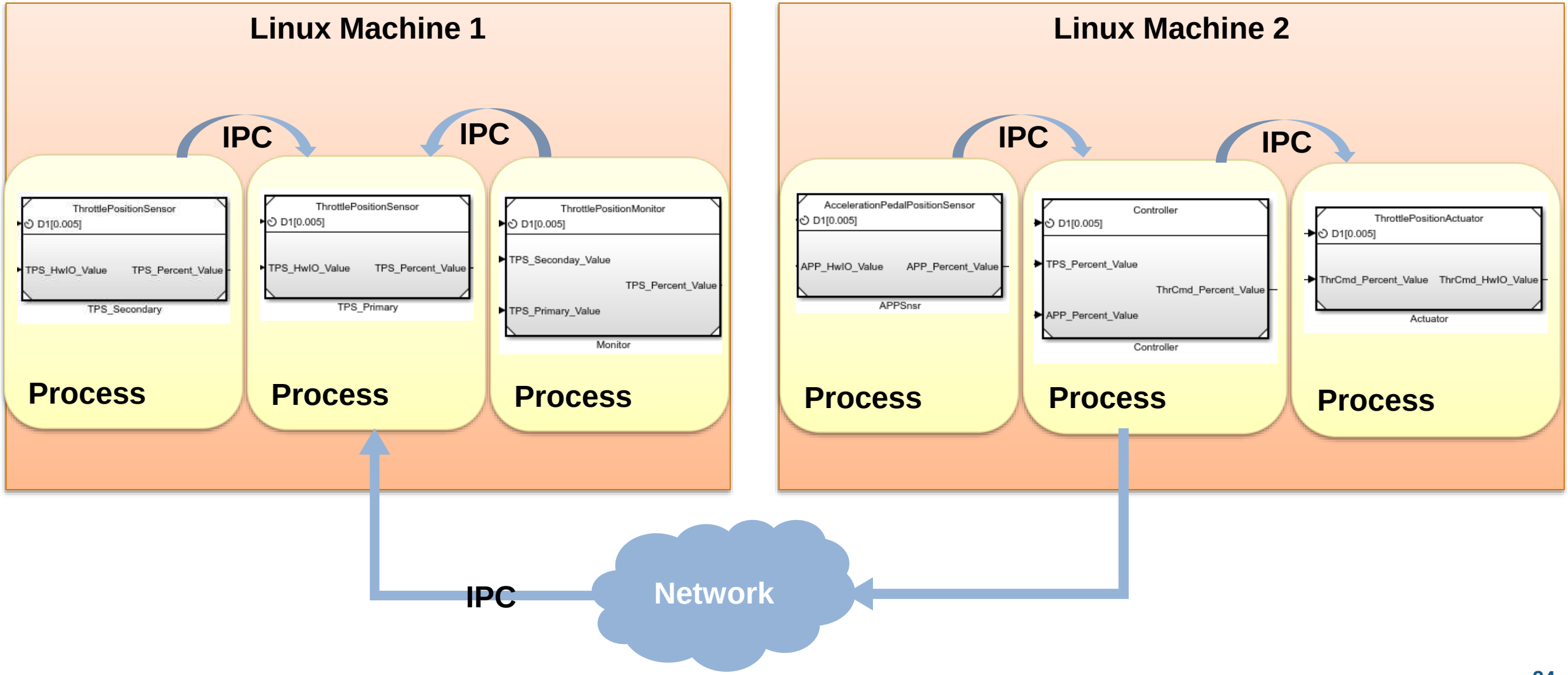
Notes: Each OS Process
- Corresponds to main() in C/C++ code
- Has own memory space & namespace
- Can be single or multi-threaded

Key Concept #1
Everything is a process .. as in “OS process”



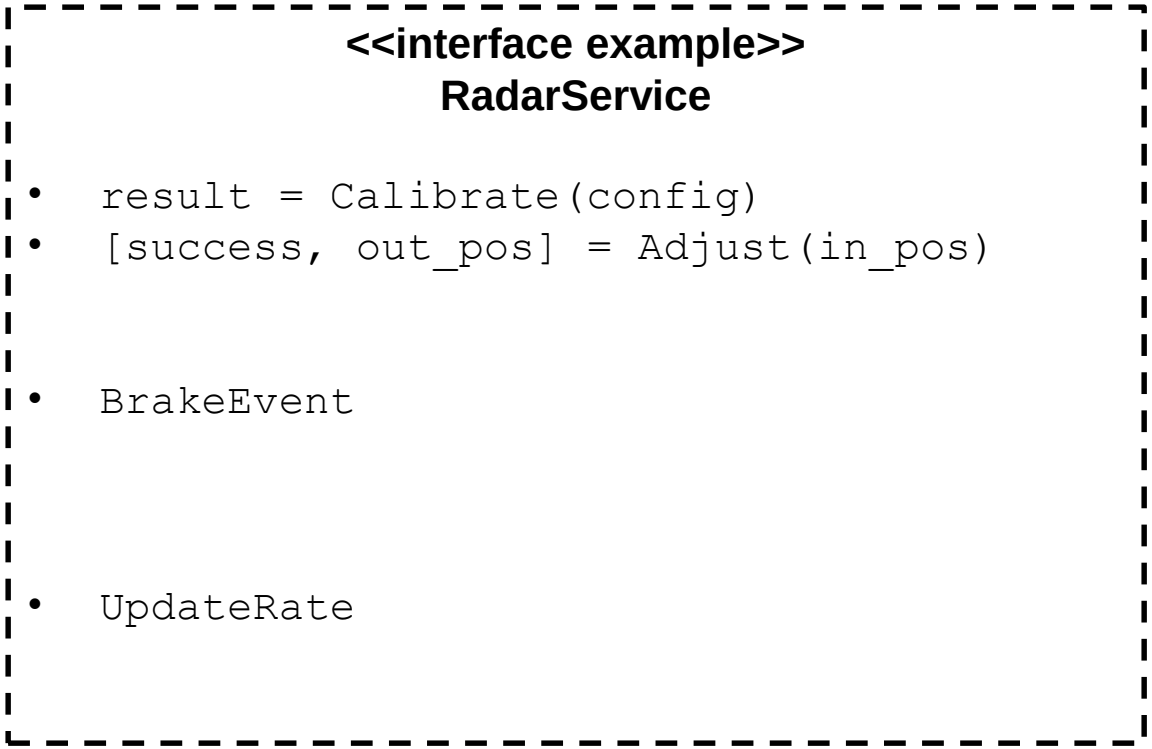


Key Concept #2
Service-oriented inter-process communication

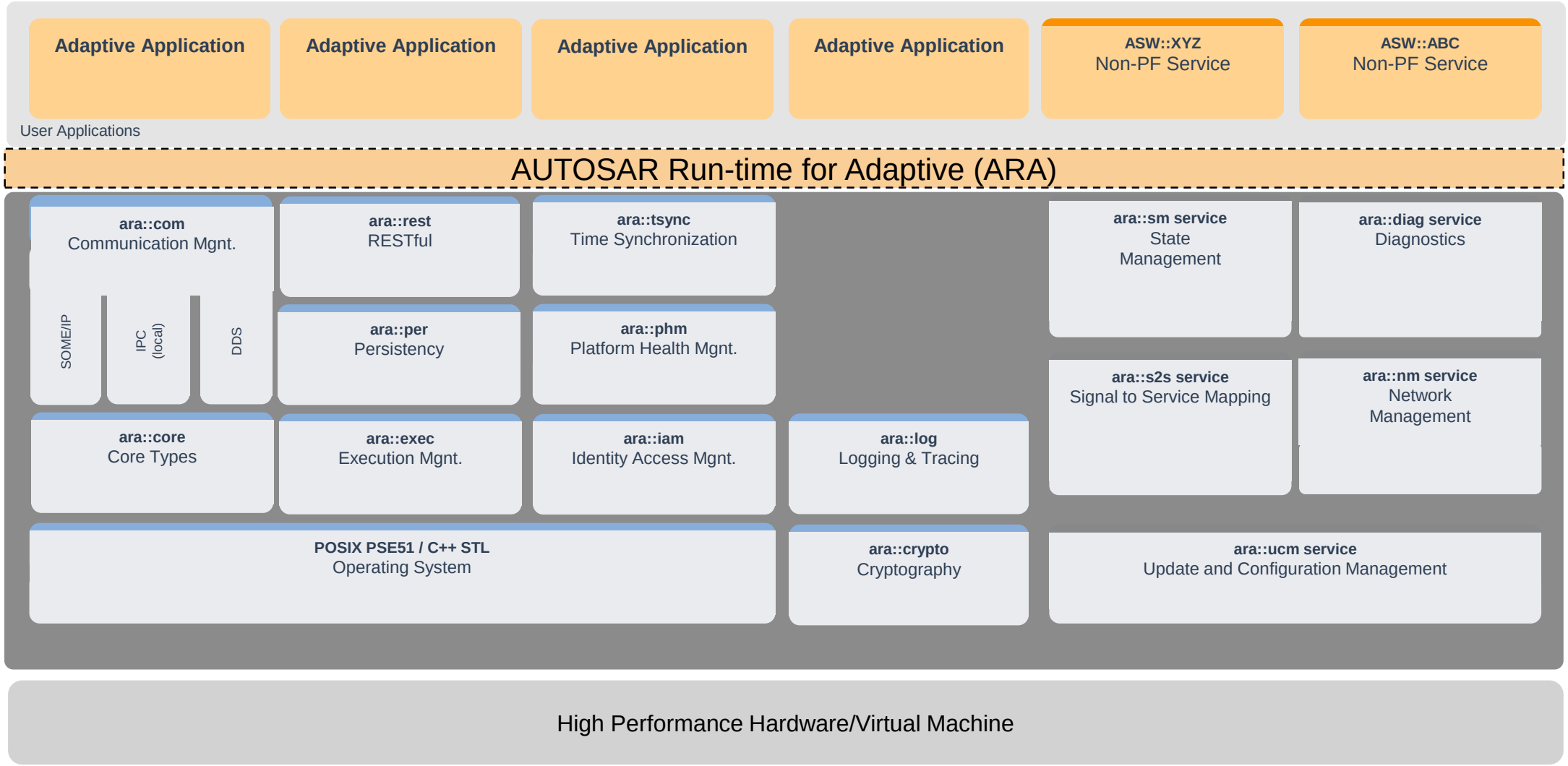


Key Concept #2
Service-oriented communication

- Service Interface can contain
  - Methods (Functions)
  - Events (Messages)
  - Fields (Data)



Key Concept #3: Everything is C++



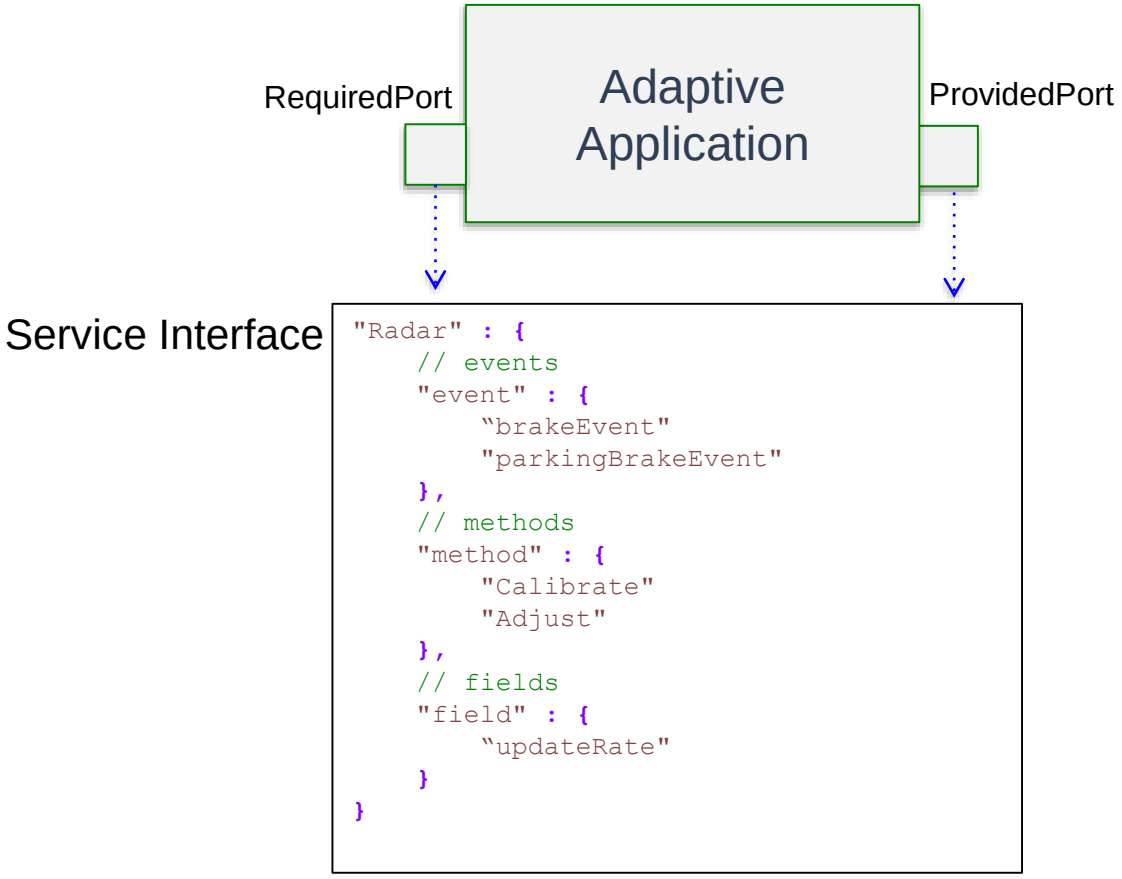
Motivation for Simulink to support Adaptive

- Simulink is heavily used for AUTOSAR Classic
- Customers have requested Simulink support for Adaptive platform

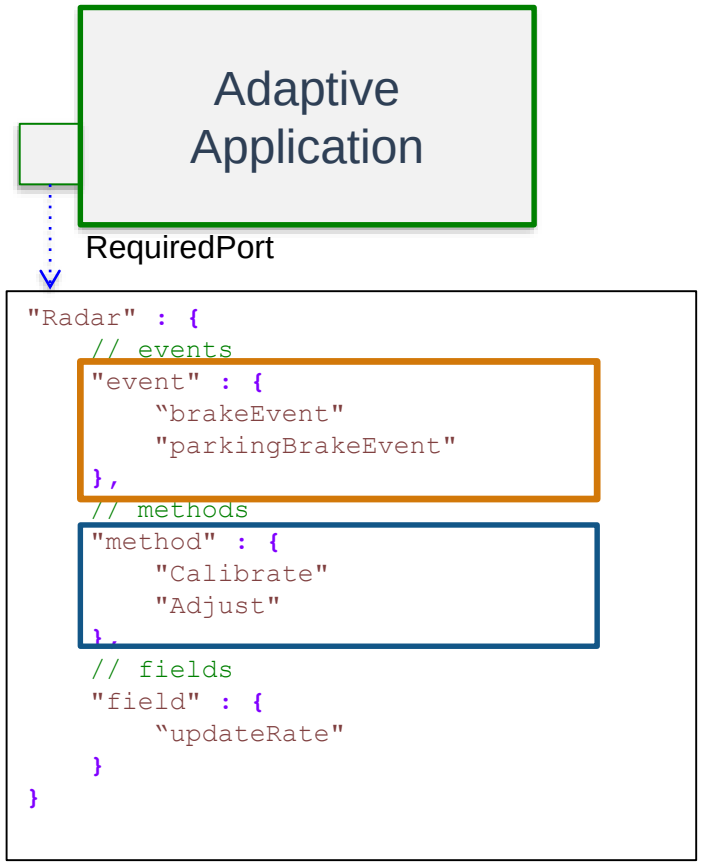
- Simulink supports service oriented modelling
- Embedded Coder generates C and C++ code

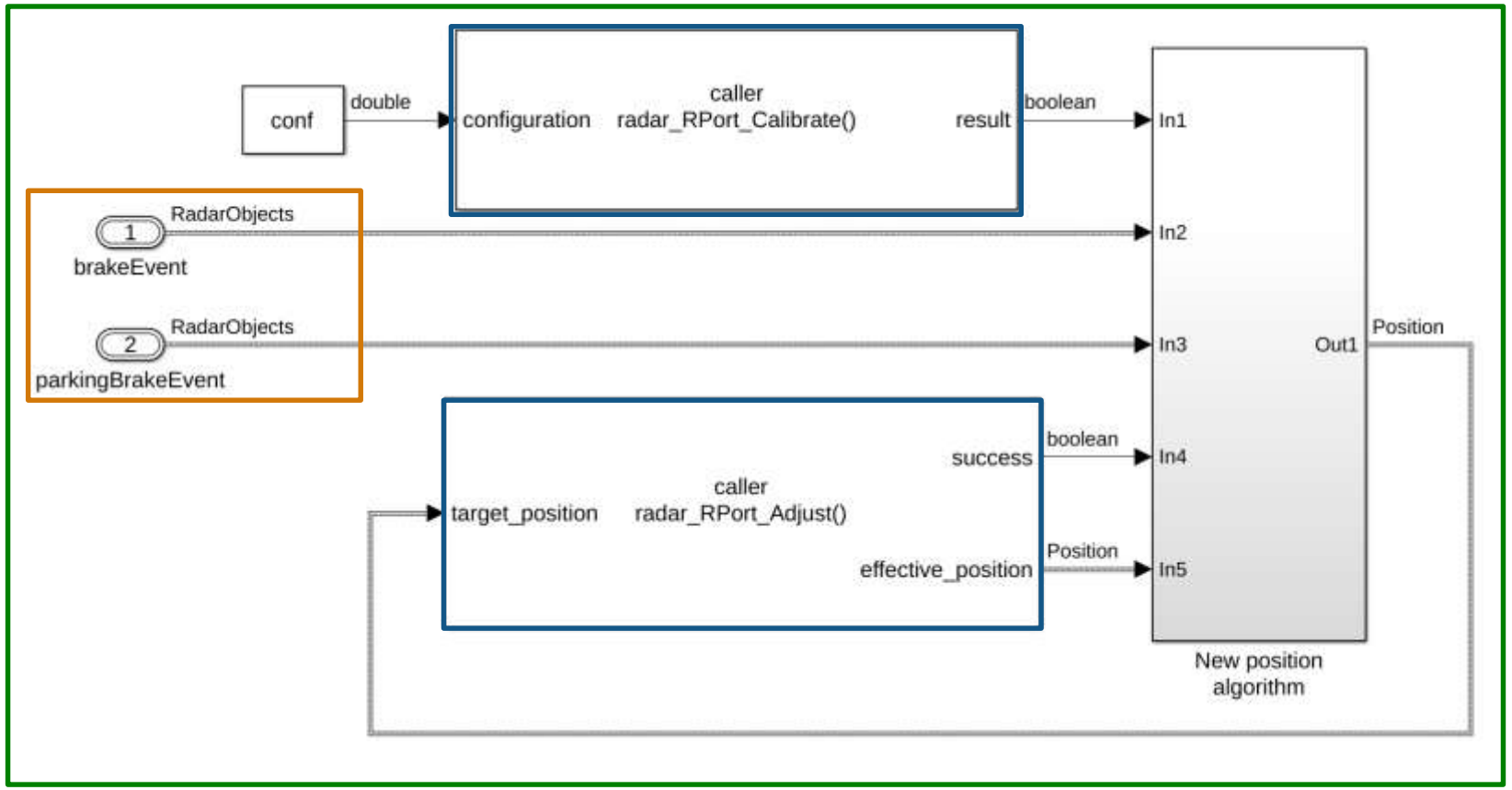
- MathWorks participates in the AUTOSAR standard development, including both Classic and Adaptive platforms

Adaptive SW Architecture Concepts

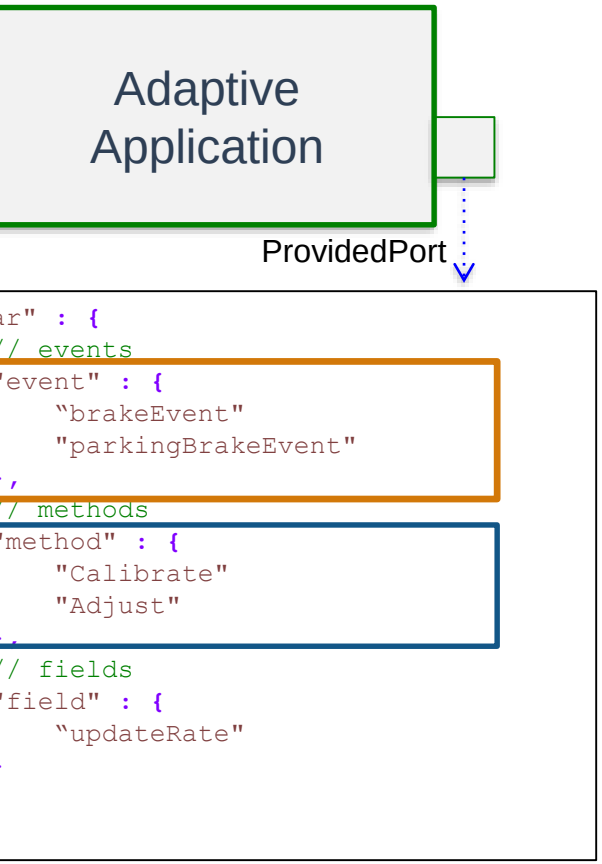


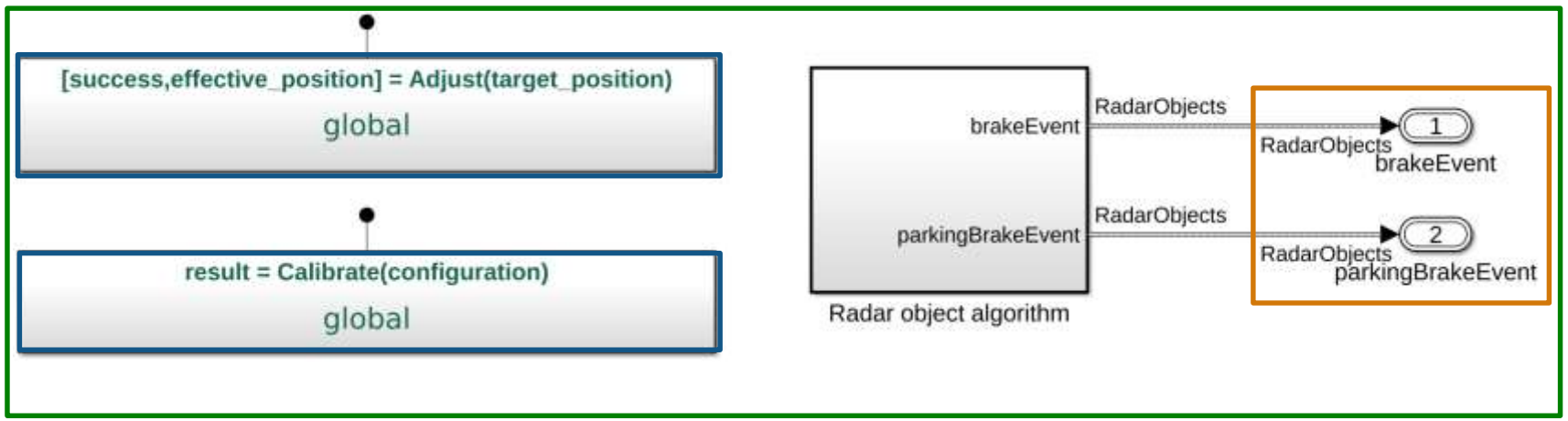
Mapping AUTOSAR AP Concepts to Simulink



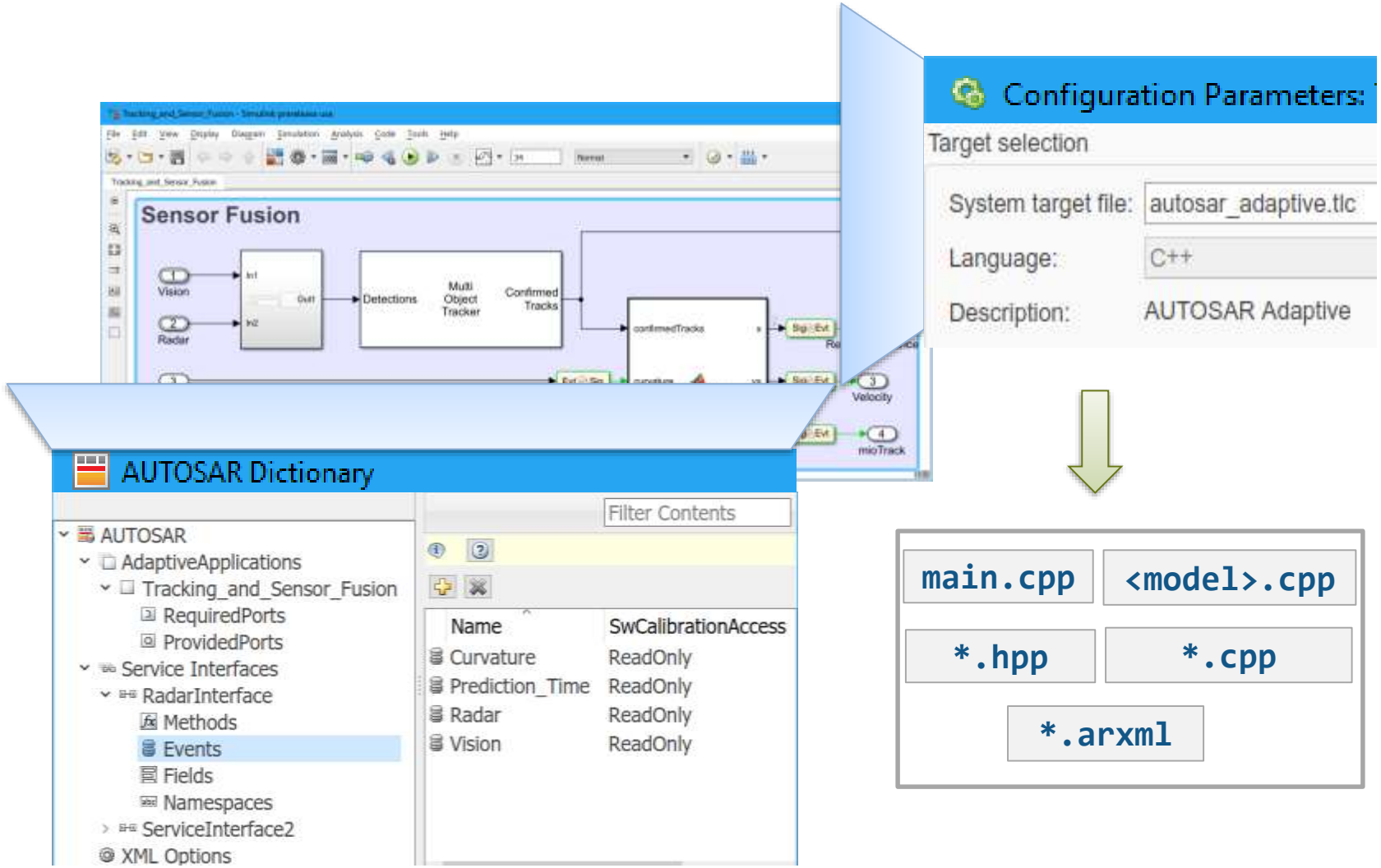


Mapping AUTOSAR AP Concepts to Simulink





Generate Production AUTOSAR Adaptive C++ Code



Out-of-box AUTOSAR support

1. Configure Model
   - ✓ Target
   - ✓ AUTOSAR Dictionary
2. Generate C++ code

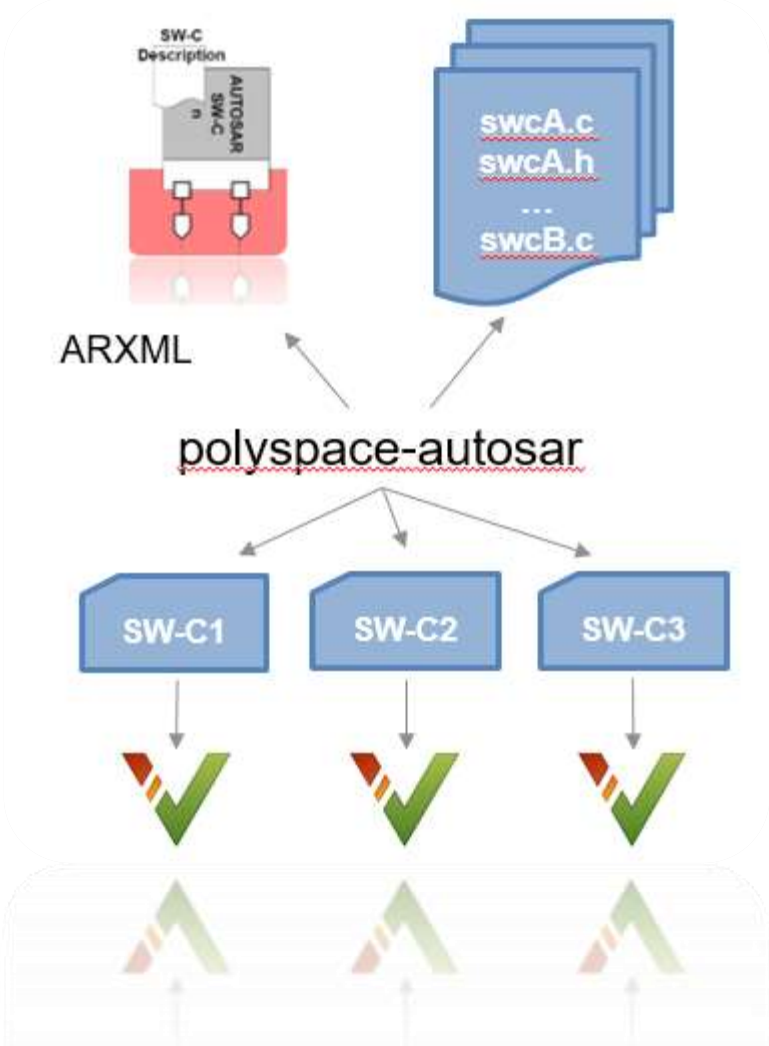
Develop Adaptive AUTOSAR Components

- autosar_LaneGuidance shipping demo

Agenda

- Introduction to AUTOSAR
- Simulink for Classic Platform
  - Automatic modeling and code generation
  - Simulation of AUTOSAR ECU software
  - Blocks for AUTOSAR Library routines
  - Importing and exporting AUTOSAR descriptions artifacts (ARXML files)
- Simulink for Adaptive Platform
  - A closer look at the Adaptive layers
  - Motivation for Simulink to support Adaptive
  - Mapping Adaptive platform to Simulink
  - Code Generation for Adaptive components
- Polyspace for AUTOSAR
- Additional Resources

Polyspace for AUTOSAR



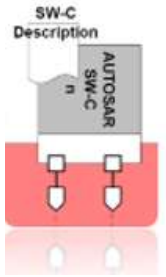
What if…?

- The communication between the software architect and developer is made easy
- An easy setup process for verification
- Setup only needs the ARXML and code implementations
- And also check for the Run-Time errors in the code

Check if…

- Implementation of Software Components Follow Specifications
- Edits to Specifications Impact
- Implementation for Run-time Errors and Mismatch with Specifications
- Implementation Against Specification Update

Polyspace and AUTOSAR

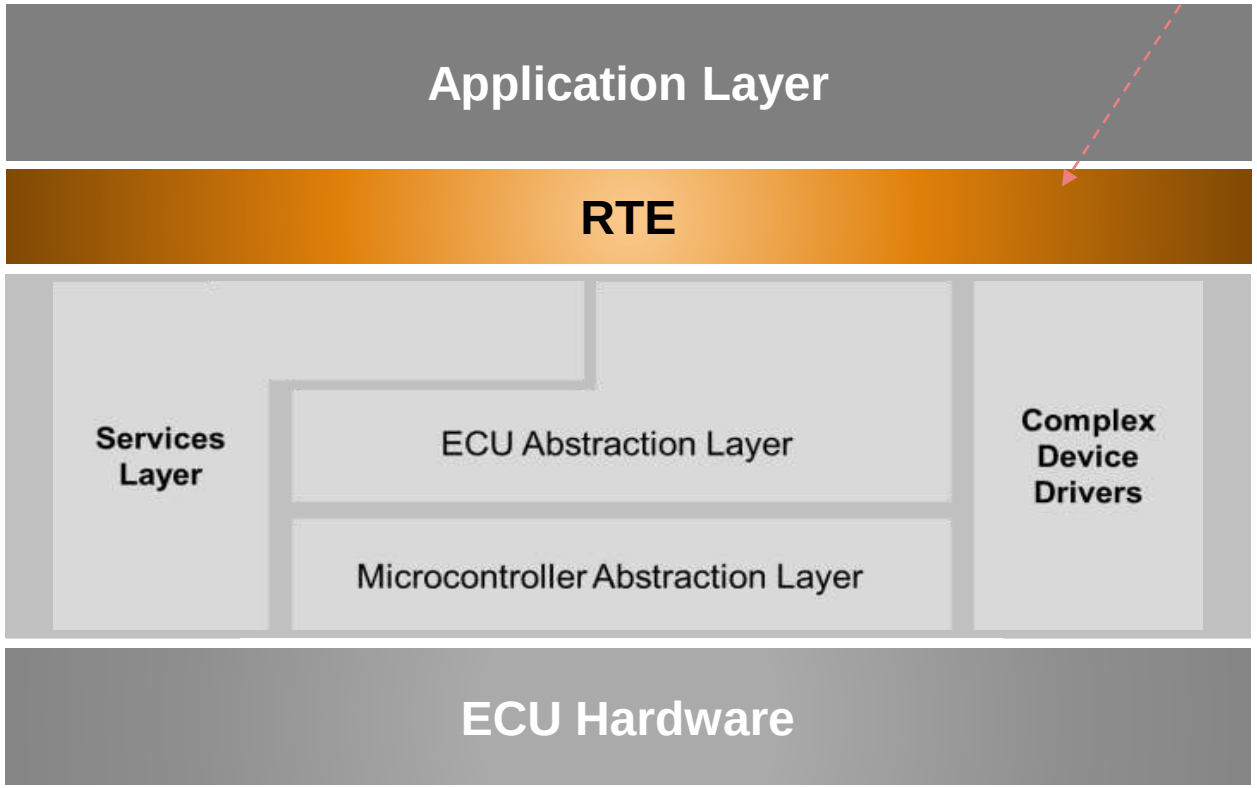


ARXML

ARXML provides specification of Application Layer and link with RTE

Polyspace verifies the match between code and ARXML

AUTOSAR architecture



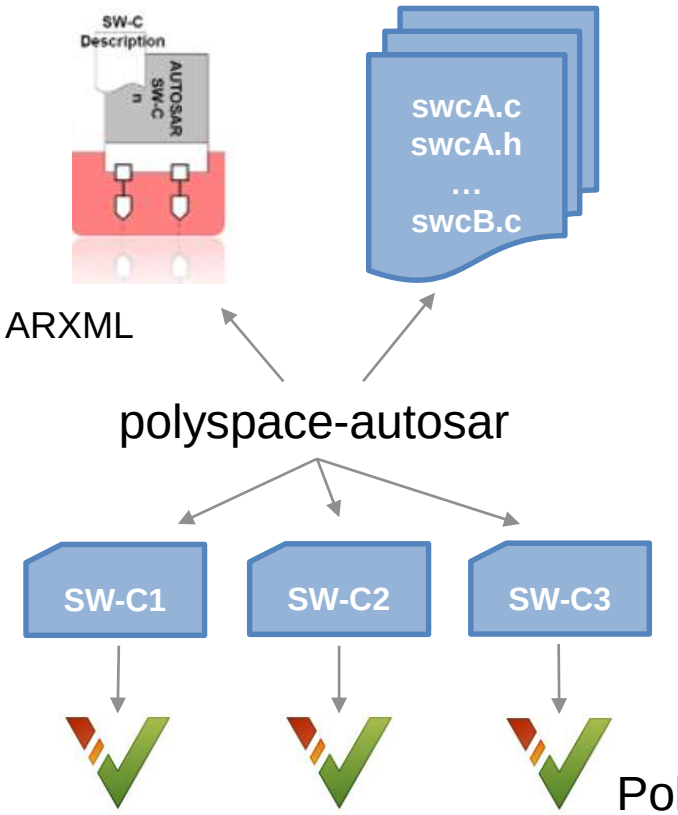
Polyspace verifies the Application Layer

Polyspace stubs the RTE Layer
RTE Layer not verified by "Polyspace for AUTOSAR"

Not verified by "Polyspace for AUTOSAR"

Polyspace for AUTOSAR(R2018a)

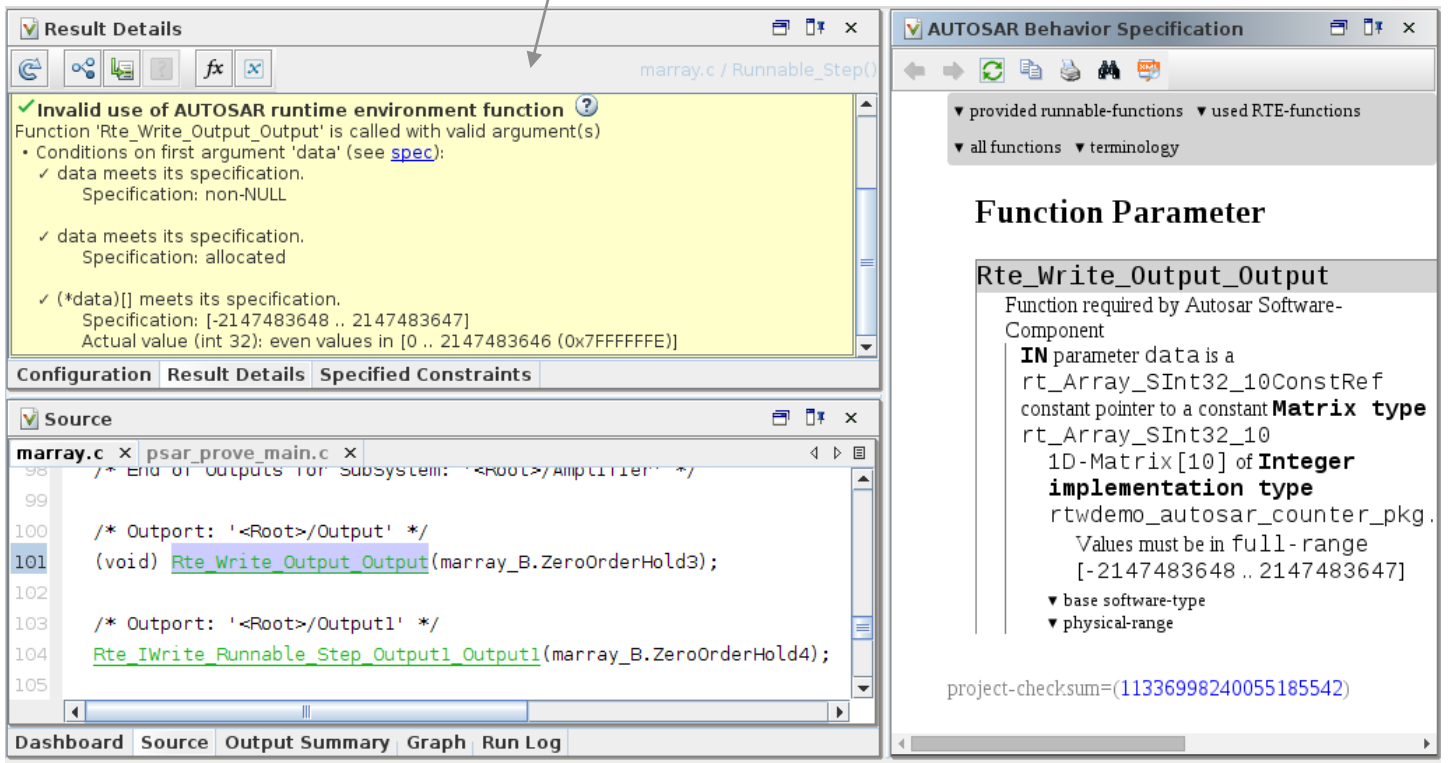
Use Polyspace to perform a sound unit static analysis of the components of an AUTOSAR software



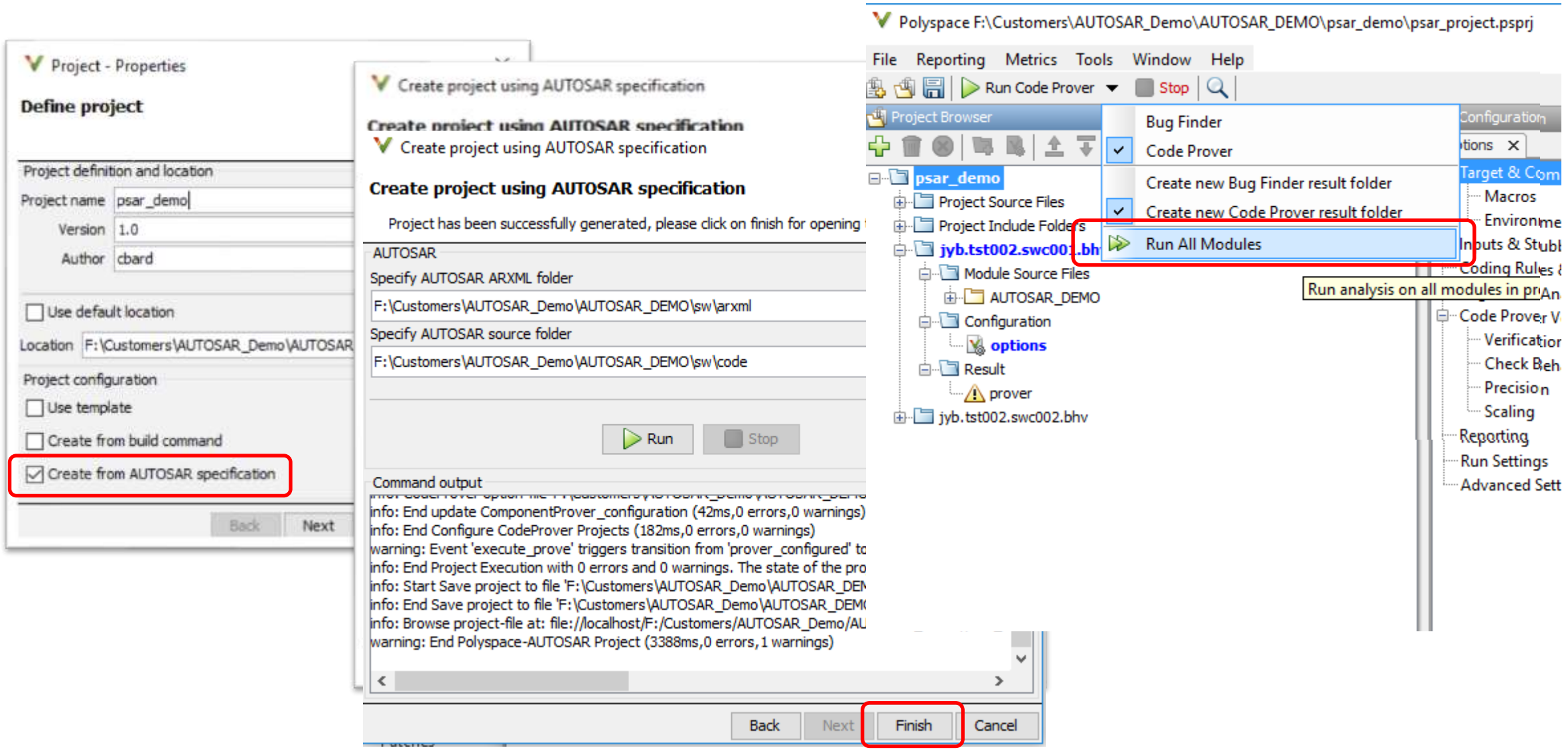
Polyspace code prover

New checks to prove that the code matches the specification

New view to detail the AUTOSAR specification



Polyspace for AUTOSAR: How do I launch from UI?



Polyspace for AUTOSAR

- Verify ARXML against Code Demo

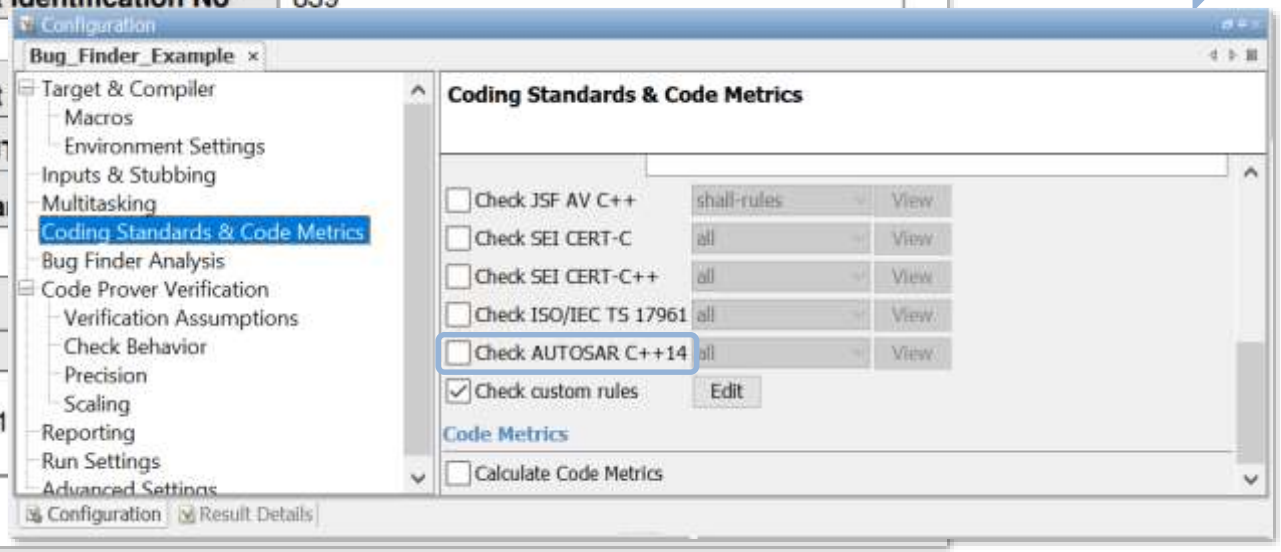
Polyspace checks AUTOSAR C++14 Rules

AUTOSAR
Guidelines for the use of the C++14 language in critical and safety-related systems
AUTOSAR AP Release 17-03

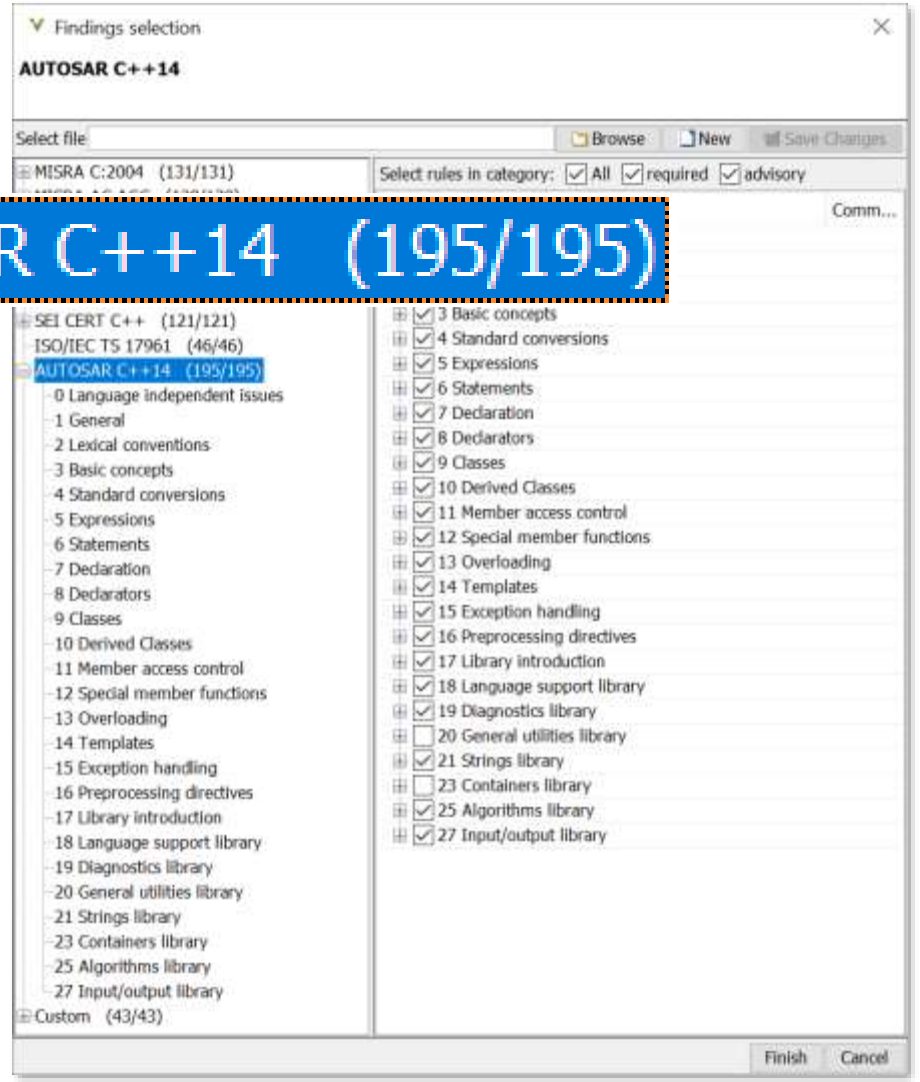
Document Title	Guidelines for the use of the C++14 language in critical and safety-related systems
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	839

Document
Part of AU
Part of Sta

Date
2017-03-31



AUTOSAR C++14 (195/195)



https://www.autosar.org/fileadmin/user_upload/standards/adaptive/17-03/AUTOSAR_RS_CPP14Guidelines.pdf

-autosar-cpp14

Benefits of using Polyspace for AUTOSAR

- Polyspace automatically modularizes analysis based on AUTOSAR components
- Polyspace detects mismatch between code and AUTOSAR XML spec
  - AUTOSAR runnable not implemented
  - Invalid result of AUTOSAR runnable implementation
  - Invalid use of AUTOSAR runtime environment function
- Prove absence of certain types of run-time errors in runnables (e.g. OverFlow, divByZero)

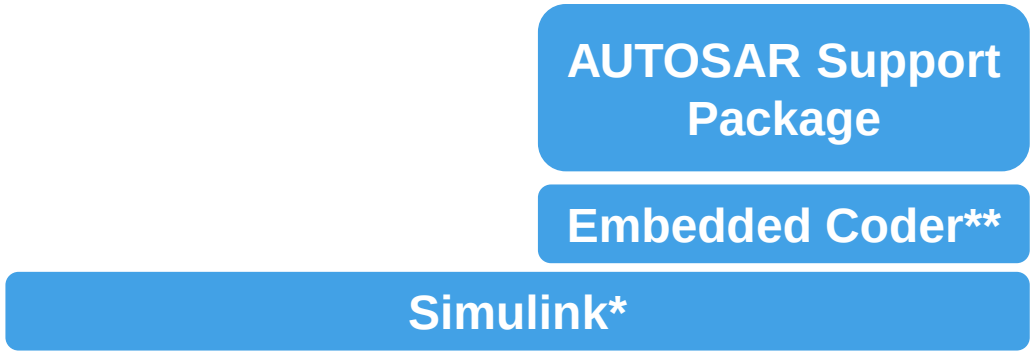
Summary

AUTOSAR Blockset :

- Model AUTOSAR Classic and Adaptive software components
- Simulate AUTOSAR compositions and ECUs
- Generate optimized AUTOSAR C/C++ code, roundtrip ARXML, and perform SIL and PIL verification (requires Embedded Coder®)
- Is well-suited for applications involving embedded production deployment
- Is key part of Model-Based Design by providing detailed specification of embedded software

AUTOSAR Support Transition

- R2018b and earlier



Required

AUTOSAR Blockset

- R2019a and later

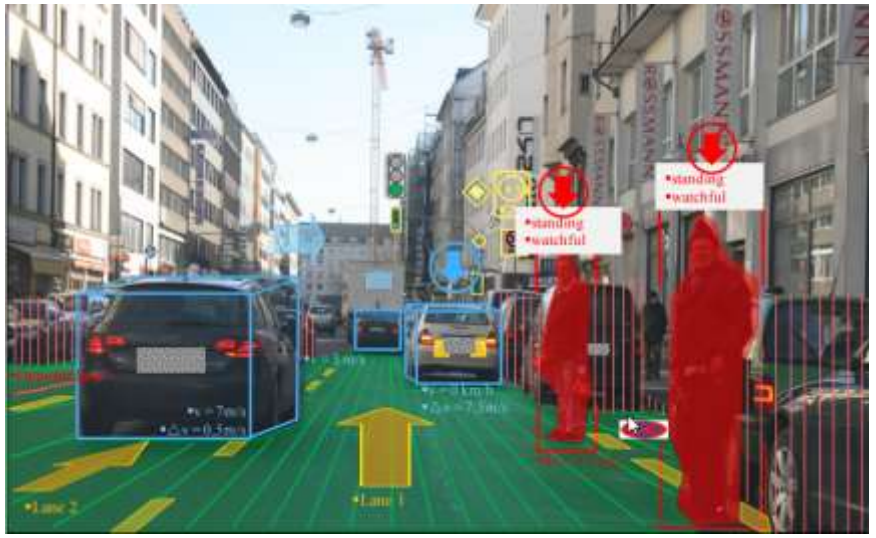
Required for sim

Required for code

*Requires MATLAB
**Requires MATLAB Coder and Simulink Coder

User Articles/Presentations

- BMW - Model-Based Software Development: And OEM's Perspective
- FCA Global Powertrain Controls - Leveraging MBD, auto-code generation and AUTOSAR to architect and implement an Engine Control Application for series production
- LG Chem - Developing AUTOSAR and ISO 26262 Compliant Software for a Hybrid Vehicle Battery Management System with Model-Based Design
- John Deere - Vertical AUTOSAR System Development at John Deere



To learn more, please visit AUTOSAR Blockset page

Come see us at our demo booth