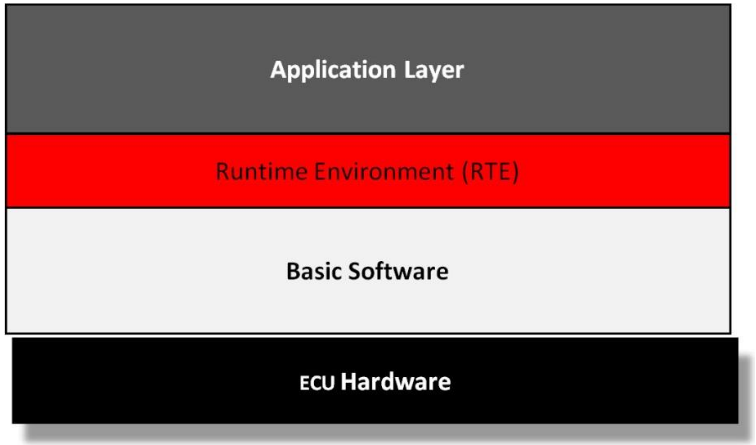
Simulink for AUTOSAR:

Best Practices

李智慧
高级技术咨询顾问

What is AUTOSAR?

*AUT*omotive *O*pen *S*ystem *AR*chitecture



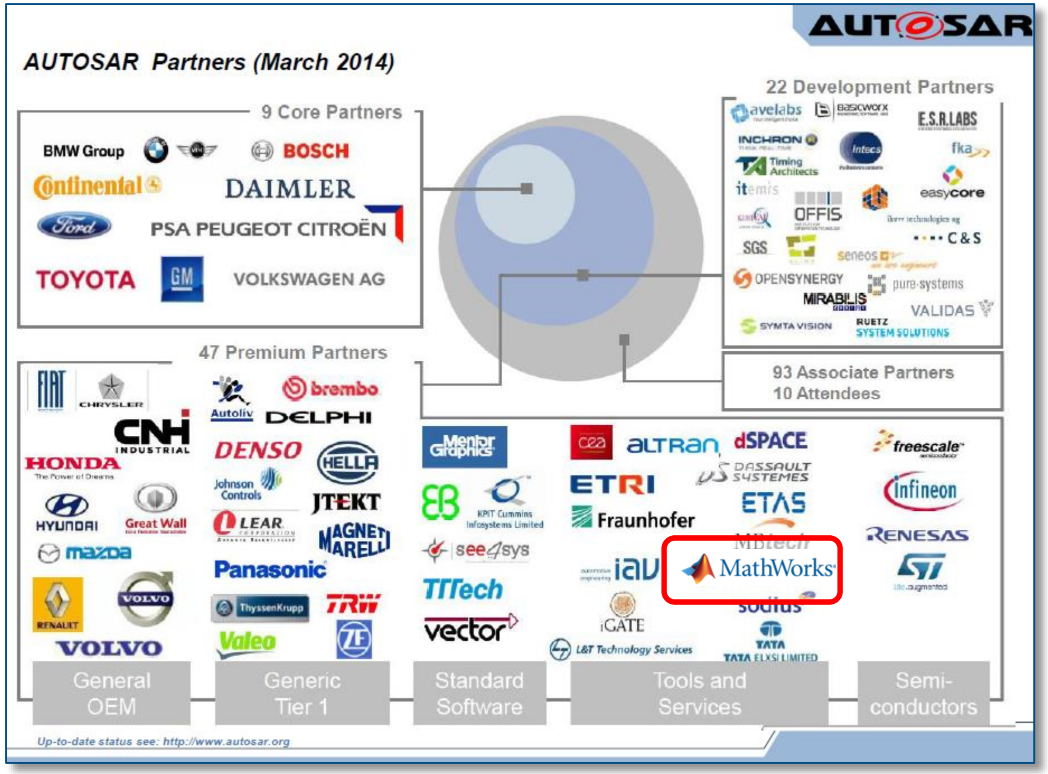
Application Layer

Runtime Environment (RTE)

Basic Software

ECU Hardware

- *Objective:*
  Establish an open standard for automotive E/E architecture



- *Partnership*
  Consisting of more than 180 companies from the global automotive industry

Latest update:
http://www.autosar.org/partners/current-partners/

Agenda

Simulink for AUTOSAR - Introduction

- Workflows
- Capabilities

Simulink for AUTOSAR – User Stories

- Production Code Generation with Embedded Coder

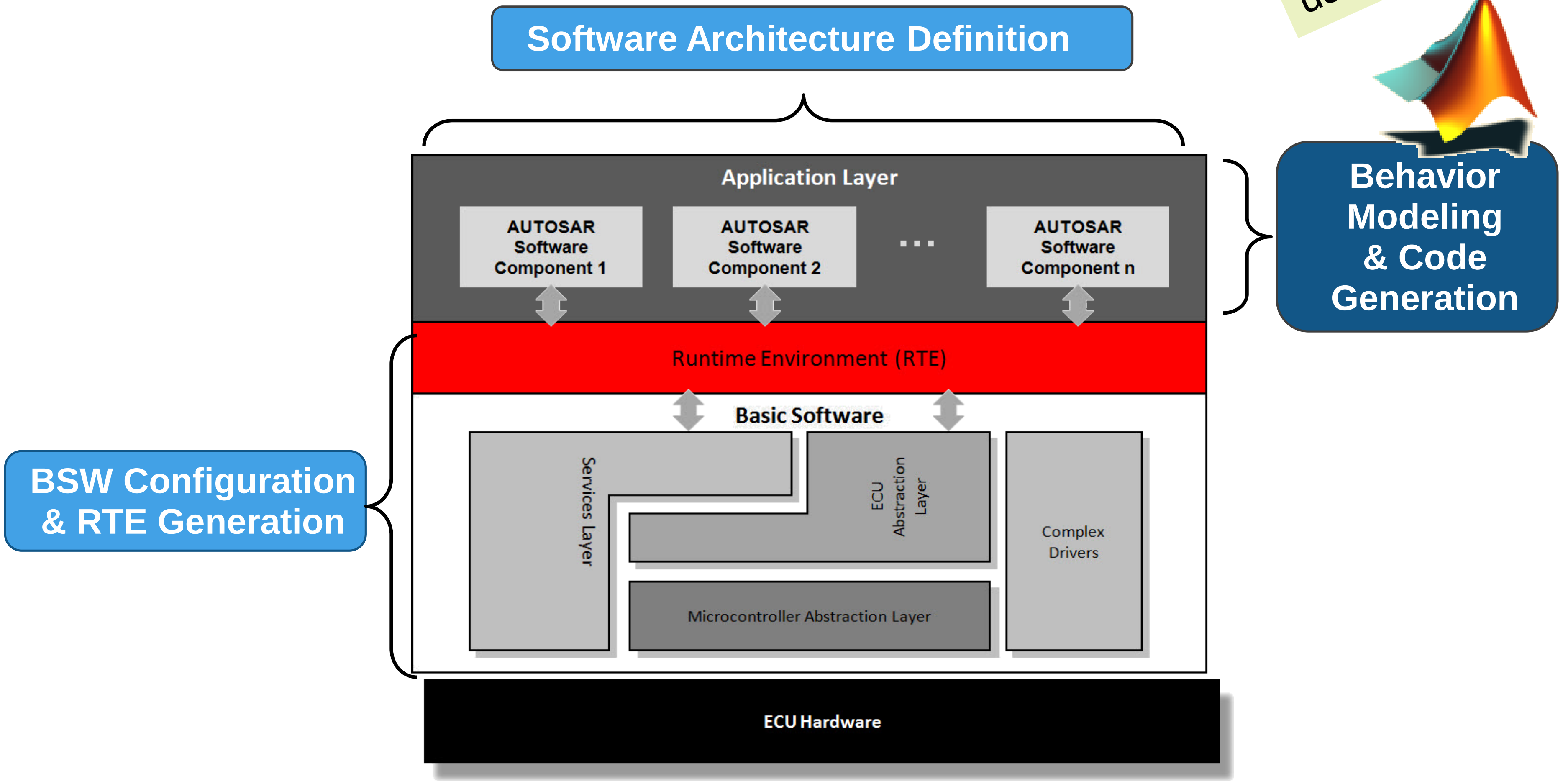
Simulink for AUTOSAR – Best Practices

- Best Practices for using Simulink for AUTOSAR

Summary & Conclusions

Simulink & Stateflow for Behavior Modeling, Embedded Coder for Production Code

"All toolboxes for MBD are still usable."



Software Architecture Definition

Application Layer

AUTOSAR Software Component 1

AUTOSAR Software Component 2

...

AUTOSAR Software Component n

Runtime Environment (RTE)

Basic Software

Services Layer

ECU Abstraction Layer

Complex Drivers

Microcontroller Abstraction Layer

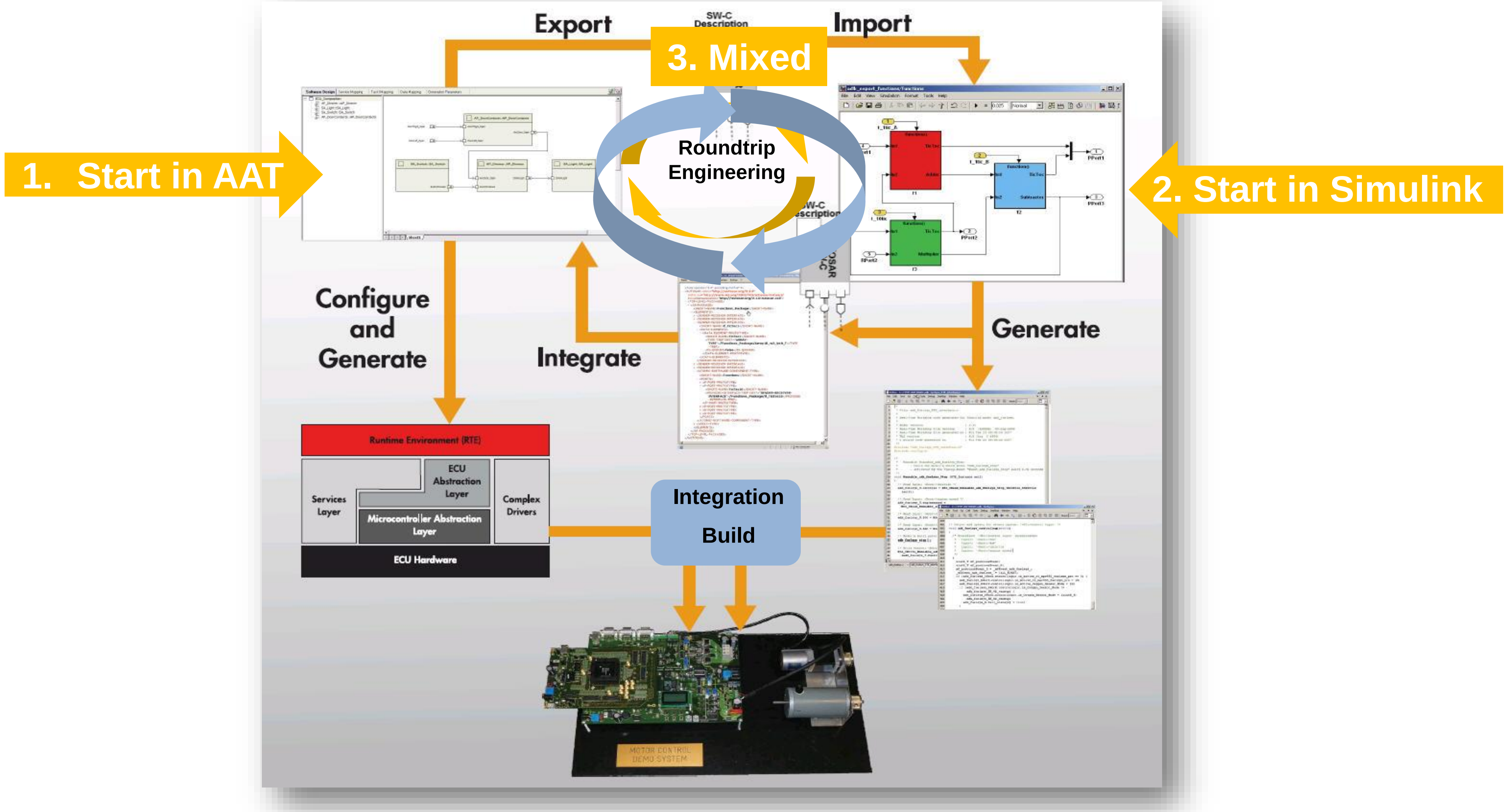
ECU Hardware

Behavior Modeling & Code Generation

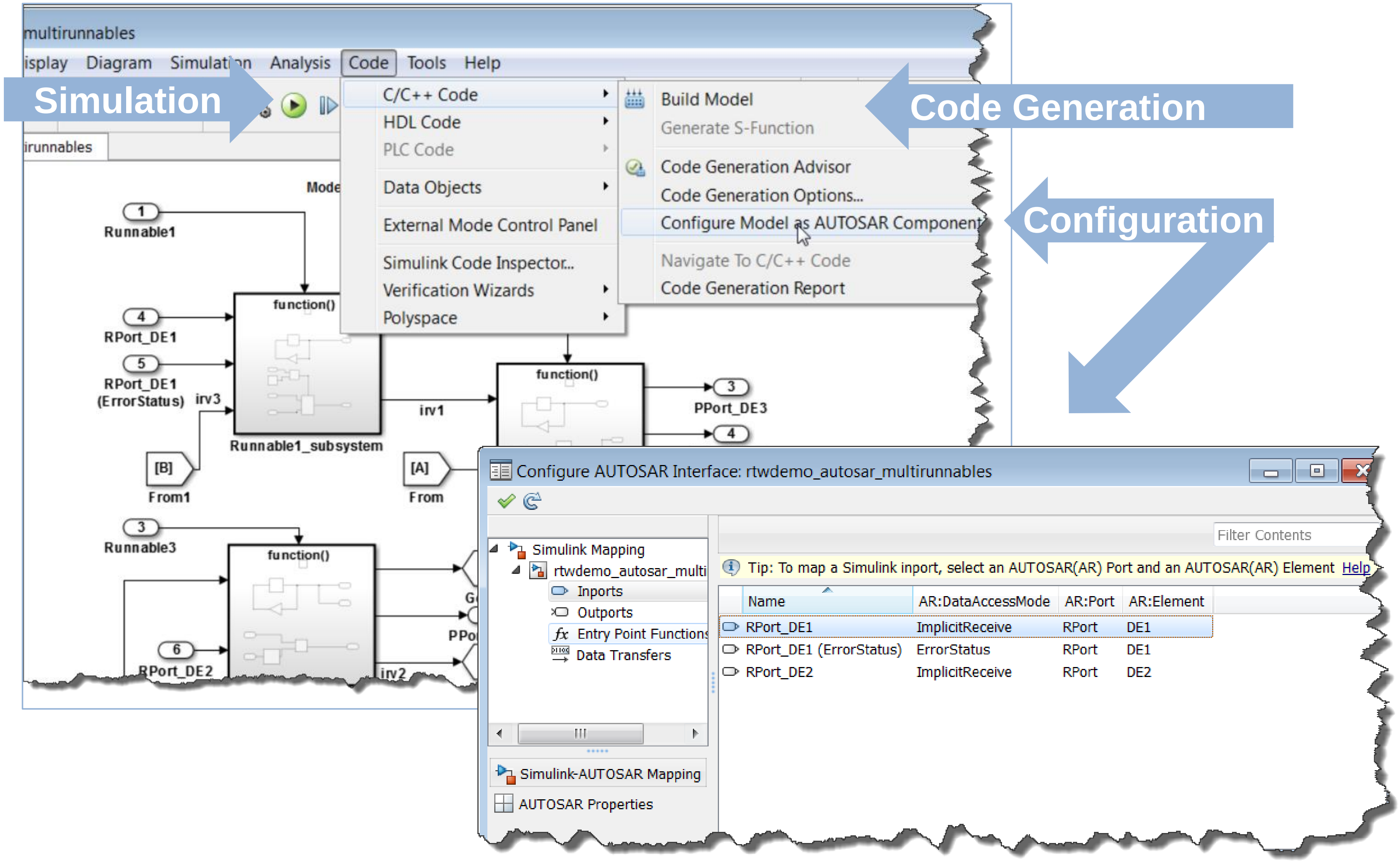
BSW Configuration & RTE Generation

Workflows

1. Top-Down, 2. Bottom-Up, 3. Mixed



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

Agenda

Simulink for AUTOSAR - Introduction

- Workflows
- Capabilities

Simulink for AUTOSAR – User Stories

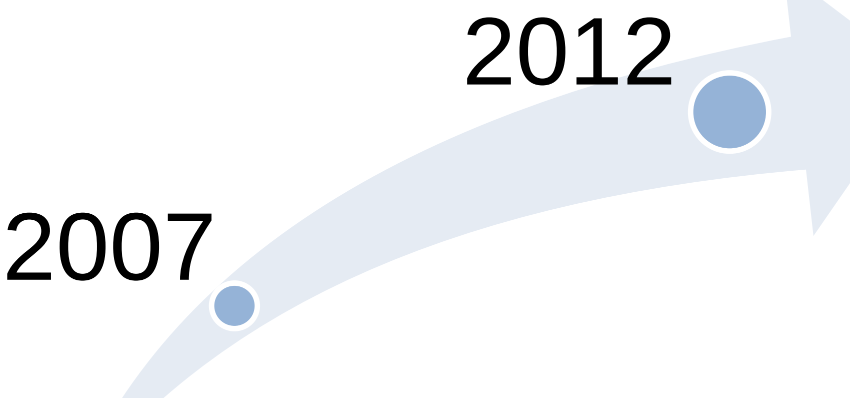
- Production Code Generation with Embedded Coder

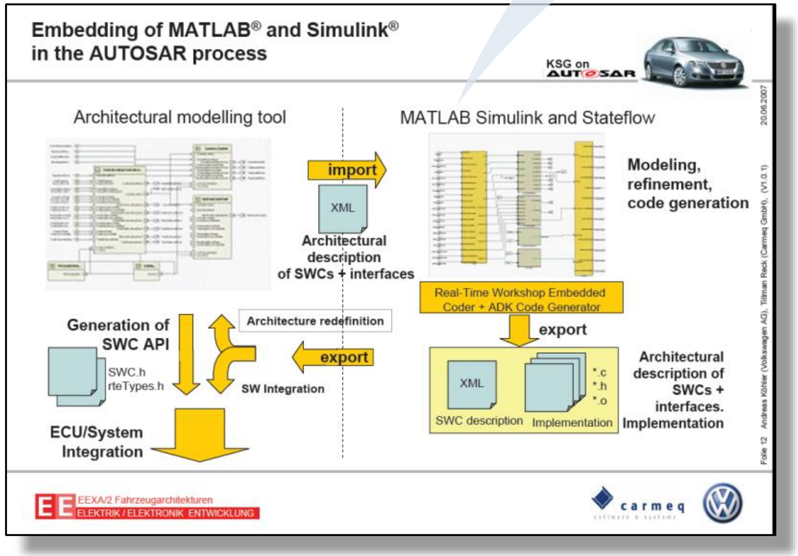
Simulink for AUTOSAR – Best Practices

- Best Practices for using Simulink for AUTOSAR

Summary & Conclusions

Long-term Successful Collaboration with Volkswagen…





Embedding of MATLAB® and Simulink® in the AUTOSAR process

KSG on AUTOSAR

Architectural modelling tool

MATLAB Simulink and Stateflow

import

XML

Architectural description of SWCs + interfaces

Modeling, refinement, code generation

Real-Time Workshop Embedded Coder + ADK Code Generator

export

Generation of SWC API

SWC.h
rteTypes.h

Architecture redefinition

export

SW Integration

XML

SWC description

*.c
*.h
*.o

Implementation

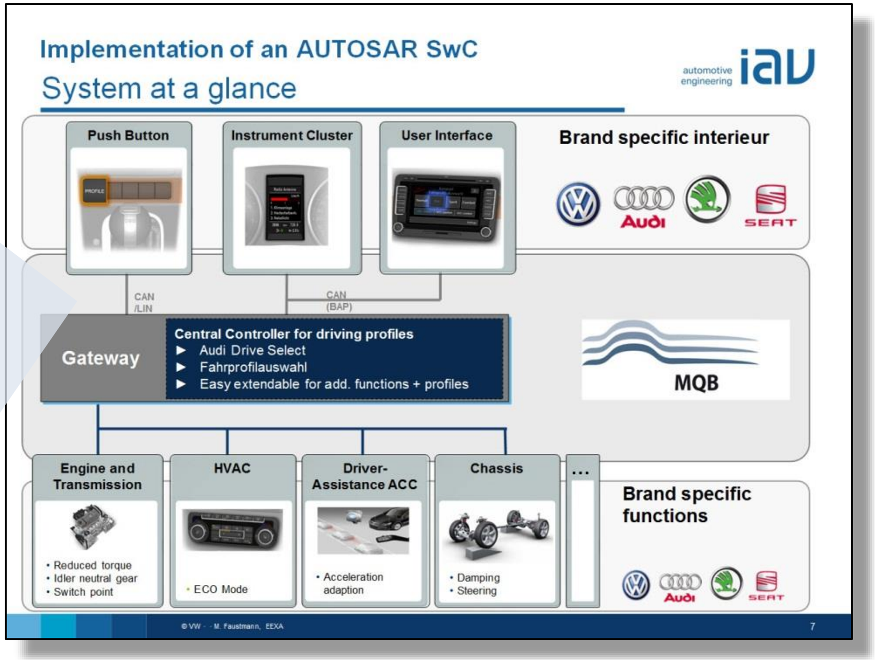
Architectural description of SWCs + interfaces. Implementation

ECU/System Integration

EEXA/2 Fahrzeugarchitekturen
ELEKTRIK / ELEKTRONIK ENTWICKLUNG

carmeq
software & systems

Folie 12 Andreas Köhler (Volkswagen AG), Tillman Reck (Carmeq GmbH), (V1.0.1) 20.06.2007



Implementation of an AUTOSAR SwC

System at a glance

automotive engineering iav

Push Button

Instrument Cluster

User Interface

Brand specific interieur

CAN /LIN

CAN (BAP)

Gateway

Central Controller for driving profiles
- Audi Drive Select
- Fahrprofilauswahl
- Easy extendable for add. functions + profiles

MQB

Engine and Transmission
- Reduced torque
- Idler neutral gear
- Switch point

HVAC
- ECO Mode

Driver-Assistance ACC
- Acceleration adaption

Chassis
- Damping
- Steering

…

Brand specific functions

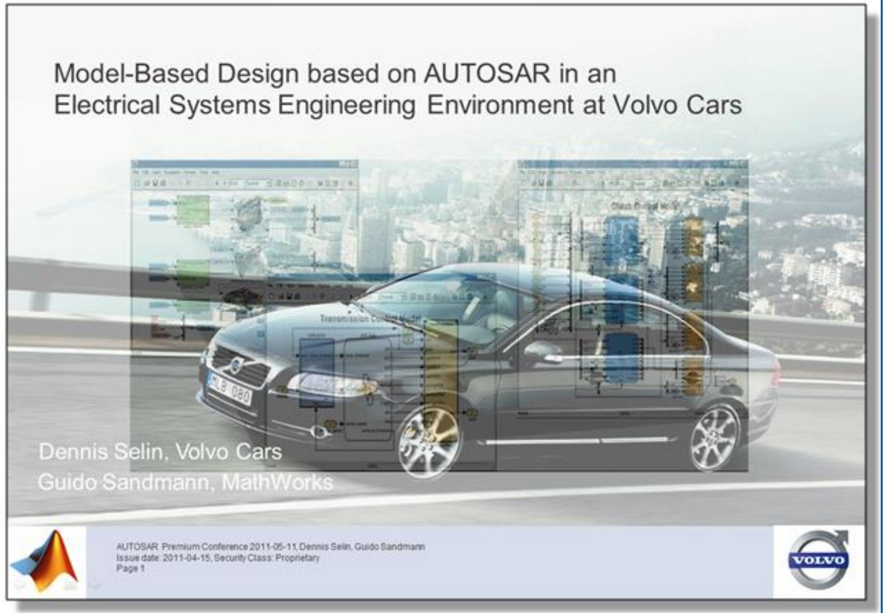
© VW - - M. Faustmann, EEXA 7

…from a "Proof of Concept" project

…to series production across brands

More User Stories…

Market Situation and Rollout Strategies: Volvo Cars

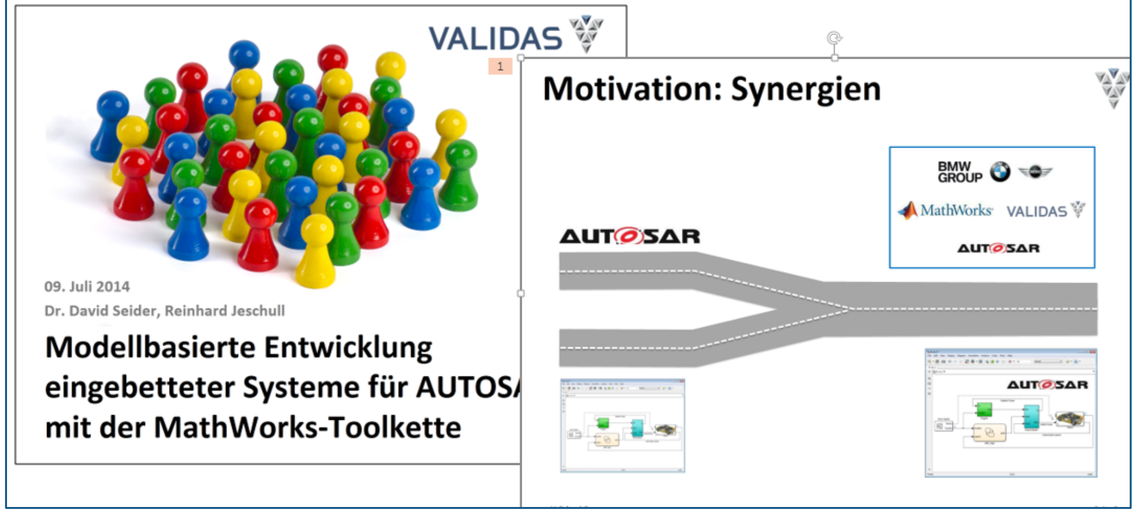


Model-Based Design based on AUTOSAR in an Electrical Systems Engineering Environment at Volvo Cars

Dennis Selin, Volvo Cars
Guido Sandmann, MathWorks

AUTOSAR Premium Conference 2011-05-11, Dennis Selin, Guido Sandmann
Issue date: 2011-04-15, Security Class: Proprietary
Page 1

Market situation and rollout strategies at our most important customers: Validas for BMW

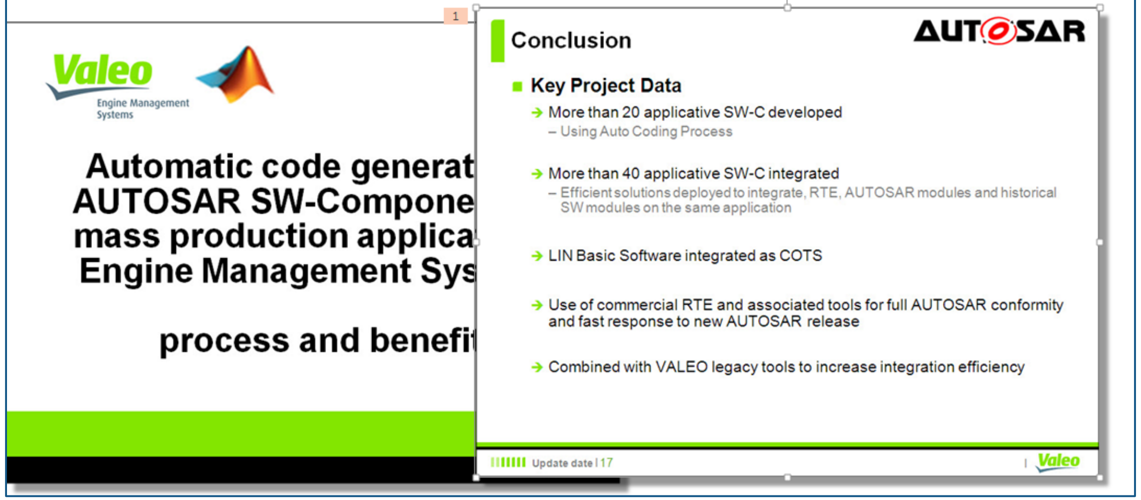


09. Juli 2014
Dr. David Seider, Reinhard Jeschull

Modellbasierte Entwicklung eingebetteter Systeme für AUTOS
mit der MathWorks-Toolkette

Motivation: Synergien

Market Situation and Rollout Strategies at Our Most Important Customers: Valeo



Automatic code generat
AUTOSAR SW-Compone
mass production applica
Engine Management Sys

process and benefi

Conclusion

- Key Project Data
  - → More than 20 applicative SW-C developed
    - – Using Auto Coding Process
  - → More than 40 applicative SW-C integrated
    - – Efficient solutions deployed to integrate, RTE, AUTOSAR modules and historical SW modules on the same application
  - → LIN Basic Software integrated as COTS
  - → Use of commercial RTE and associated tools for full AUTOSAR conformity and fast response to new AUTOSAR release
  - → Combined with VALEO legacy tools to increase integration efficiency

Update date I 17 | Valeo

Agenda

Simulink for AUTOSAR - Introduction

- Workflows
- Capabilities

Simulink for AUTOSAR – User Stories

- Production Code Generation with Embedded Coder

Simulink for AUTOSAR – Best Practices

- Best Practices for using Simulink for AUTOSAR

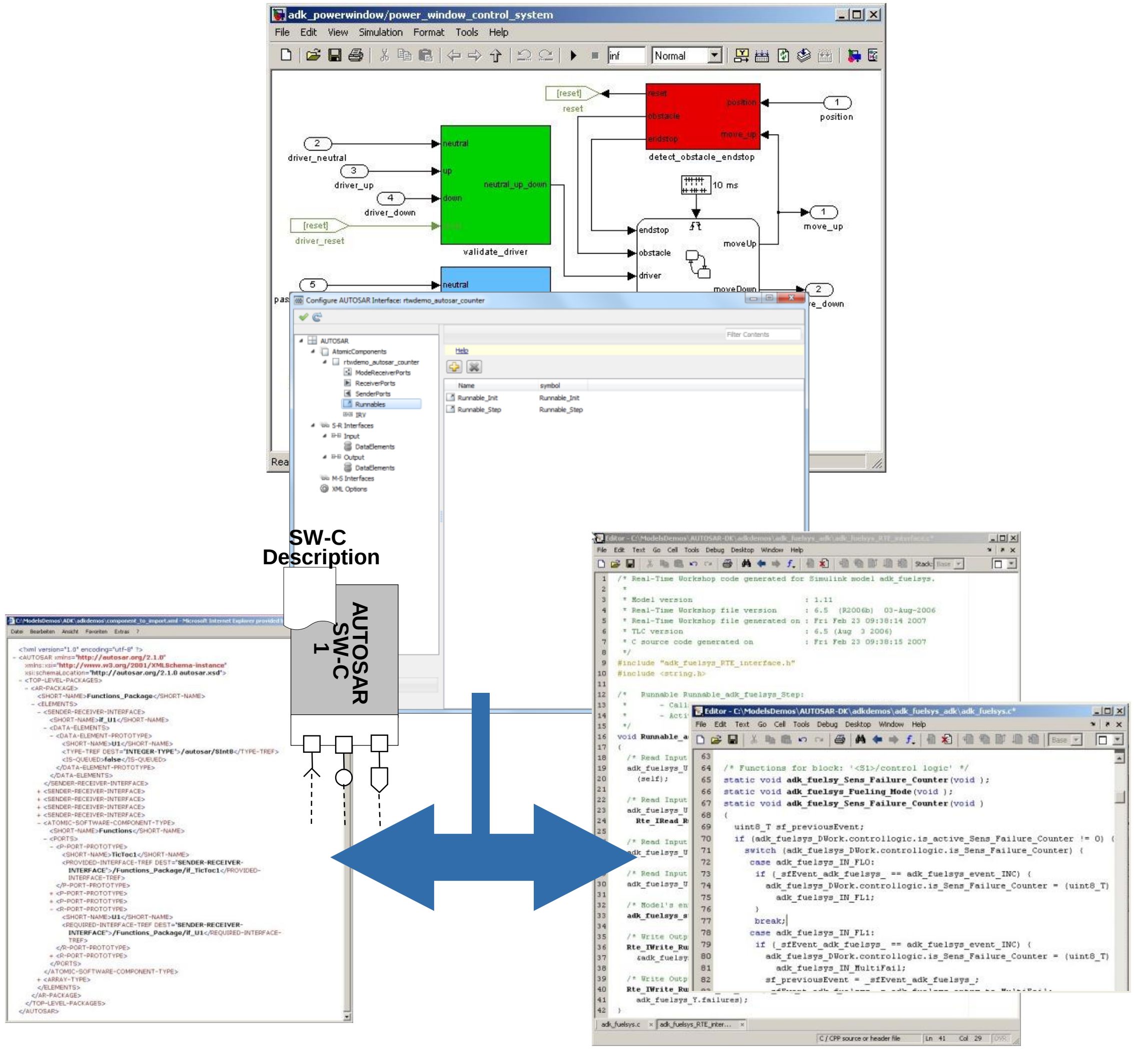
Summary & Conclusions

#1 Decide strategy for migrating existing Simulink models to AUTOSAR

- Clean sheet start

- Start with existing Simulink models

- Maintain one model for AUTOSAR and non-AUTOSAR

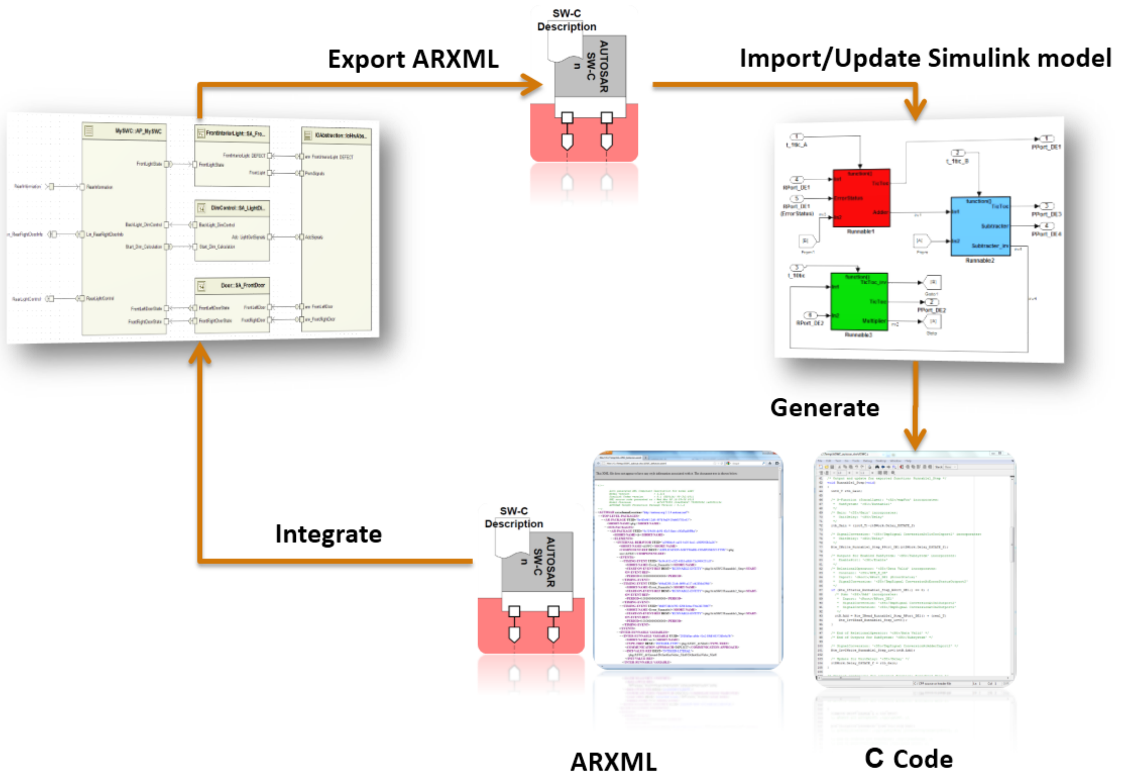

adk_powerwindow/power_window_control_system
validate_driver
detect_obstacle_endstop
Configure AUTOSAR Interface: rtwdemo_autosar_counter
SW-C Description
AUTOSAR SW-C 1

#2 Use one AUTOSAR workflow

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

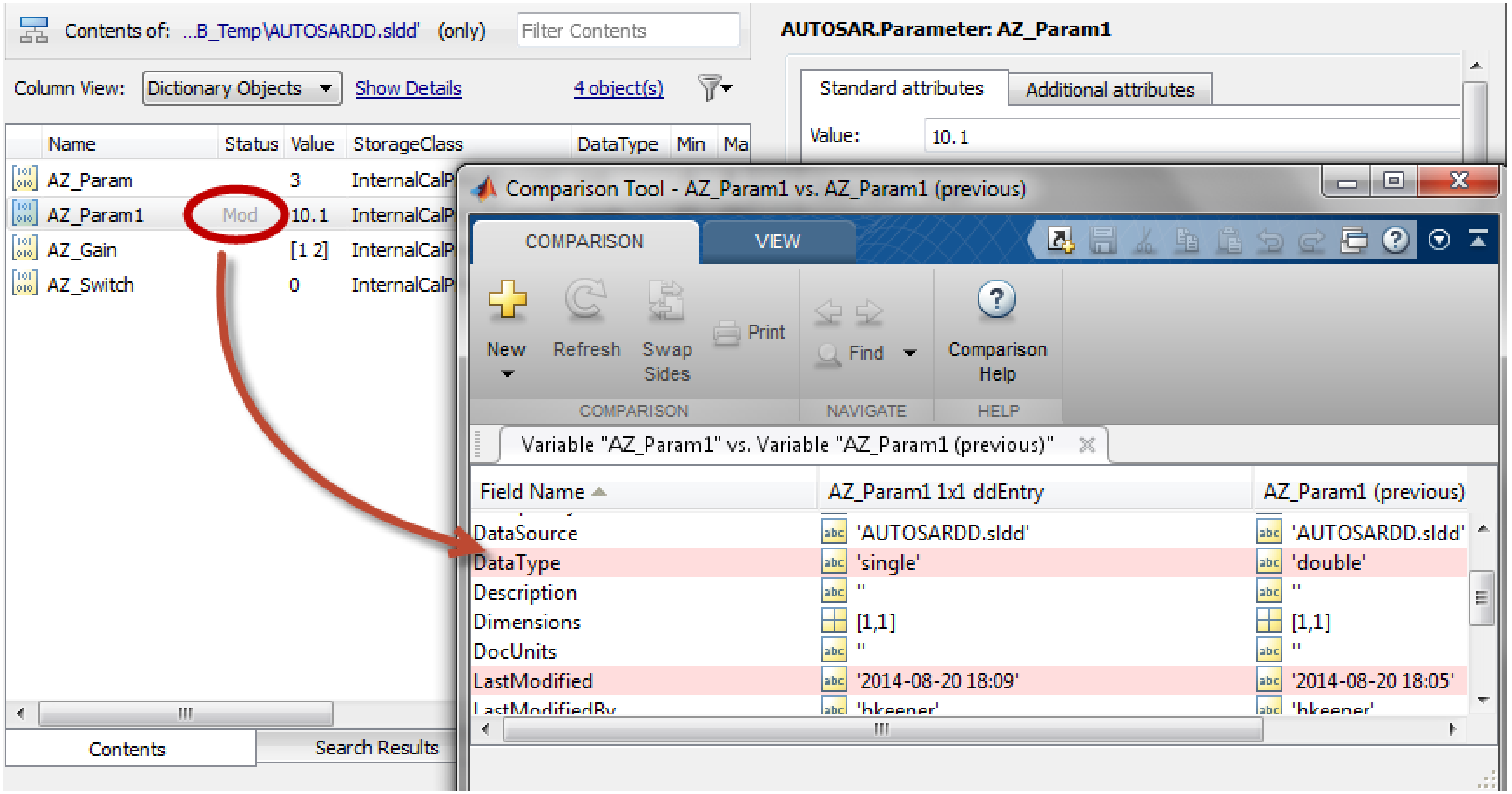
- Select tools that best support your workflow and AUTOSAR concepts

- Select simplest approach for applying AUTOSAR configuration to your Simulink model



#3 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)

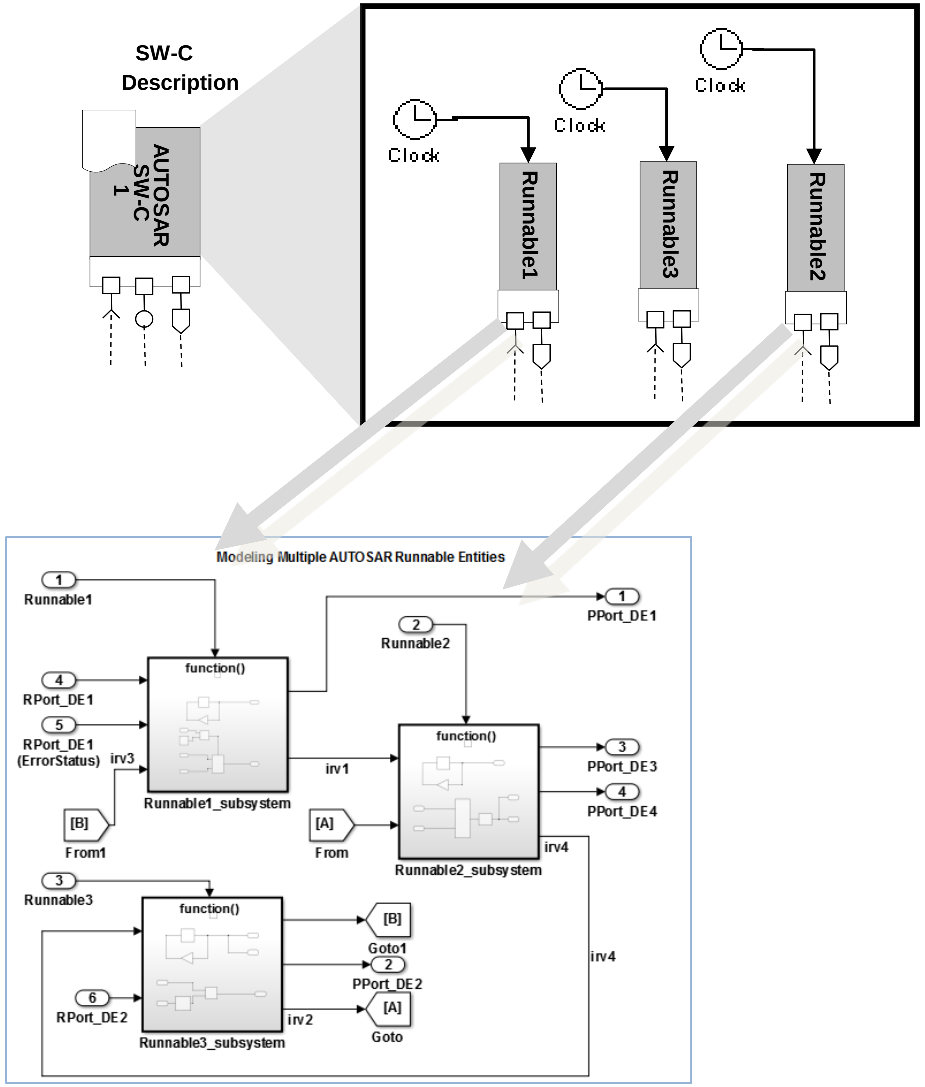
Variable "AZ_Param1" vs. Variable "AZ_Param1 (previous)"

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'

#4 Establish modeling standards
– For Simulink and AUTOSAR

- Base it on your workflow and data management

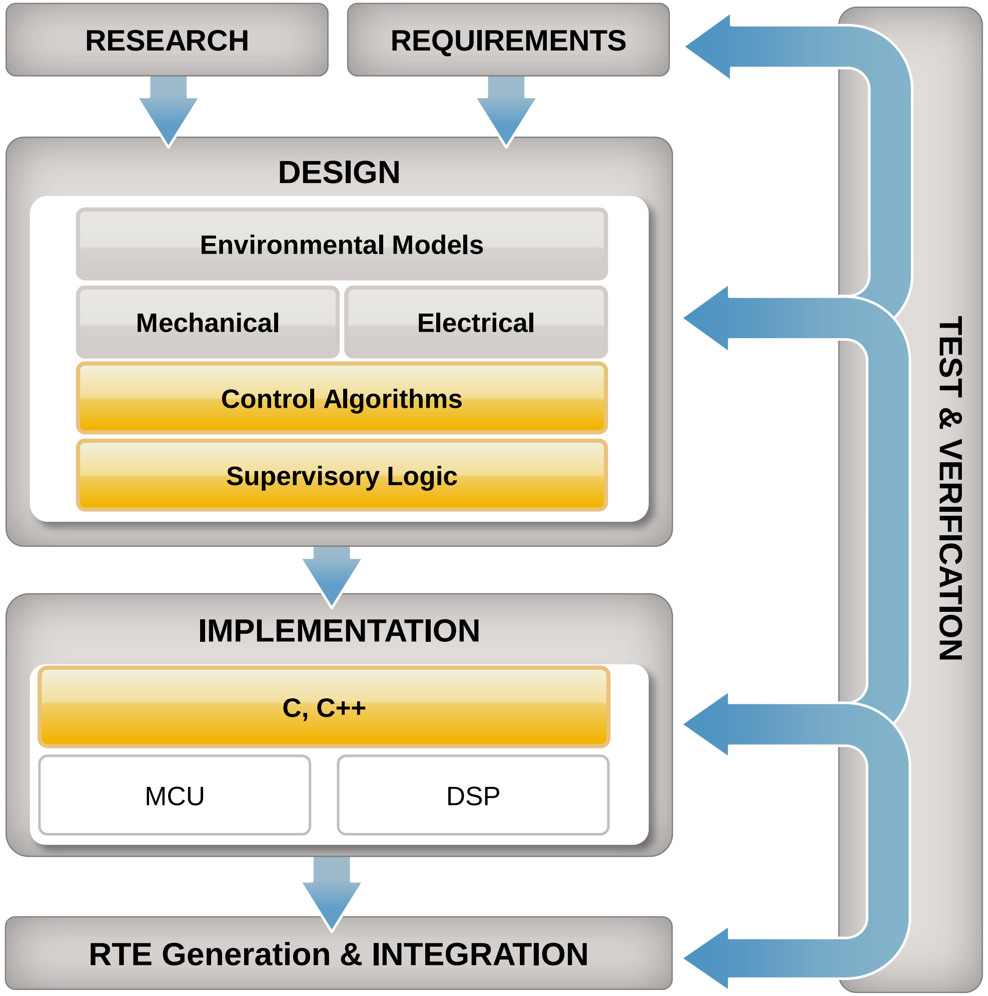
- Use Simulink Model Advisor to enforce modeling style early in model development



#5 Simulate before you generate code

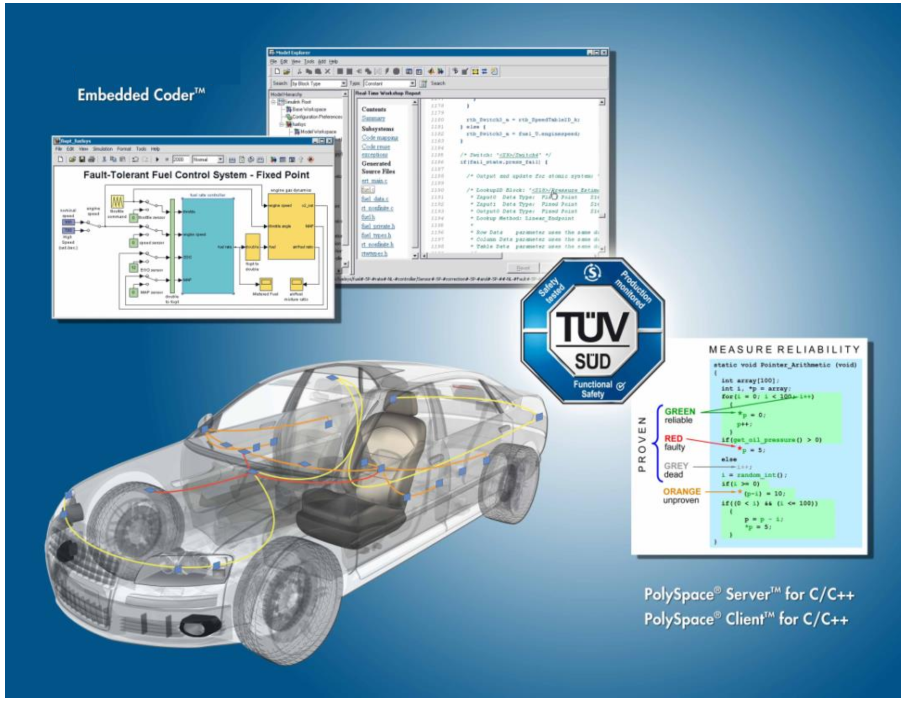
– Take advantage of **early verification** through simulation

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



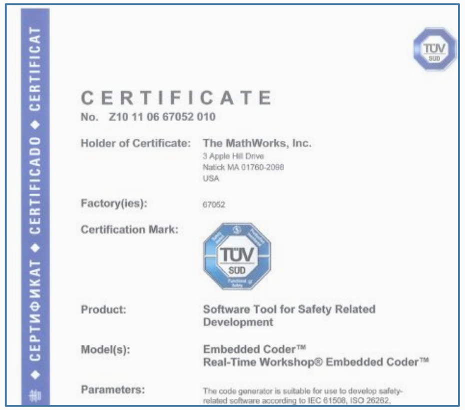
#6 Plan ahead for ISO 26262 – Determine how AUTOSAR process will address safety-standards

- Products supported for ISO 26262 tool qualification include:
  - Embedded Coder
  - Simulink V&V
  - Simulink Design Verifier
  - PolySpace
- 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;
    }
}
```

PolySpace® Server™ for C/C++
PolySpace® Client™ for C/C++



СЕРТИФИКАТ ◆ CERTIFICADO ◆ CERTIFICAT ◆ CERTIFICAT

CERTIFICATE
No. Z10 11 06 67052 010

Holder of Certificate: The MathWorks, Inc.
3 Apple Hill Drive
Natick MA 01760-2098
USA

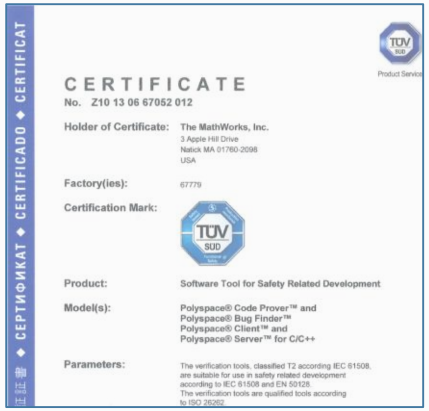
Factory(ies): 67052

Certification Mark:

Product: Software Tool for Safety Related Development

Model(s): Embedded Coder™
Real-Time Workshop® Embedded Coder™

Parameters: The code generator is suitable for use to develop safety-related software according to IEC 61508, ISO 26262,



СЕРТИФИКАТ ◆ CERTIFICADO ◆ CERTIFICAT ◆ CERTIFICAT

Product Service

CERTIFICATE
No. Z10 13 06 67052 012

Holder of Certificate: The MathWorks, Inc.
3 Apple Hill Drive
Natick MA 01760-2098
USA

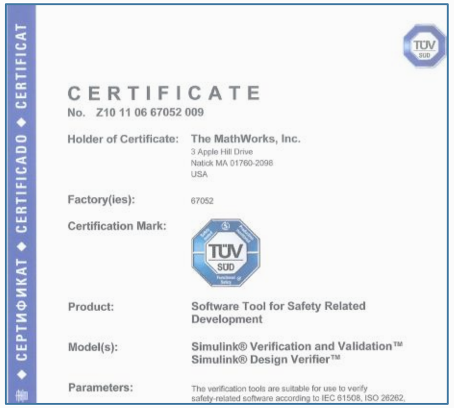
Factory(ies): 67779

Certification Mark:

Product: Software Tool for Safety Related Development

Model(s): Polyspace® Code Prover™ and
Polyspace® Bug Finder™
Polyspace® Client™ and
Polyspace® Server™ for C/C++

Parameters: The verification tools, classified T2 according IEC 61508, are suitable for use in safety related development according to IEC 61508 and EN 50128.
The verification tools are qualified tools according to ISO 26262.



СЕРТИФИКАТ ◆ CERTIFICADO ◆ CERTIFICAT ◆ CERTIFICAT

CERTIFICATE
No. Z10 11 06 67052 009

Holder of Certificate: The MathWorks, Inc.
3 Apple Hill Drive
Natick MA 01760-2098
USA

Factory(ies): 67052

Certification Mark:

Product: Software Tool for Safety Related Development

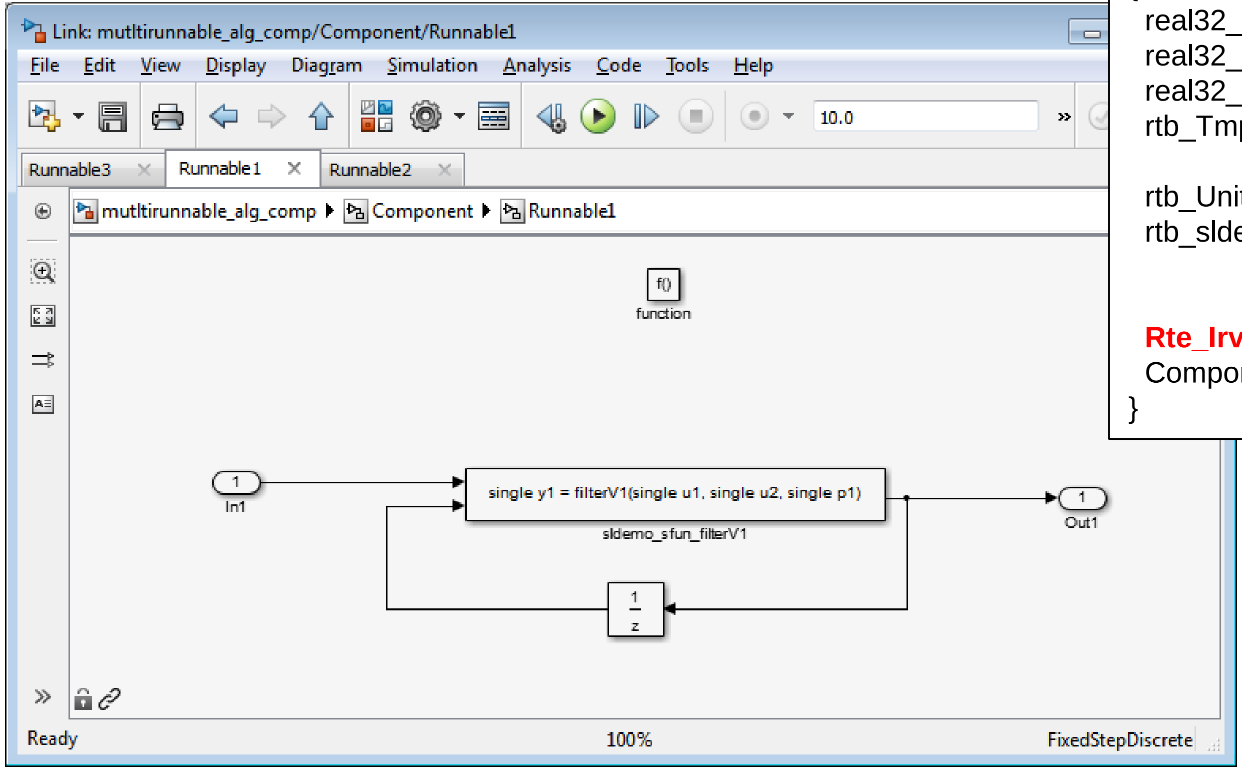
Model(s): Simulink® Verification and Validation™
Simulink® Design Verifier™

Parameters: The verification tools are suitable for use to verify safety-related software according to IEC 61508, ISO 26262,

#7 Use Simulink to migrate 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
Out1
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;
}
```

#8 Automate, automate, automate

– Use API’s for workflow automation!

- **Manual process is 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

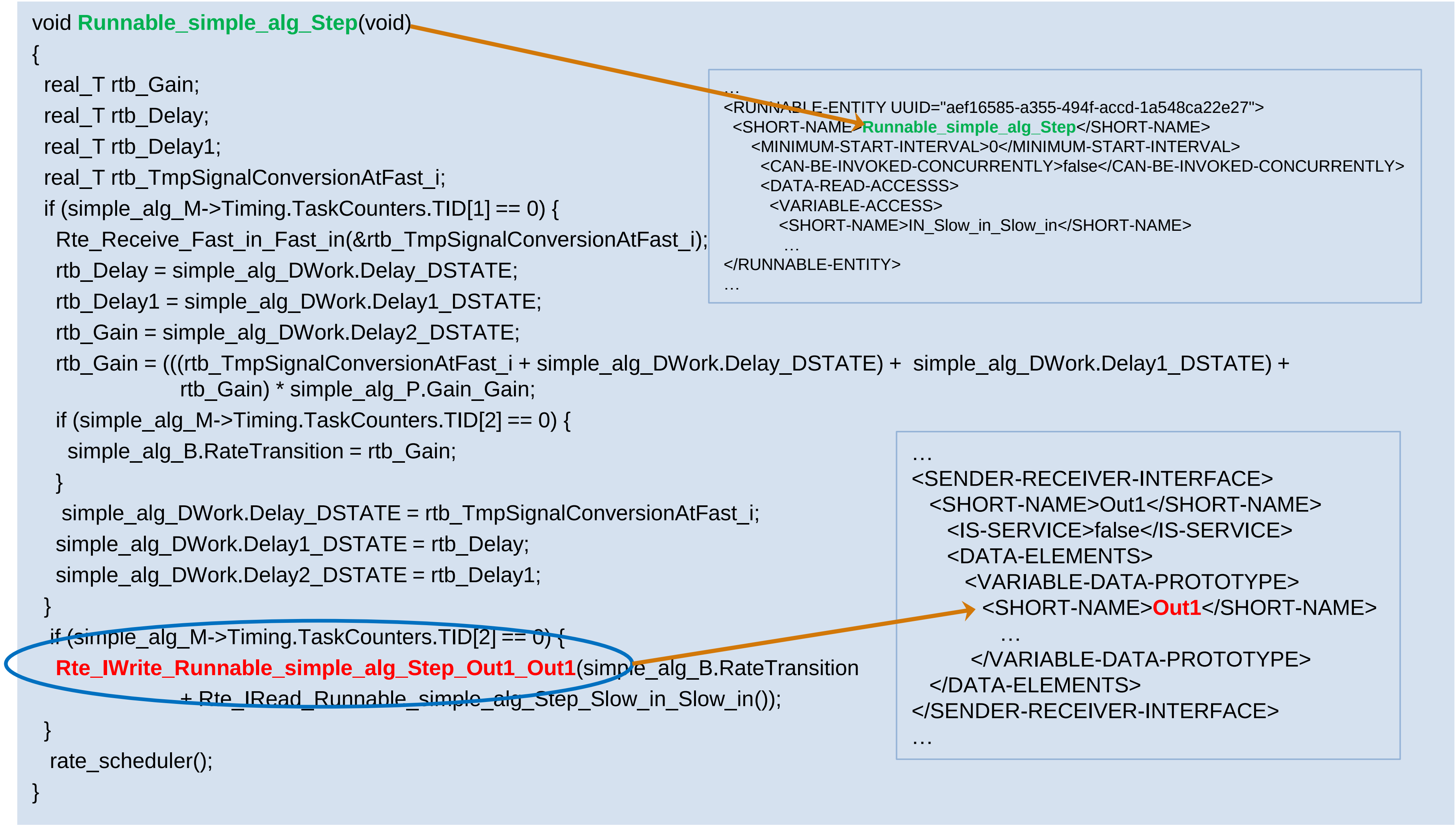
```
%% Setup AUTOSAR Configuration
programmatically

model = 'rtwdemo_autosar_counter';

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

#9 Use production code generation

– Hand coding AUTOSAR is painful (Code and description)



```
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());
  }
   rate_scheduler();
}
```

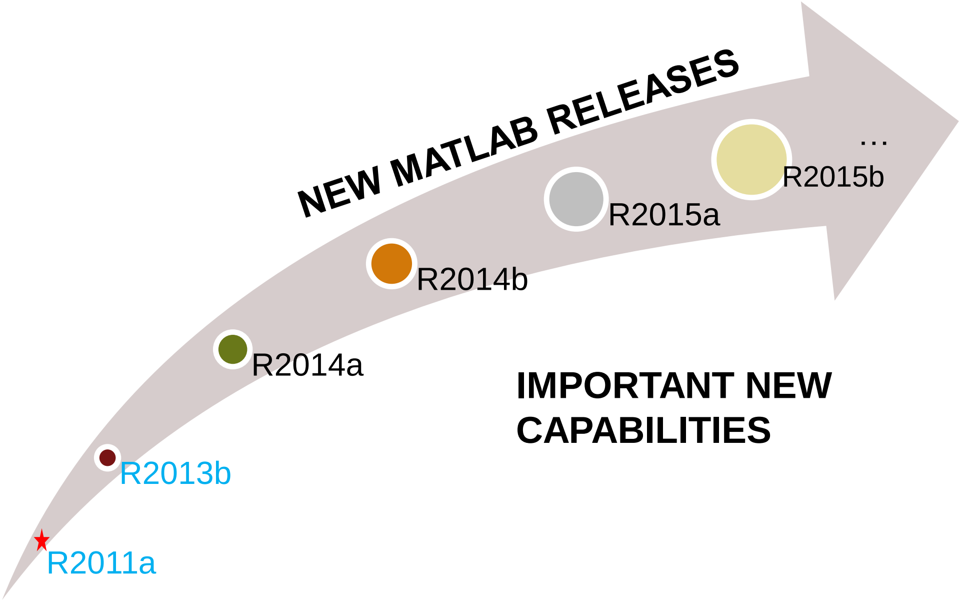
```
…
<RUNNABLE-ENTITY UUID="aef16585-a355-494f-accd-1a548ca22e27">
  <SHORT-NAME>Runnable_simple_alg_Step</SHORT-NAME>
      <MINIMUM-START-INTERVAL>0</MINIMUM-START-INTERVAL>
        <CAN-BE-INVOKED-CONCURRENTLY>false</CAN-BE-INVOKED-CONCURRENTLY>
        <DATA-READ-ACCESSS>
          <VARIABLE-ACCESS>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

#10 Actively plan for migration

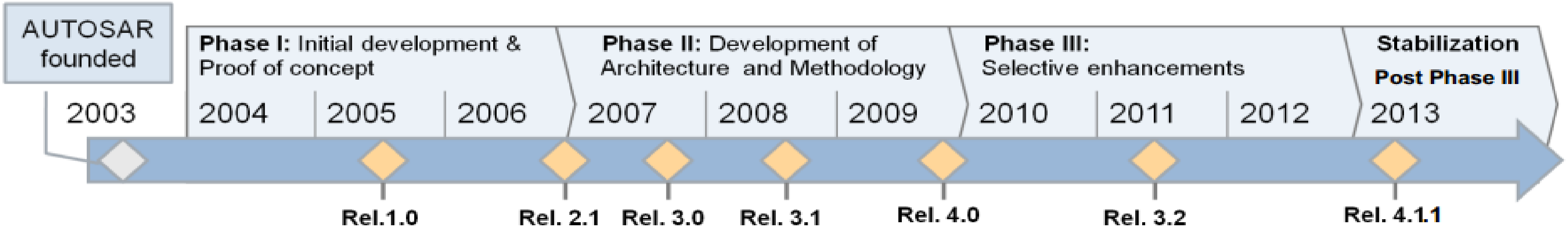
– Tools and standards are changing rapidly

- 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



AUTOSAR founded

Phase I: Initial development & Proof of concept

Phase II: Development of Architecture and Methodology

Phase III: Selective enhancements

Stabilization Post Phase III

Source: AUTOSAR, 6th Open Conference 11.13.2013

***R4.2.1 has been released in 2014**
MATLAB 2015b supports this revision

Best practices for using Simulink with AUTOSAR

- Decide strategy for migrating existing Simulink models to AUTOSAR
- Use one AUTOSAR workflow
- Decide data management
- Establish modeling standard
- Simulate before code generation
- Plan ahead for ISO 26262
- Use Simulink to migrate legacy code to AUTOSAR
- Automate, automate, automate
- Use production code generation
- Actively plan for migration

Agenda

Simulink for AUTOSAR - Introduction

- Workflows
- Capabilities

Simulink for AUTOSAR – User Stories

- Production Code Generation with Embedded Coder

Simulink for AUTOSAR – Best Practices

- Best Practices for using Simulink for AUTOSAR

Summary & Conclusions

Summary

- Simulink and Embedded Coder provide extensive AUTOSAR capabilities out-of-the-box, along with API's for workflow automation
- Leading automotive companies are successfully deploying AUTOSAR for production by leveraging MathWorks tools and industry experience
- Take advantage of best practices for deploying AUTOSAR with Production Code Generation to accelerate your projects while reducing risk and improving quality

Thank you for your attention!

Accelerating the pace of engineering and science

http://www.mathworks.com/products/
http://www.mathworks.com/services/consulting/
http://www.mathworks.com/solutions/automotive/