MathWorks

12 Years of AUTOSAR
Enabling Innovation with Model-Based Design

AUTOSAR

Michael Fröstl – Pilot Engineering @ MathWorks Germany

MathWorks
AUTOMOTIVE CONFERENCE 2015

```
switch(braindump)
{
```

case ‘AUTOSAR Acronym’:

AUTOSAR

means

① **AUTO**mobile **S**earch **A**nd **R**escue

② **AUT**hentic **S**portscar **A**spect **R**atio

③ **AUT**omotive **O**pen **S**ystem **Ar**chitecture

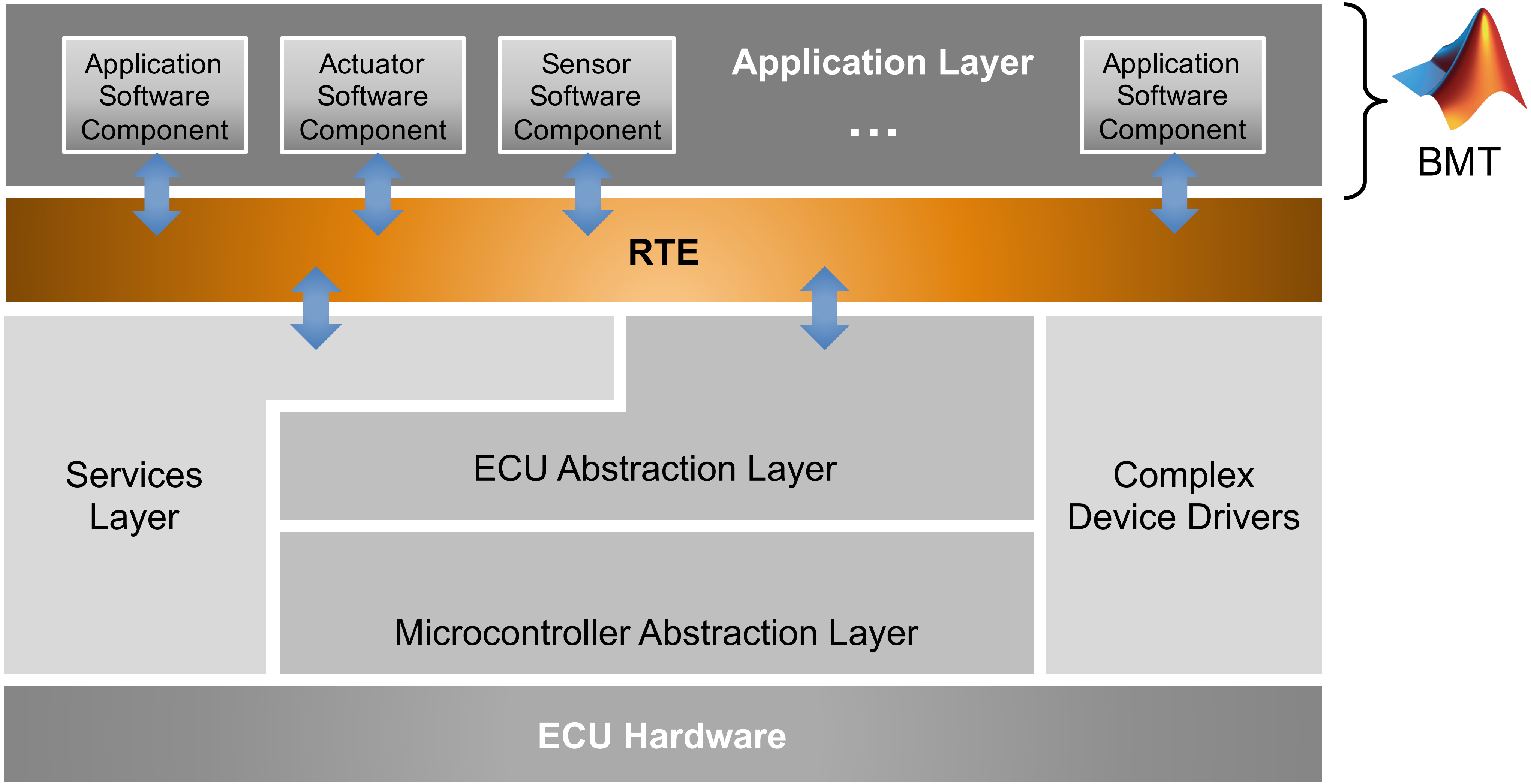
④ **AUT**ocar **O**ccupant **S**pecific **A**version **R**ate

⑤ **AUTO**recovery **S**oftware **A**bstraction **R**eloaded

```
default :
  printf(“Wrong session?”);
}
```

AUTOSAR_Overview();

AUTOSAR – 3-layered Architecture

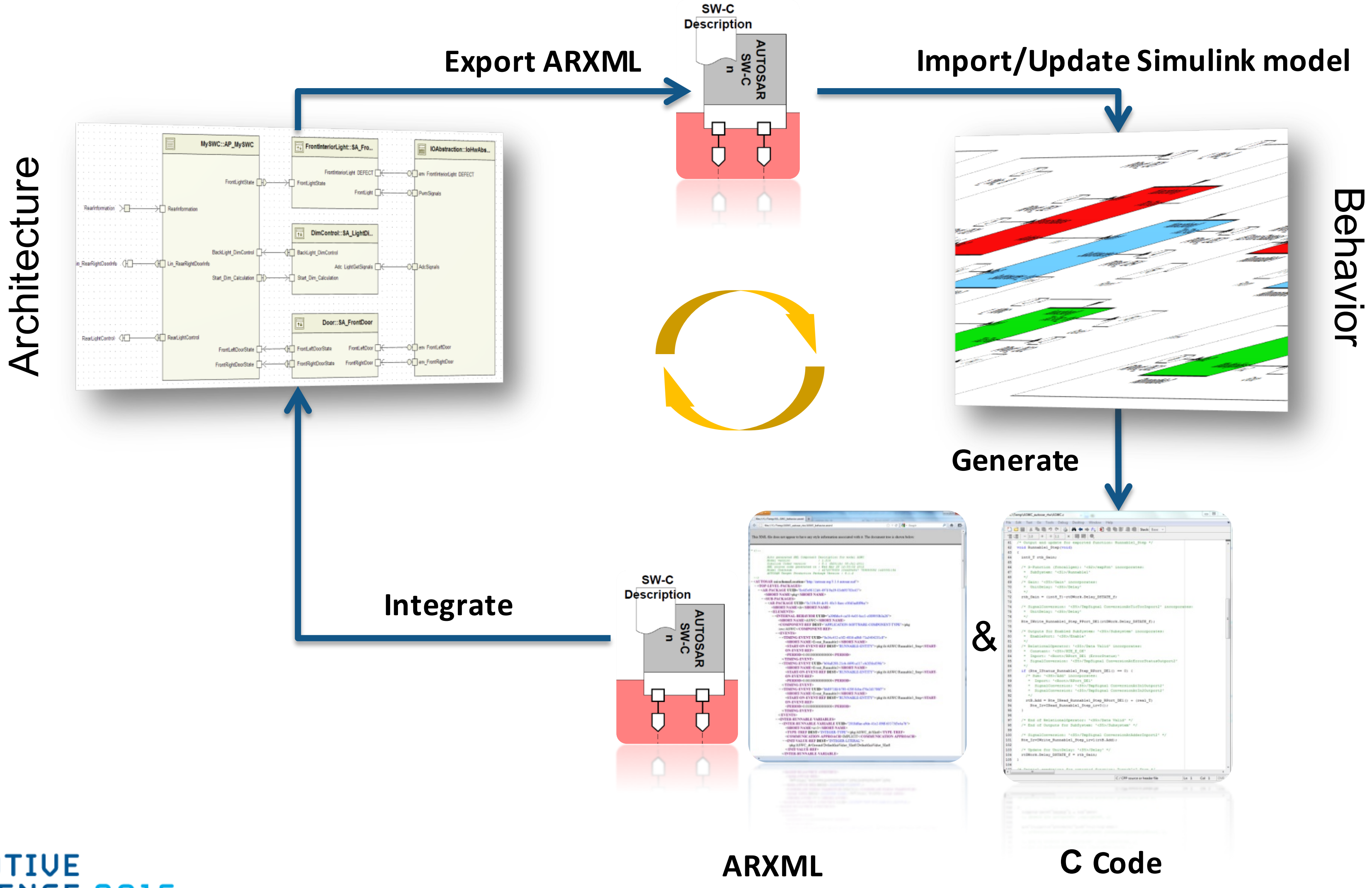

Application Software Component
Actuator Software Component
Sensor Software Component
Application Layer
...
Application Software Component
BMT
RTE
Services Layer
ECU Abstraction Layer
Microcontroller Abstraction Layer
Complex Device Drivers
ECU Hardware

MathWorks AUTOSAR Approach

No separate AUTOSAR Blockset needed	• Code-generation through Mapping
AUTOSAR Software Component Approach with Simulink	• Simulink for developing behavior • Import and Export of SW Component Description Files (ARXML)
Simultaneous generation of C-code and ARXML-Files	• Consistency between C code and ARXML SW-C description files
AUTOSAR Support Package for Embedded Coder	• Available via web download* • Allows more frequent updates and fixes

* http://www.mathworks.com/hardware-support/autosar.html

Support for AUTOSAR Workflows



Export ARXML

SW-C Description

AUTOSAR SW-C

Import/Update Simulink model

Architecture

Behavior

Generate

Integrate

SW-C Description

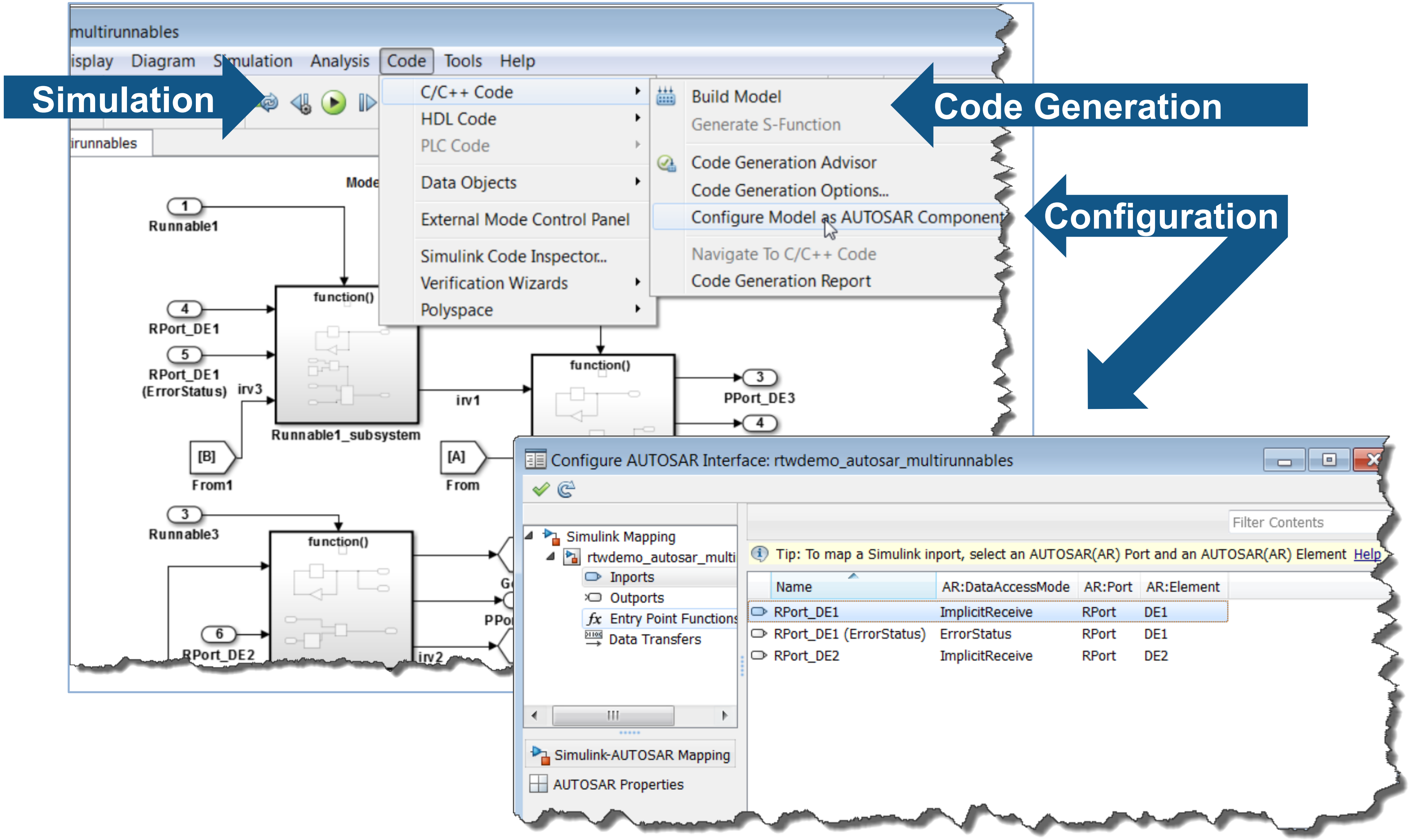
AUTOSAR SW-C

&

ARXML

C Code

Capabilities



Simulation

Code Generation

Configuration

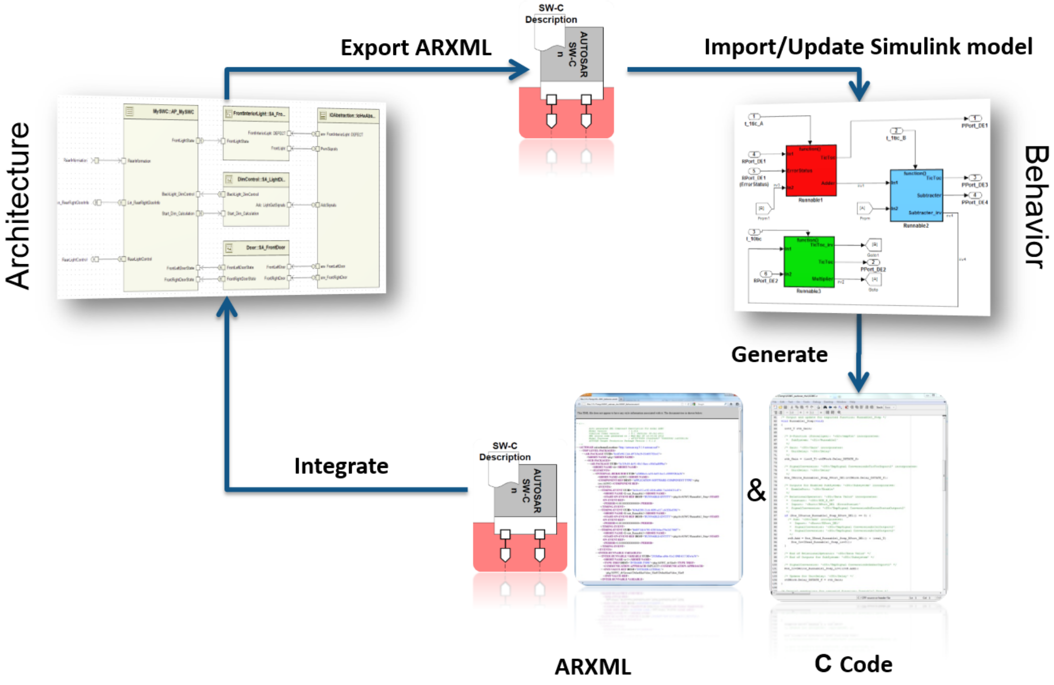
Tip: To map a Simulink inport, select an AUTOSAR(AR) Port and an AUTOSAR(AR) Element Help

Name	AR:DataAccessMode	AR:Port	AR:Element
RPort_DE1	ImplicitReceive	RPort	DE1
RPort_DE1 (ErrorStatus)	ErrorStatus	RPort	DE1
RPort_DE2	ImplicitReceive	RPort	DE2

Getting Started

- **Bottom-Up Approach**
  Start with an existing Simulink model

- **Top-Down Approach**
  Start with ARXML files containing AUTOSAR Component descriptions



```
switch(topics)
{
```

case ‘AUTOSAR – Top 5’:

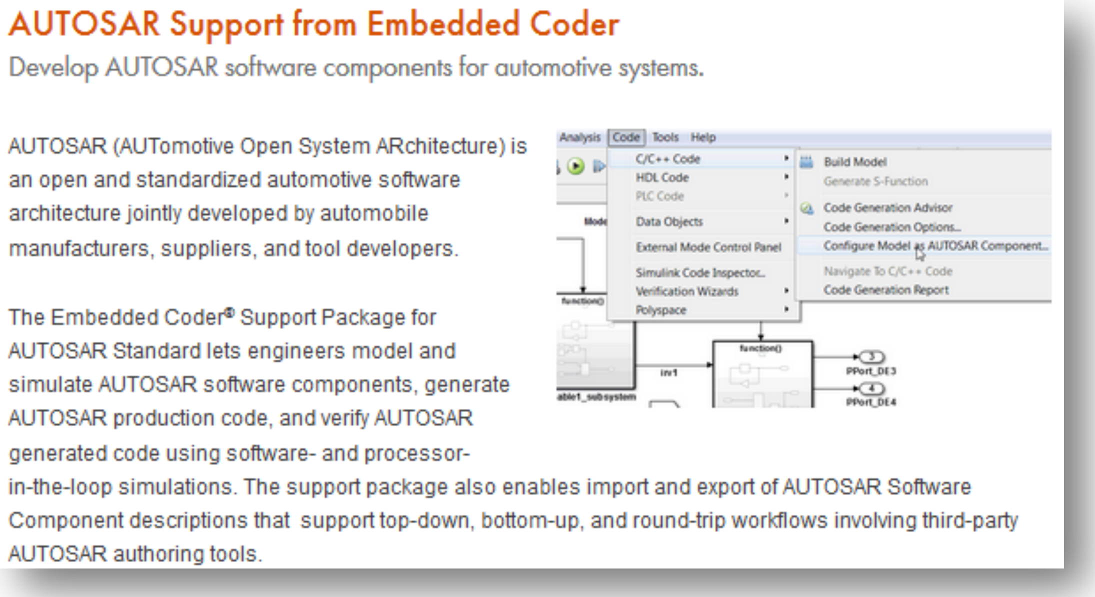
⑤

Embedded Coder® Support Package for AUTOSAR Standard

Embedded Coder® Support Package for AUTOSAR Standard

Embedded Coder ® add-on support for the AUTOSAR standard

- Flexible infrastructure to introduce important new capabilities, also in-between half-yearly MathWorks Release cycle
- Perform a wide range of AUTOSAR-related workflows in Simulink®, including:
  - Create and modify an AUTOSAR configuration for a model
  - Model AUTOSAR elements
  - Generate ARXML and AUTOSAR-compatible C code from a model



AUTOSAR Support from Embedded Coder

Develop AUTOSAR software components for automotive systems.

AUTOSAR (AUTomotive Open System ARchitecture) is an open and standardized automotive software architecture jointly developed by automobile manufacturers, suppliers, and tool developers.

The Embedded Coder® Support Package for AUTOSAR Standard lets engineers model and simulate AUTOSAR software components, generate AUTOSAR production code, and verify AUTOSAR generated code using software- and processor-in-the-loop simulations. The support package also enables import and export of AUTOSAR Software Component descriptions that support top-down, bottom-up, and round-trip workflows involving third-party AUTOSAR authoring tools.

http://de.mathworks.com/hardware-support/autosar.html

④

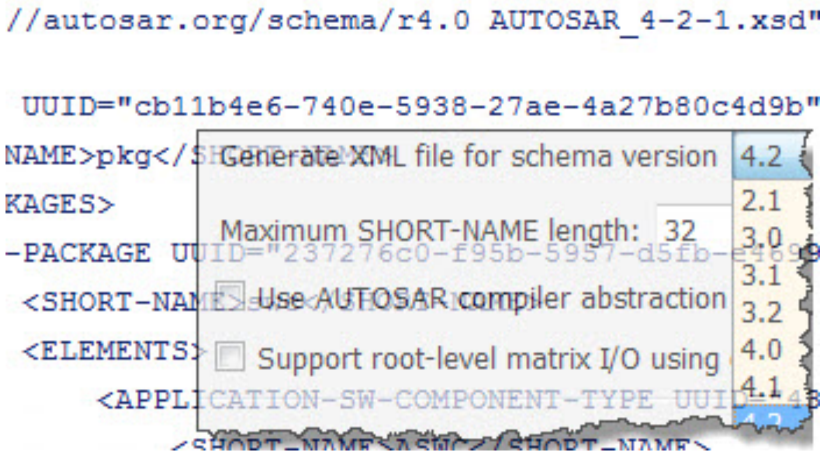
AUTOSAR 4.1.3 / 4.2.1

AUTOSAR 4.1.3 / 4.2.1

Seamless support for AUTOSAR Release 4.2.1 and 4.1.3 schema

- Import detects AUTOSAR 4.2.1 release from ARXML file
- User selects AUTOSAR release from configuration set options for code generation and ARXML export
- AUTOSAR 4.1+ features
  - PRPortPrototype
  - InitEvent
  - …

R2015b



```
//autosar.org/schema/r4.0 AUTOSAR_4-2-1.xsd"

 UUID="cb11b4e6-740e-5938-27ae-4a27b80c4d9b"
NAME>pkg</S
KAGES>
-PACKAGE UUID="237276c0-f95b-5957-d5fb-e469
 <SHORT-NAME>
 <ELEMENTS>
     <APPLICATION-SW-COMPONENT-TYPE UUID="43
        <SHORT-NAME>ASWC</SHORT-NAME>
```

Generate XML file for schema version 4.2

Maximum SHORT-NAME length: 32

□ Use AUTOSAR compiler abstraction

□ Support root-level matrix I/O using

③

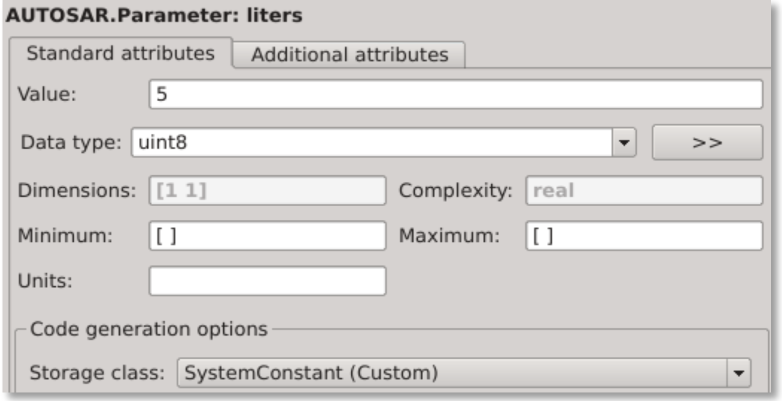
AUTOSAR Variant Handling

AUTOSAR Variant Handling

R2015a

Model AUTOSAR variants in Simulink

- VariationPointProxy with
  - Condition Access – `Simulink Variant Subsystem`
  - Value Access – `AUTOSAR.Parameter` with CSC `System Constant`
    - VariationPointProxy objects automatically generated
    - System constant definitions generated in separate ARXML file



AUTOSAR.Parameter: liters

Standard attributes | Additional attributes

Value: 5

Data type: uint8

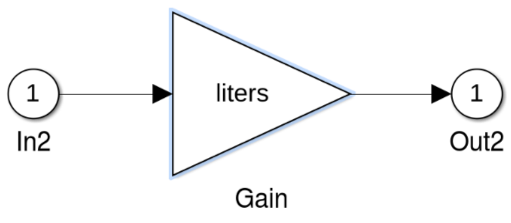
Dimensions: [1 1] Complexity: real

Minimum: [ ] Maximum: [ ]

Units:

Code generation options

Storage class: SystemContant (Custom)



```
<VARIATION-POINT-PROXYS>
    <VARIATION-POINT-PROXY UUID="10594079-04
        <SHORT-NAME>vpp_liters</SHORT-NAME>
        <VALUE-ACCESS BINDING-TIME="PRE-COMP
    </VARIATION-POINT-PROXY>
</VARIATION-POINT-PROXYS>
```

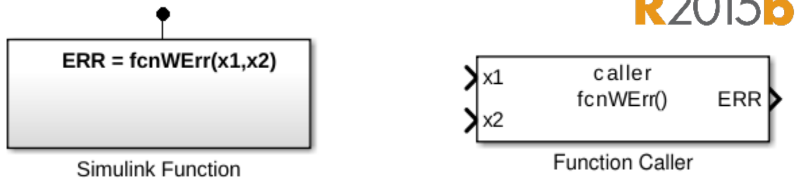
```
/* Gain: '<Root>/Gain' */
if (Rte_SysCon_vpp_liters > 7) {
  tmp = MAX_uint8_T;
} else {
  tmp = (uint8_T)(Rte_SysCon_vpp_liters << 5);
}
```

②

AUTOSAR Client-Server Semantics

AUTOSAR Client-Server Semantics

R2015b



Leverage Simulink Functions for AUTOSAR Client/Server

- ARXML import and update support for AUTOSAR Client/Server
  - AUTOSAR client port and operation represented as Function Caller
  - AUTOSAR server runnable represented as Simulink Function
- Use AUTOSAR APPLICATION-ERROR status for C/S communication
  - Helps to Communicate with AUTOSAR basic software that use error status, e.g. Diagnostic Event Manager (DEM)

```
Std_ReturnType fcnWErr(int8 x1, int8 x2)
{
   if (uh_oh) {
      return RTE_E_NOT_OK;
   }
   ..
   return RTE_E_OK;
}
```

```
<POSSIBLE-ERRORS>
   <APPLICATION-ERROR>
      <SHORT-NAME>E_OK</SHORT-NAME>
      <ERROR-CODE>0</ERROR-CODE>
   </APPLICATION-ERROR>
   <APPLICATION-ERROR>
      <SHORT-NAME>E_NOT_OK</SHORT-NAME>
      <ERROR-CODE>1</ERROR-CODE>
   </APPLICATION-ERROR>
</POSSIBLE-ERRORS>
```

①

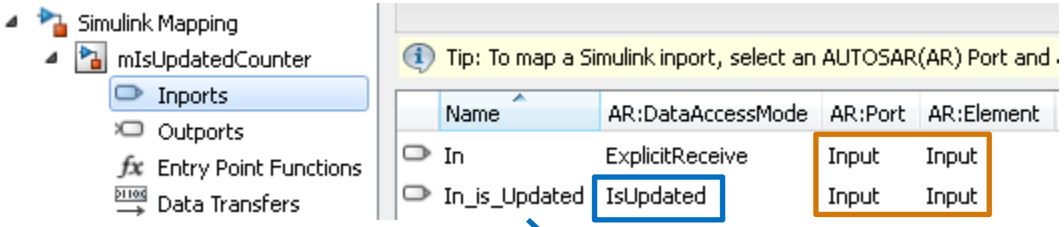
Advanced AUTOSAR APIs

Advanced AUTOSAR APIs

R2015b

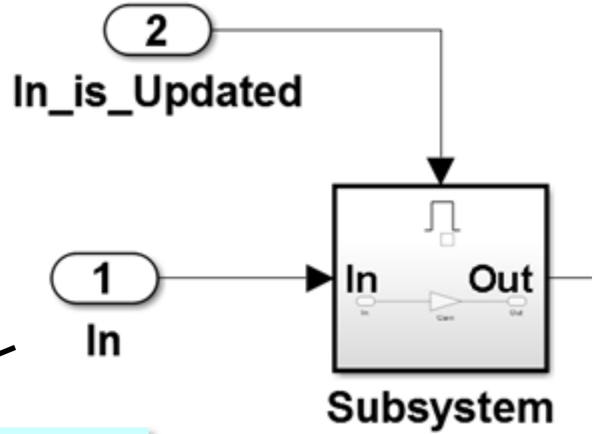
Simulink modeling and ARXML roundtrip support for

- RTE APIs
  - Conditional Rte_Read / Rte_Write
  - Rte_IsUpdated
  - Rte_Invalidate
- Asynchronous NvM Service calls
  - NvM_WriteBlock, NvM_ReadBlock, …
- E2E wrapper
- MFX / MFL / IFX / IFL Library Routines
- ReferenceBase Support
- …



Tip: To map a Simulink inport, select an AUTOSAR(AR) Port and

Name	AR:DataAccessMode	AR:Port	AR:Element
In	ExplicitReceive	Input	Input
In_is_Updated	IsUpdated	Input	Input



```
if (Rte_IsUpdated_Input_Input()){
  Rte_Read_Input_Input(&tmp);
  ...
}
```

```
<NONQUEUED-RECEIVER-COM-SPEC>
  ...
  <ENABLE-UPDATE>true</ENABLE-UPDATE>
```

case ‘AUTOSAR – Best Practices’:

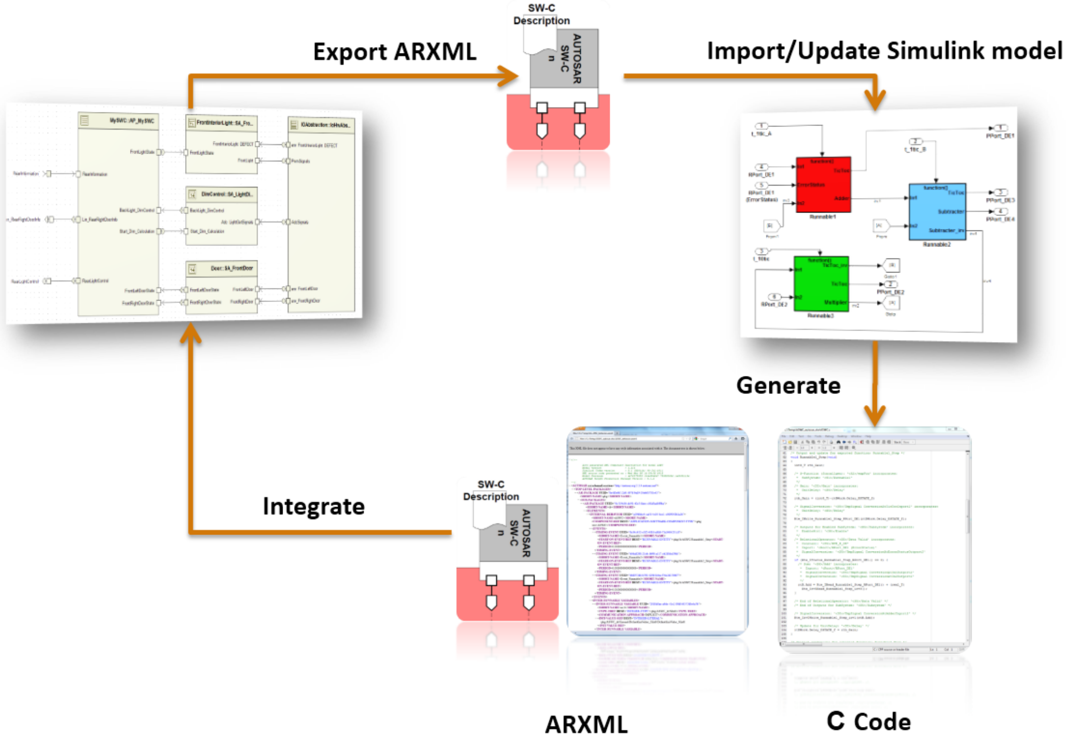
①

Use one AUTOSAR workflow

#1 Use one AUTOSAR workflow

- Select top-down or bottom-up approach
- Round-trip works best with one clear owner of data

- Select tools that best support your workflow and AUTOSAR concepts
- Select simplest approach for applying AUTOSAR configuration to your Simulink model

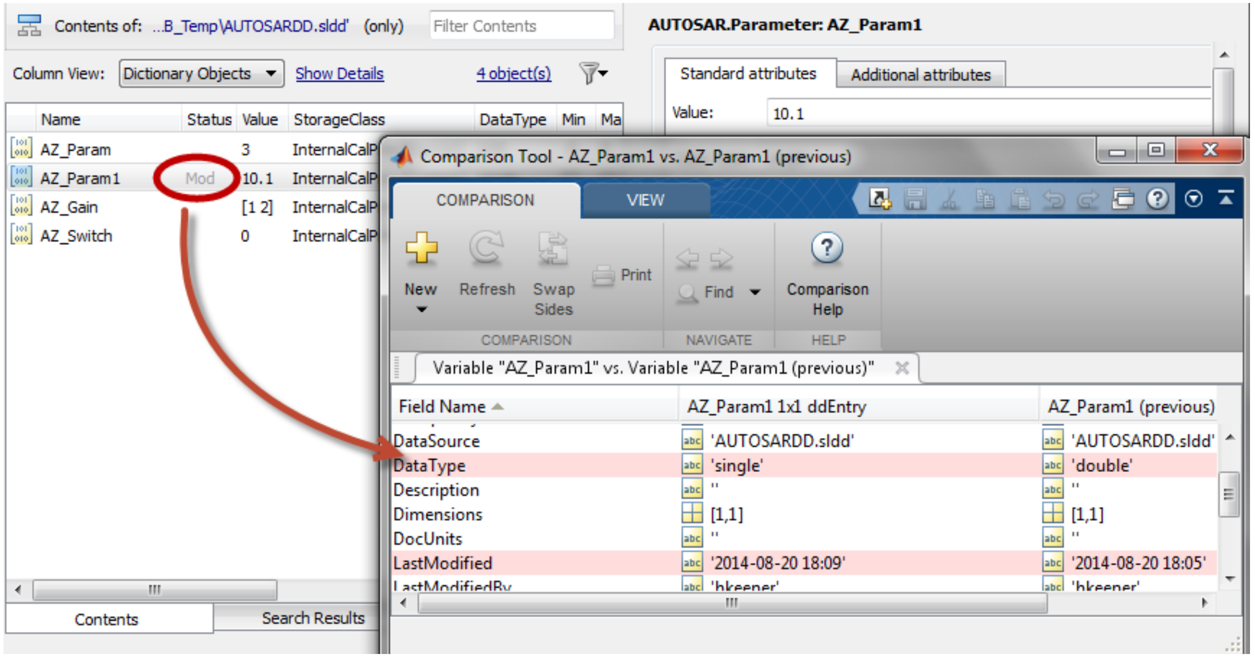


②

Decide data management

#2 Decide data management

- Will Simulink or AUTOSAR tools manage data?
- Will projects or teams define and manage data?
- How will change management be handled?



Contents of: ...B_Temp\AUTOSARDD.sldd' (only)

Column View: Dictionary Objects

Name	Status	Value	StorageClass	DataType	Min	Ma
AZ_Param		3	InternalCalP			
AZ_Param1	Mod	10.1	InternalCalP			
AZ_Gain		[1 2]	InternalCalP			
AZ_Switch		0	InternalCalP			

AUTOSAR.Parameter: AZ_Param1

Value: 10.1

Comparison Tool - AZ_Param1 vs. AZ_Param1 (previous)

Variable "AZ_Param1" vs. Variable "AZ_Param1 (previous)"

Field Name	AZ_Param1 1x1 ddEntry	AZ_Param1 (previous)
DataSource	'AUTOSARDD.sldd'	'AUTOSARDD.sldd'
DataType	'single'	'double'
Description	''	''
Dimensions	[1,1]	[1,1]
DocUnits	''	''
LastModified	'2014-08-20 18:09'	'2014-08-20 18:05'
LastModifiedBy	'hkeener'	'hkeener'

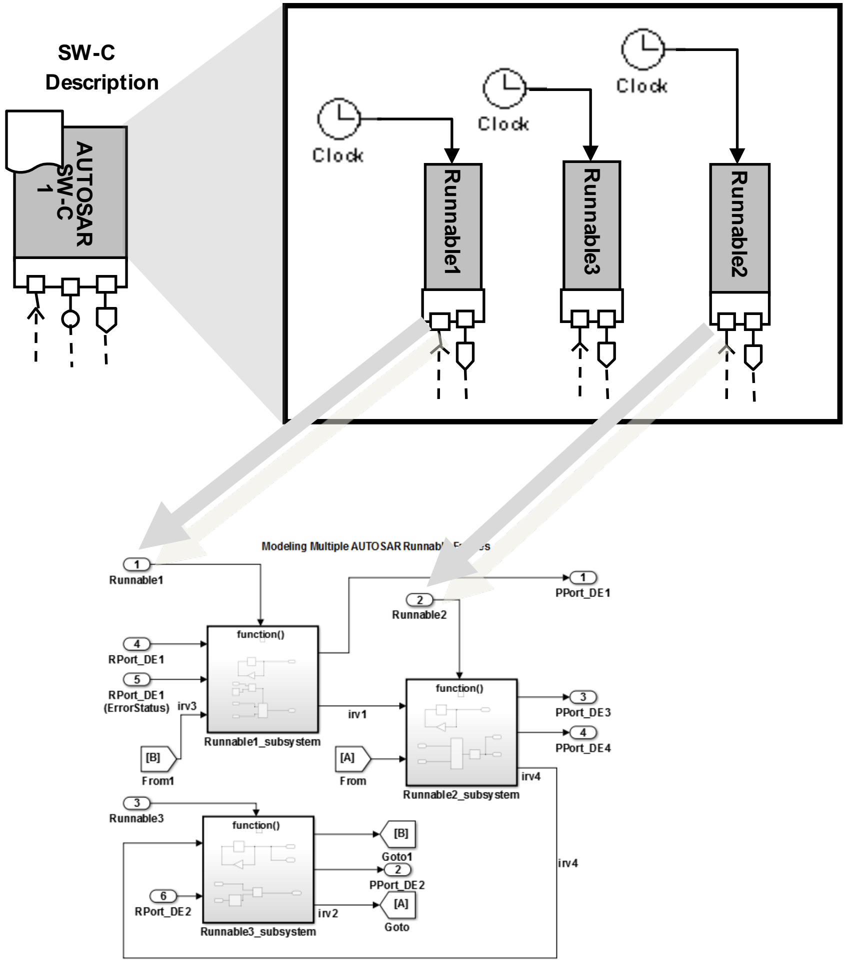
③

Establish modeling standards

#3 Establish modeling standards

- For Simulink and AUTOSAR

- Base it on your workflow and data management
- Use Simulink Model Advisor to enforce modeling style early in model development



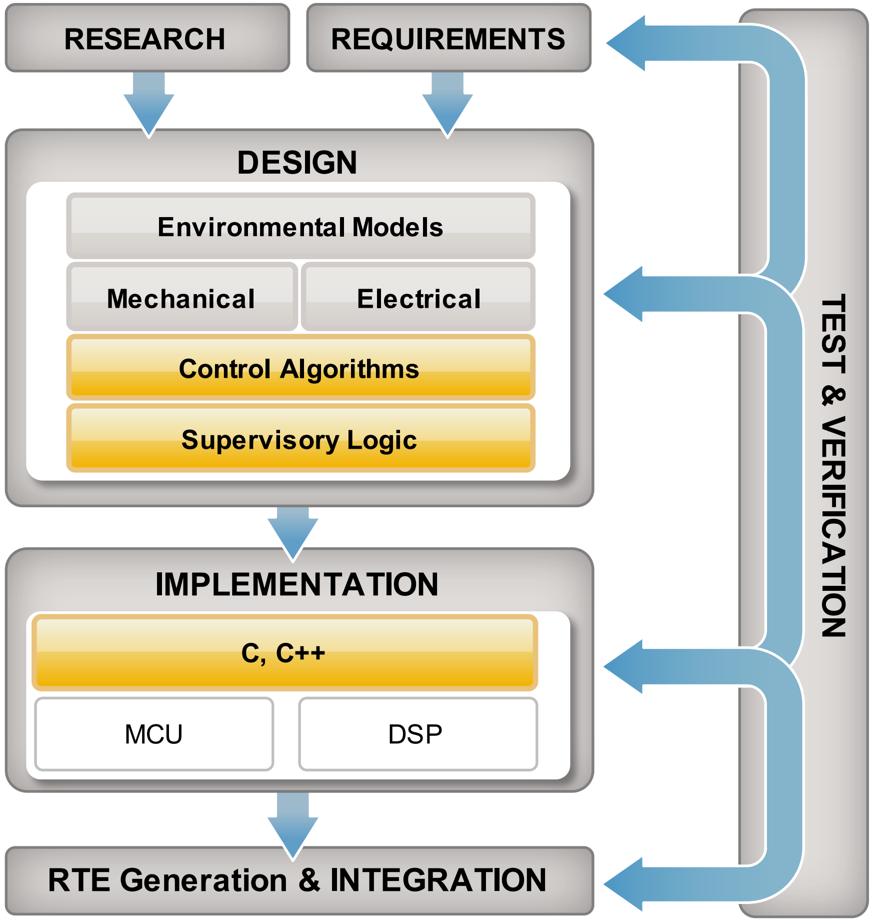
④

Simulate before you generate code

#4 Simulate before you generate code

- Take advantage of early verification through simulation

- Make sure SWC implementation is correct early
- Simulate multiple SWC's together in Simulink before code integration
- Use SIL and PIL to verify generated code at the unit level before RTE generation



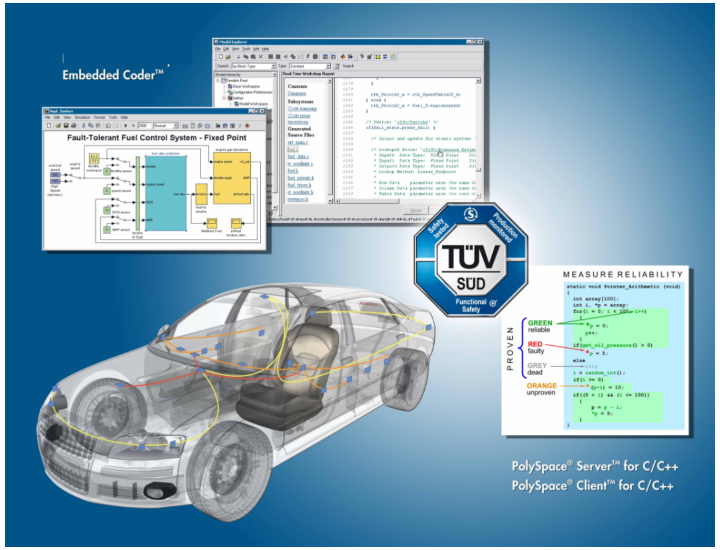
⑤

Plan ahead for ISO 26262

#5 Plan ahead for ISO 26262

- Determine how your AUTOSAR process will address safety-standards

- **Products supported for ISO 26262 tool qualification include:**
  - Embedded Coder
  - Simulink V&V
  - Simulink Design Verifier
  - PolySpace Code Verifiers
- **Artifacts certified by TÜV SÜD**
  - Requires use of V&V workflow
- **ISO 26262 Advisory Service available**



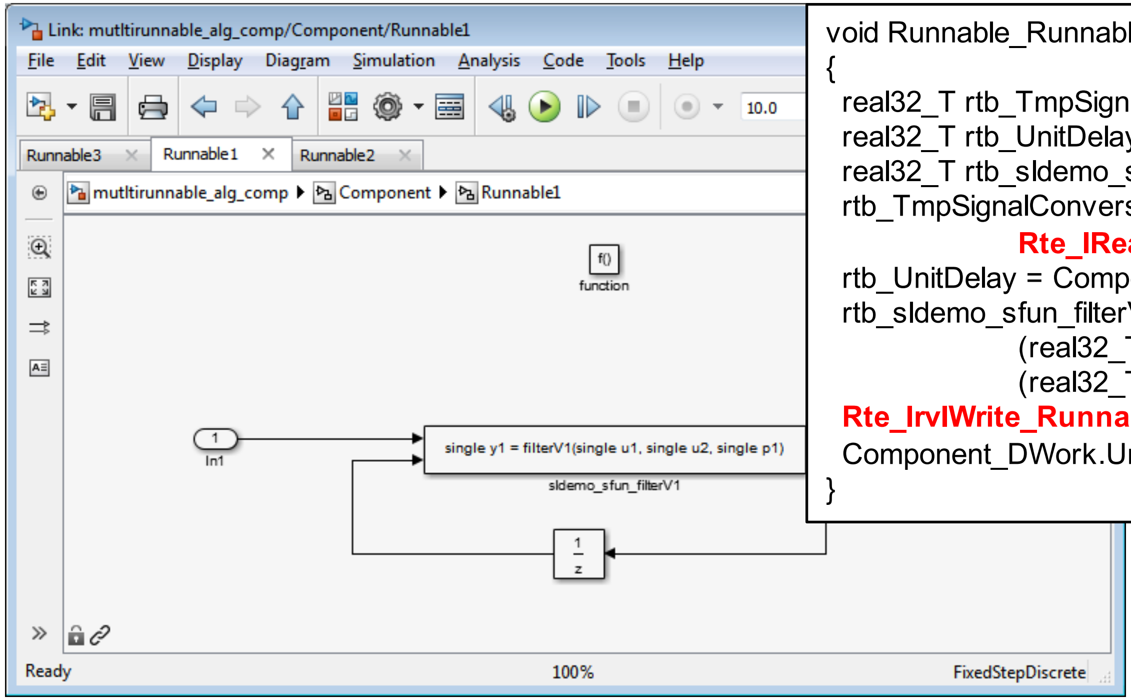
```
static void Pointer_Arithmetic (void)
{
  int array[100];
  int i, *p = array;
  for(i = 0; i < 100; i++)
    {
      *p = 0;
      p++;
    }
  if(get_oil_pressure() > 0)
    *p = 5;
  else
    i++;
  i = random_int();
  if(i >= 0)
    *(p-i) = 10;
  if((0 < i) && (i <= 100))
    {
      p = p - i;
      *p = 5;
    }
}
```

⑥

Use Simulink to help migrate your legacy code to AUTOSAR

#6 Use Simulink to help migrate your legacy code to AUTOSAR

- Reuse of Legacy Code
  - Integration for simulation, production code generation
  - Can generate AUTOSAR RTE API access points



Link: mutltirunnable_alg_comp/Component/Runnable1
File Edit View Display Diagram Simulation Analysis Code Tools Help
10.0
Runnable3
Runnable1
Runnable2
mutltirunnable_alg_comp ▸ Component ▸ Runnable1
f()
function
1
In1
single y1 = filterV1(single u1, single u2, single p1)
sldemo_sfun_filterV1
1
z
Ready
100%
FixedStepDiscrete

```
void Runnable_Runnable1(void)
{
  real32_T rtb_TmpSignalConversionAtIn1Out;
  real32_T rtb_UnitDelay;
  real32_T rtb_sldemo_sfun_filterV1;
  rtb_TmpSignalConversionAtIn1Out =
                   Rte_IRead_Runnable_Runnable1_Fast_in_Fast_in();
  rtb_UnitDelay = Component_DWork.UnitDelay_DSTATE;
  rtb_sldemo_sfun_filterV1 = filterV1( (real32_T)rtb_TmpSignalConversionAtIn1Out,
                   (real32_T)rtb_UnitDelay,
                   (real32_T)Component_P.sldemo_sfun_filterV1_p1);
  Rte_IrvIWrite_Runnable_Runnable1_a(rtb_sldemo_sfun_filterV1);
  Component_DWork.UnitDelay_DSTATE = rtb_sldemo_sfun_filterV1;
}
```

⑦

Automate, automate, automate

#7 Automate, automate, automate

- Use API’s for workflow automation!

- **Manual process difficult due to:**
  - The complexity of the standard, naming conventions
  - Iterative work cycles with AUTOSAR
  - Complex code APIs and XML file definitions

- Use documented MATLAB APIs to configure SWCs in Simulink

```
%% Setup AUTOSAR Configuration
programmatically

model = 'rtwdemo_autosar_counter';

% Modify AUTOSAR Properties
autosarProps =
autosar.api.getAUTOSARProperties(model);
set(autosarProps, 'Input', 'IsService',
true);
set(autosarProps, 'XmlOptions',
'ArxmlFilePackaging','SingleFile');
```

⑧

Use production code generation

#8 Use production code generation

- Hand coding AUTOSAR is painful

```
void Runnable_simple_alg_Step(void)
{
  real_T rtb_Gain;
  real_T rtb_Delay;
  real_T rtb_Delay1;
  real_T rtb_TmpSignalConversionAtFast_i;
  if (simple_alg_M->Timing.TaskCounters.TID[1]  == 0) {
    Rte_Receive_Fast_in_Fast_in(&rtb_TmpSignalConversionAtFast_i);
    rtb_Delay = simple_alg_DWork.Delay_DSTATE;
    rtb_Delay1 = simple_alg_DWork.Delay1_DSTATE;
    rtb_Gain = simple_alg_DWork.Delay2_DSTATE;
    rtb_Gain = (((rtb_TmpSignalConversionAtFast_i  + simple_alg_DWork.Delay_DSTATE)  +  simple_alg_DWork.Delay1_DSTATE)  + rtb_Gain) * simple_alg_P.Gain_Gain;
    if (simple_alg_M->Timing.TaskCounters.TID[2]  == 0) {
      simple_alg_B.RateTransition  = rtb_Gain;
    }
     simple_alg_DWork.Delay_DSTATE  = rtb_TmpSignalConversionAtFast_i;
    simple_alg_DWork.Delay1_DSTATE  = rtb_Delay;
    simple_alg_DWork.Delay2_DSTATE  = rtb_Delay1;
  }
   if (simple_alg_M->Timing.TaskCounters.TID[2]  == 0) {
    Rte_IWrite_Runnable_simple_alg_Step_Out1_Out1 (simple_alg_B.RateTransition
                     + Rte_IRead_Runnable_simple_alg_Step_Slow_in_Slow_in());
  }
…
```

```
…
<RUNNABLE-ENTITY UUID="aef16585-a355-494f- accd-1a548ca22e27">
 <SHORT-NAME>Runnable_simple_alg_Step</SHORT-NAME>
    <MINIMUM-START-INTERVAL>0</MINIMUM-START-INTERVAL>
      <CAN-BE-INVOKED-CONCURRENTLY>false</CAN-BE-INVOKED-CONCURRENTLY>
      <DATA-READ-ACCESSS>
        <VARIABLE-ACCESS>
          <SHORT-NAME>IN_Slow_in_Slow_in</SHORT-NAME>
          …
</RUNNABLE-ENTITY>
…
```

```
…
<SENDER-RECEIVER-INTERFACE>
  <SHORT-NAME>Out1</SHORT-NAME>
    <IS-SERVICE>false</IS-SERVICE>
    <DATA-ELEMENTS>
      <VARIABLE-DATA-PROTOTYPE>
        <SHORT-NAME>Out1</SHORT-NAME>
          …
       </VARIABLE-DATA-PROTOTYPE>
  </DATA-ELEMENTS>
</SENDER-RECEIVER-INTERFACE>
…
```

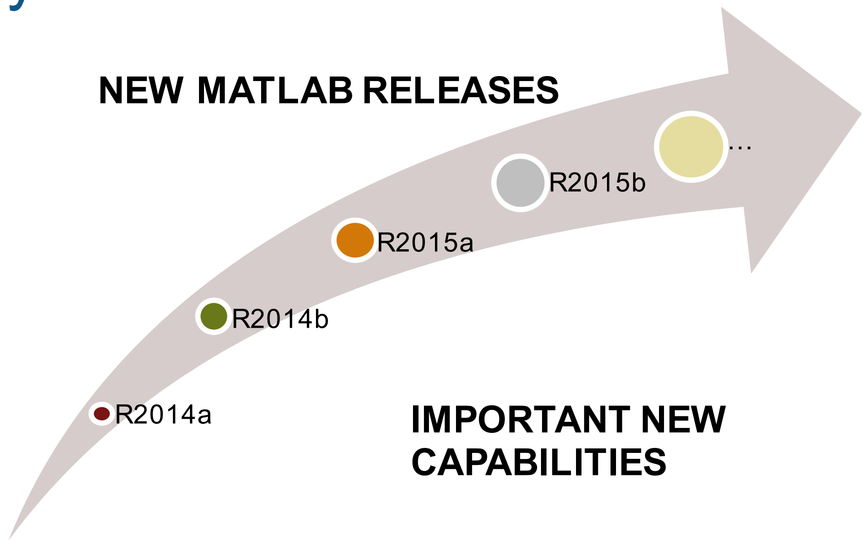
⑨

Actively plan for migration

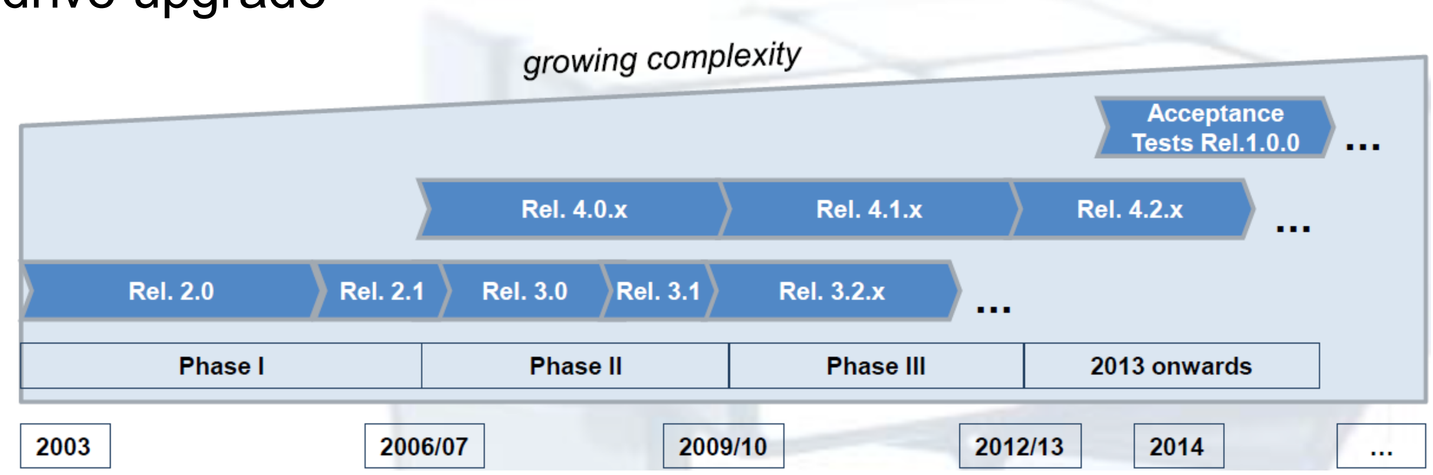
#9 Actively plan for migration

- Tools and standards are changing rapidly

- Account for:
  - New versions of AUTOSAR
  - New versions of Simulink
- Consider:
  - How often to upgrade
  - What will drive upgrade


NEW MATLAB RELEASES

IMPORTANT NEW CAPABILITIES



Source: 7th AUTOSAR Open Conference, 22.10.2014

```
case ‘Assistance’:
```

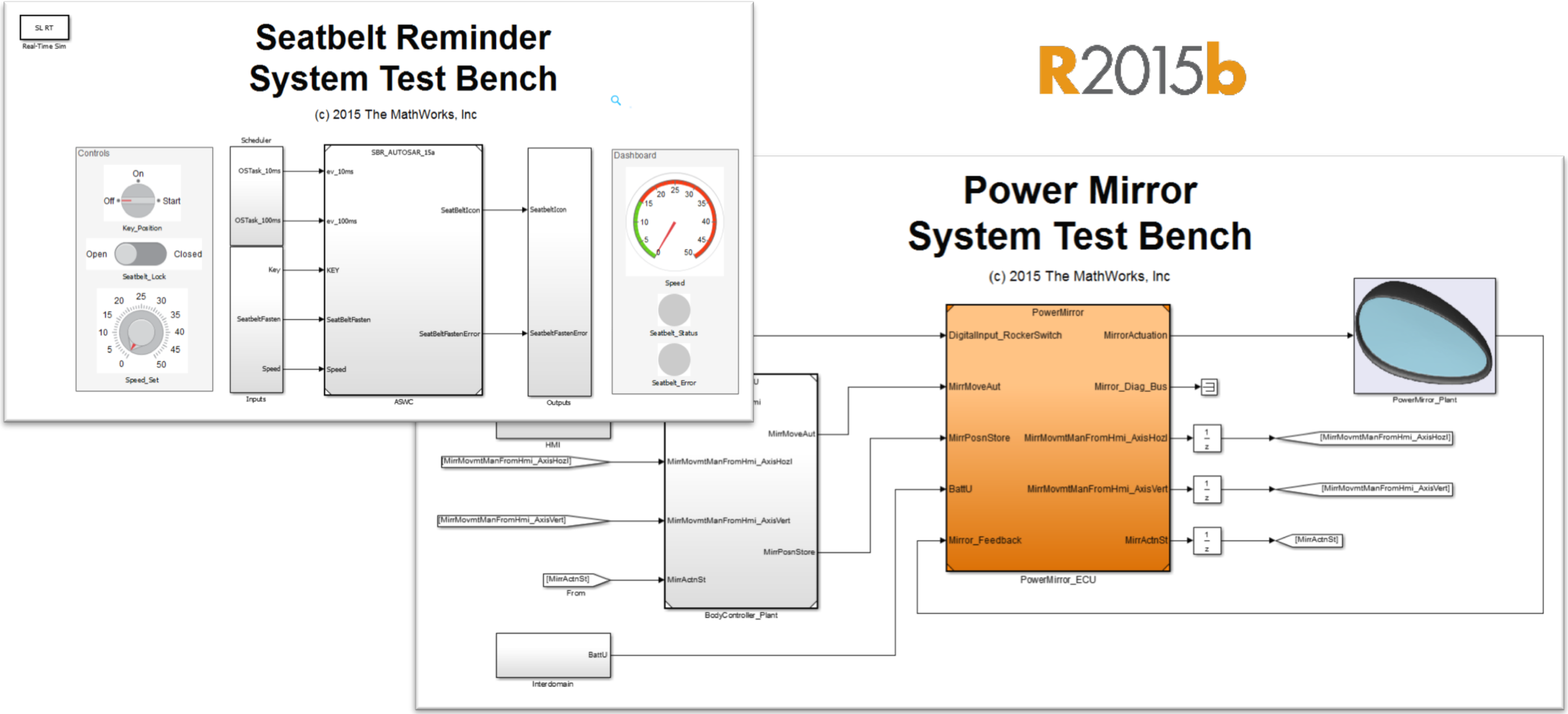
Training Services

Developing Embedded Targets Advisory Service

case ‘AUTOSAR Demo Pod’:

Model. Code. Production.

R2015b



MathWorks
AUTOMOTIVE
CONFERENCE 2015

```
default :
  printf(“Brain up-to-date!”);
}
```

And one last thing …
AUTOSAR – Antagonizing the „German Coast Guard“ Effect

Source: https://youtu.be/zkalf0odHs8 German Coast Guard Commercial 'We are Sinking' [HD]