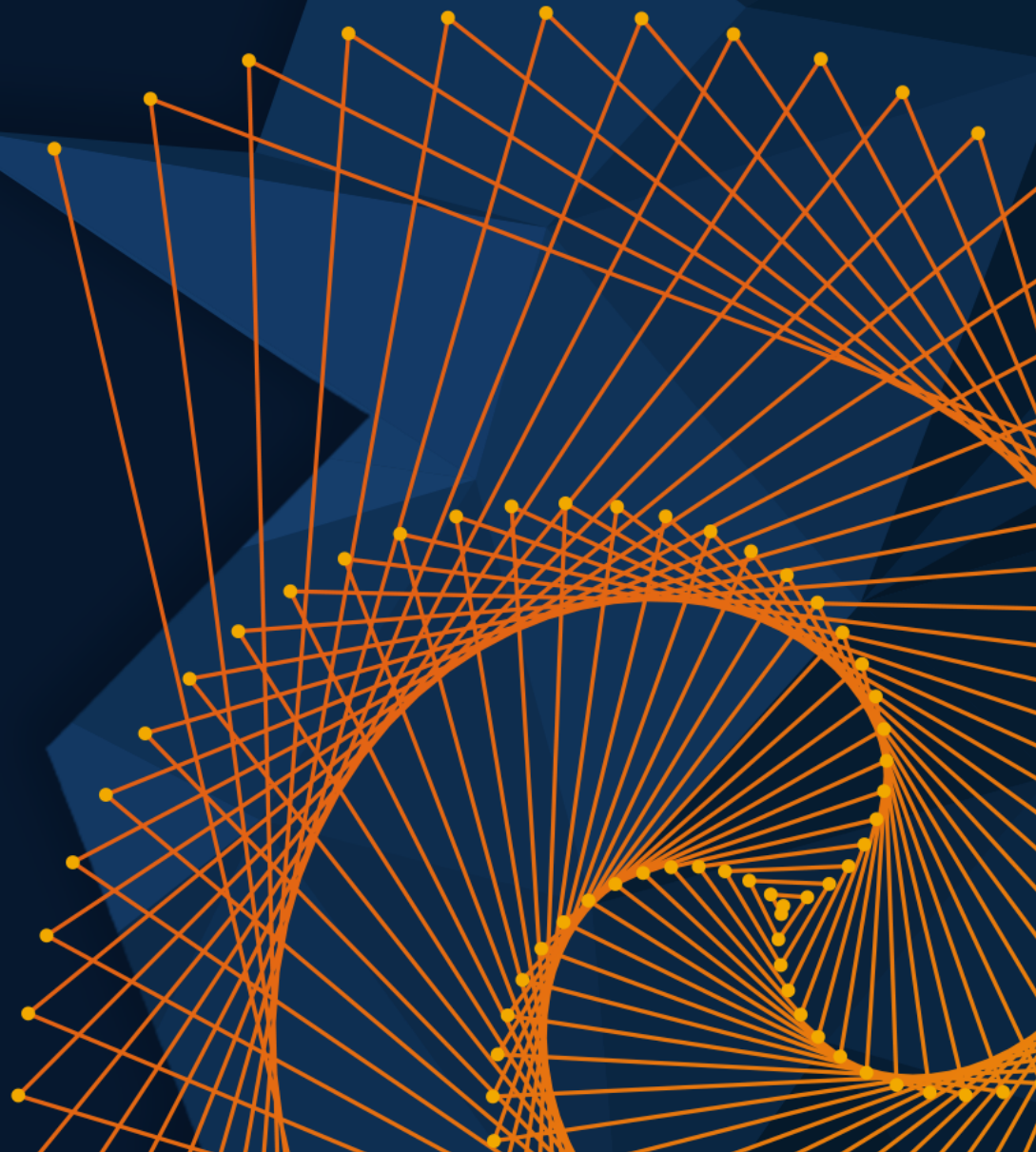
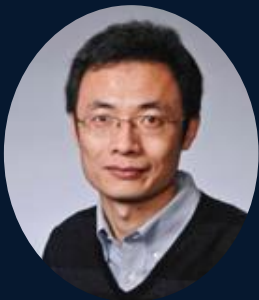


# MATLAB EXPO

5月28日, 2024 | 北京

## 基于Simulink平台的AUTOSAR软件 架构设计和软件实现

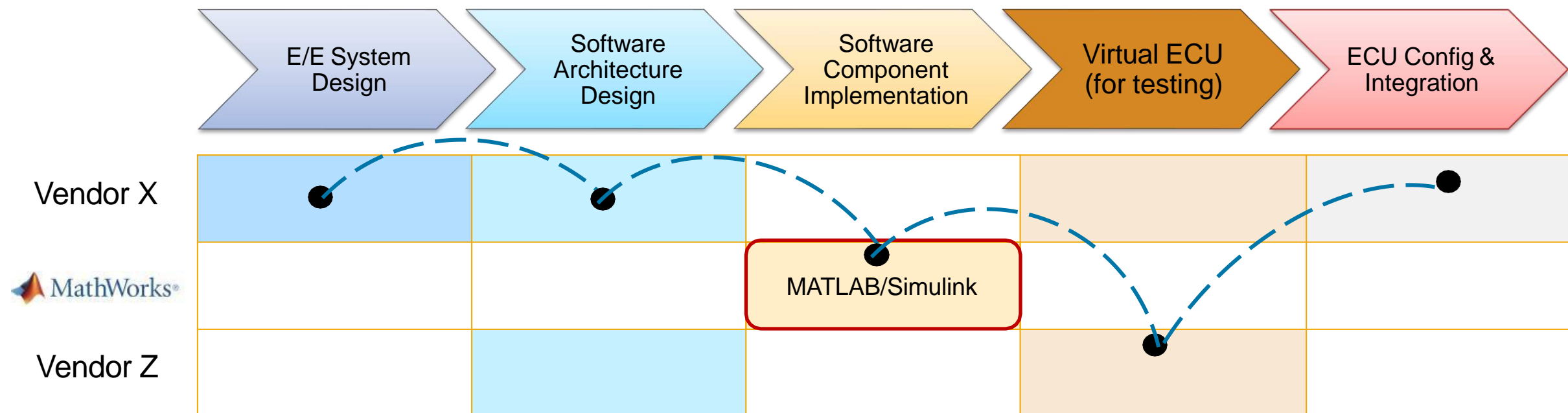
董淑成 MathWorks



# 主要内容

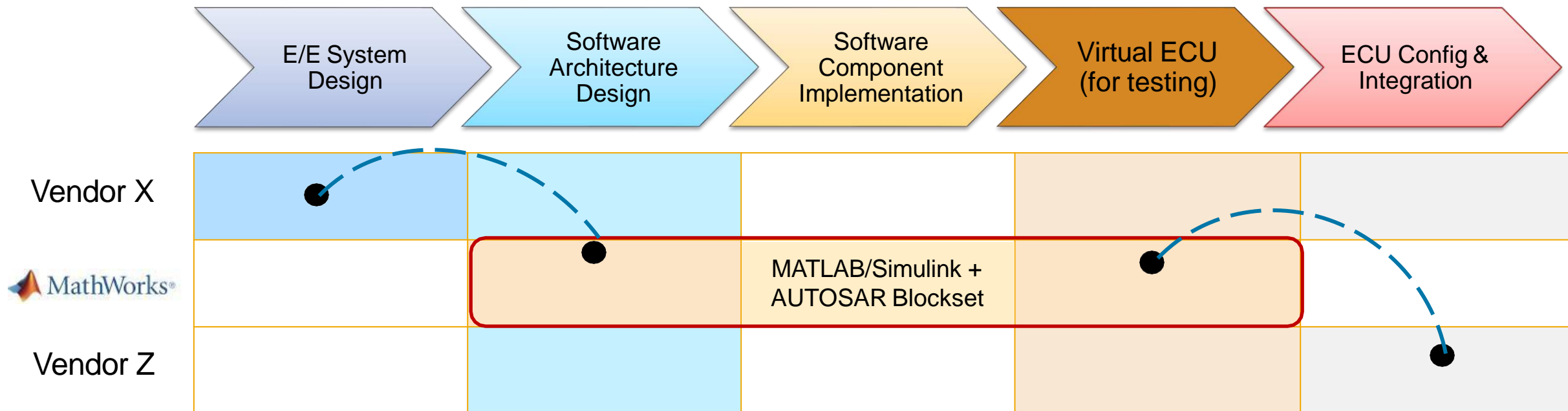
- AUTOSAR解决方案的进化
- System Composer实现软件架构设计
- SWC的Simulink实现
- 基于Excel的AUTOSAR架构设计

# AUTOSAR 软件开发中的挑战



- 过多的交接点影响了开发效率
- 不同工具覆盖不同的标准子集

# AUTOSAR 解决方案 (1)



- 简化端到端的工作流程
- 减少流程中的数据交接
- 提升功能设计交付质量

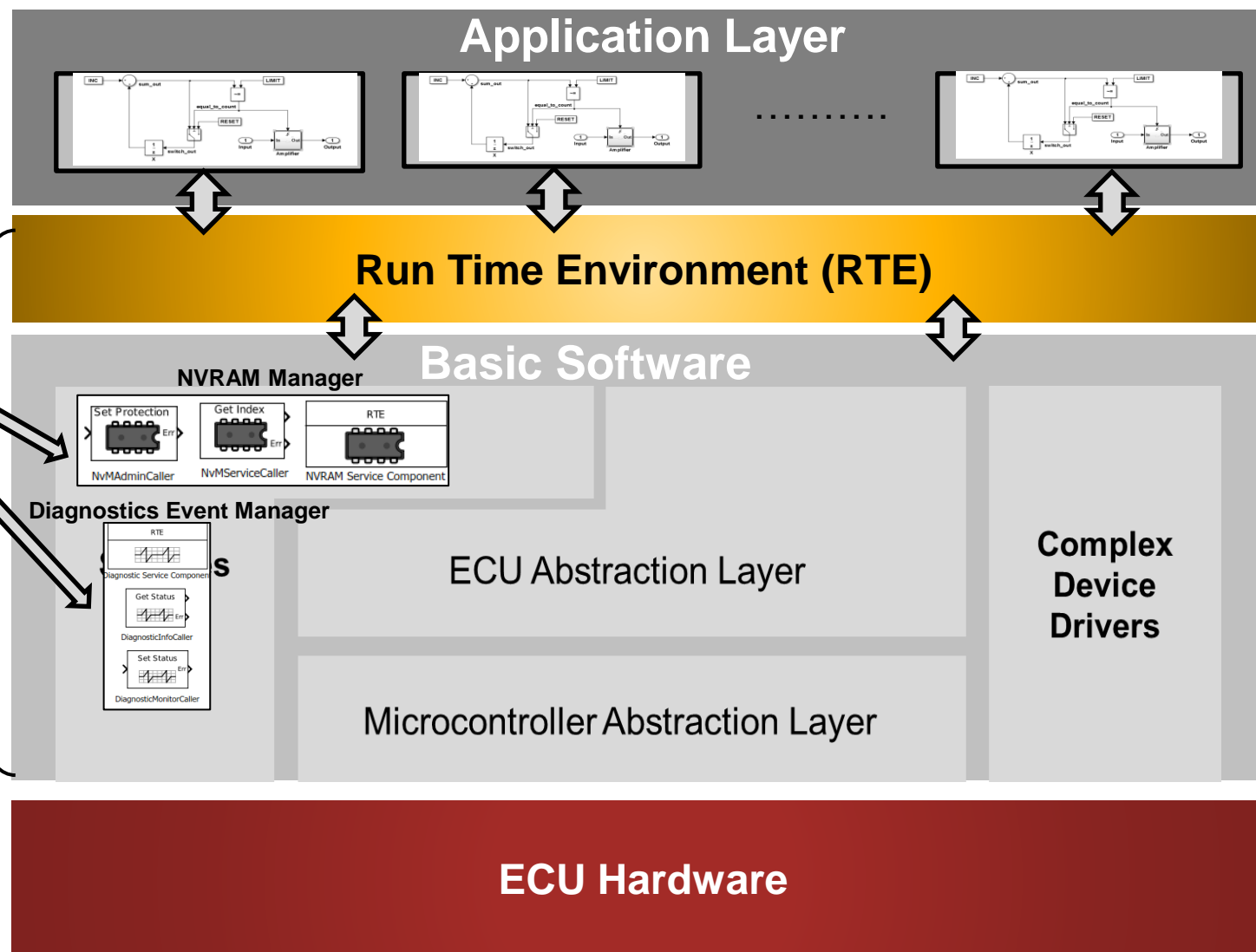
# AUTOSAR 解决方案 (2)

① 软件架构设计

② 行为建模和代码生成

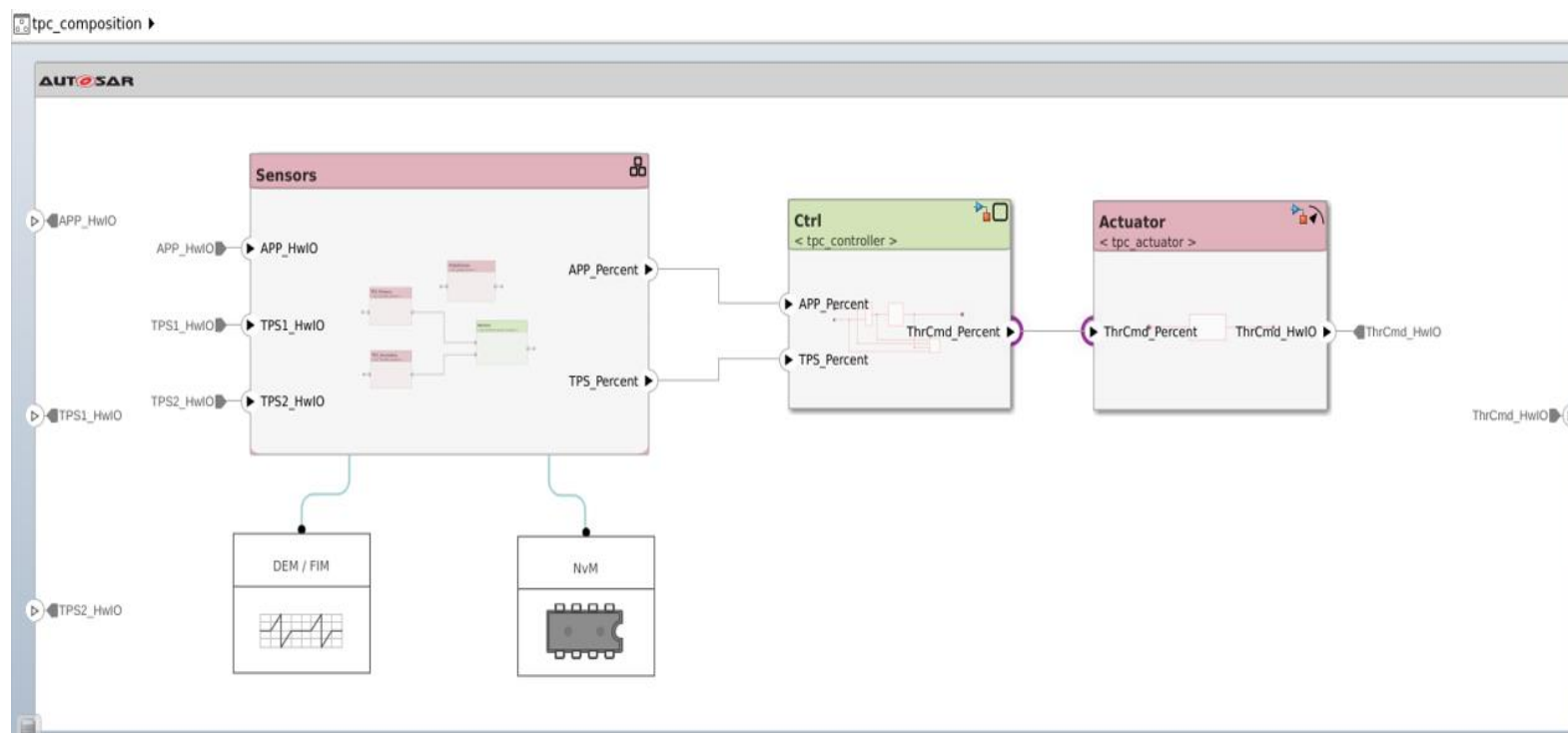
③ 建模和仿真  
BSW Service

BSW配置和  
RTW生成



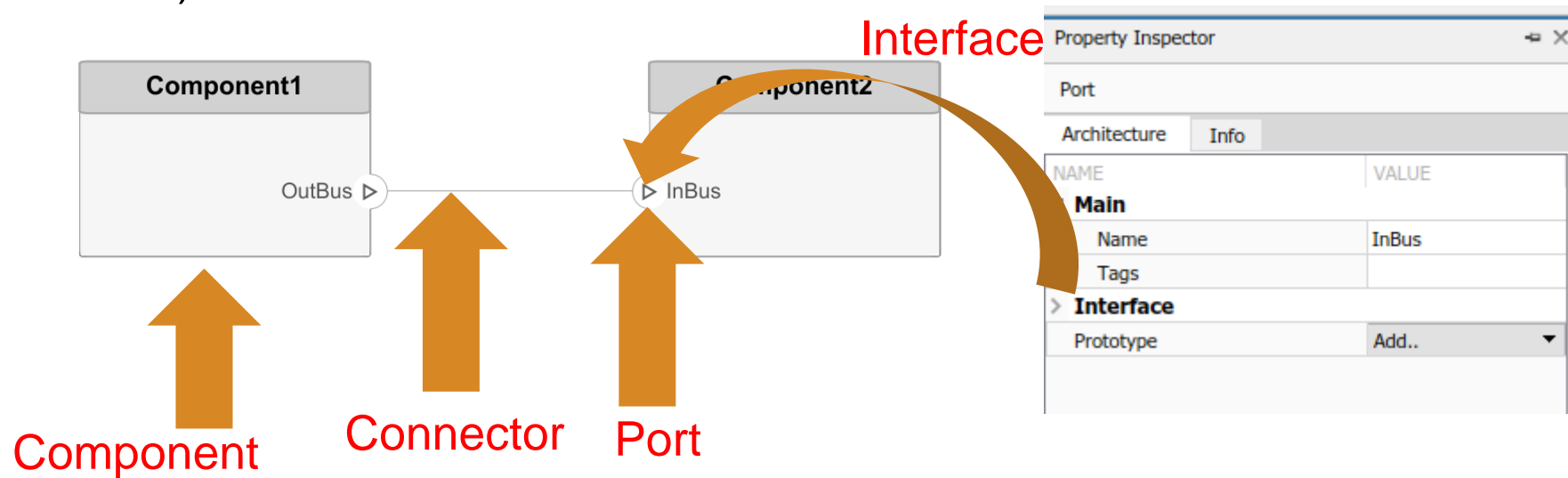
# System Composer实现软件架构设计

- System Composer是系统架构和软件架构设计工具
- System Composer + AUTOSAR Blockset 实现AUTOSAR架构



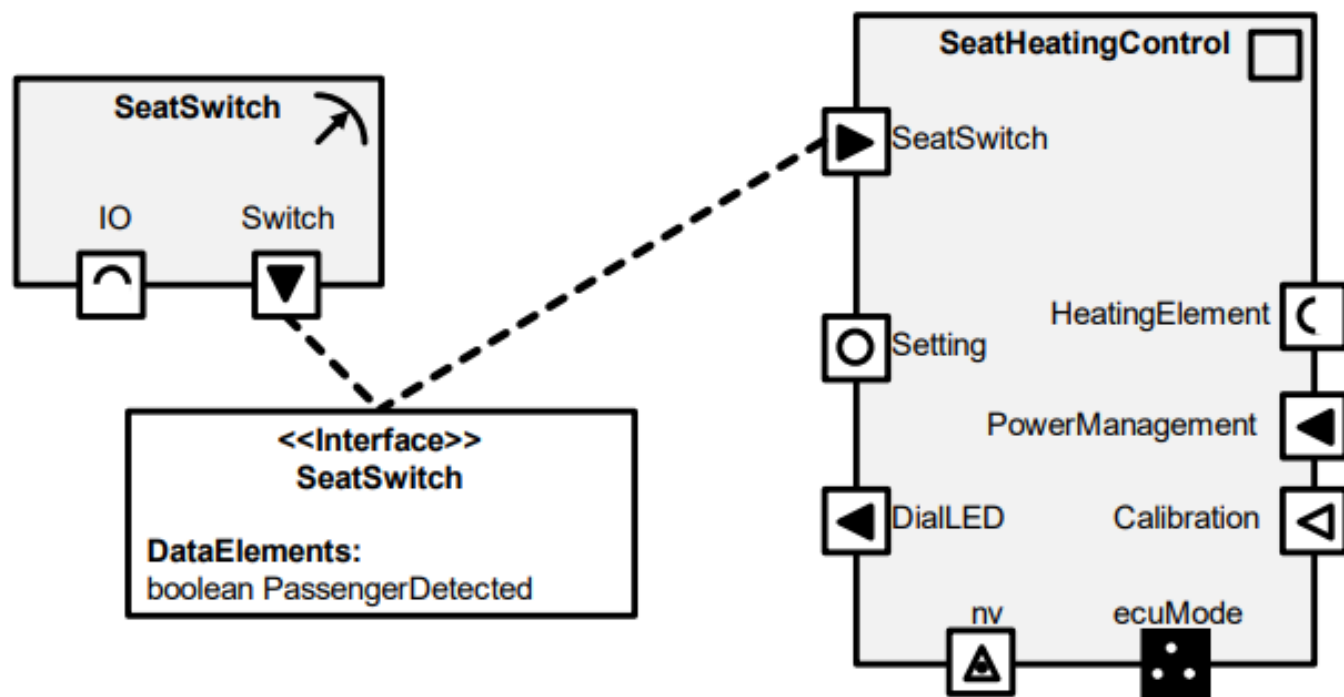
# 基本的架构设计元模型

- 架构设计的基本元素
  - 组件 (Component)
  - 端口 (Port)
  - 接口 (Interface)
  - 连接 (Connector)



# AUTOSAR软件架构的基本元素

- 软件组件 SWC
  - SWC
  - Composition
- 端口 Port
  - PPort
  - RPort
  - PRPort
- 接口 PortInterface
  - Sender-Receiver
  - Client-Server
- 连接 Connector
  - Data transmission
  - Operation invocation





# AUTOSAR 软件架构设计

The screenshot displays the Simulink environment for AUTOSAR software architecture design. The main workspace shows a block diagram with the following components:

- Sensors** (red box): A block containing sub-components for APP, TPS1, and TPS2, with inputs for APP\_HwIO, TPS1\_HwIO, and TPS2\_HwIO, and outputs for APP\_Percent, TPS\_Percent, and ThrCmd\_Percent.
- Ctrl** (green box): A controller block receiving ThrCmd\_Percent and outputting ThrCmd\_HwIO.
- Actuator** (purple box): A block receiving ThrCmd\_HwIO.
- DEM / FIM** (blue box): A block representing the Diagnostic Event Manager / Fault Injection Monitor.
- NvM** (blue box): A block representing Non-Volatile Memory.

Annotations and tool windows include:

- Simulate** (blue arrow): Points to the Simulate button in the top toolbar.
- Export** (orange box): A dialog window for exporting the composition, with "Export ECU extract" checked.
- Implement internal behavior** (red text): Points to a detailed sub-diagram of the controller's internal logic.
- R2022b Author Runnables** (green box): Points to the Functions Editor window.
- Interfaces** (purple box): A window listing the interfaces used in the model, including APP\_Percent, TPS\_HwIO, TPS\_Percent, ThrCmd\_HwIO, ThrCmd\_Percent, and hwio\_t.
- R2022b Author interfaces** (purple box): Points to the Interfaces window.

Execution Order	Function Name	Software Component	Period
1	PedalSensor_D1	Sensors/PedalSensor	0.005
2	TPS_Primary_D1	Sensors/TPS_Primary	0.005
3	TPS_Secondary_D1	Sensors/TPS_Secondary	0.005
4	Monitor_D1	Sensors/Monitor	0.005
5	Ctrl_D1	Ctrl	0.005
6	Actuator_D1	Actuator	0.005

Author Components & Composition hierarchy

Simulate calls to BSW

Author interfaces  
**R2022b**

Generate code and arxml

# AUTOSAR 软件架构设计 —— Runnable 定义

The screenshot displays the Simulink environment for AUTOSAR software architecture design. The main workspace shows a block diagram with components like Sensors, Ctrl, and Actuator. Annotations include:

- Author Components & Composition hierarchy:** A red box highlights the Sensors block, and a red arrow points to the Model Browser on the left.
- Simulate calls to BSW:** A blue arrow points from the Simulate button to the DEM / FIM and NvM blocks.
- Implement internal behavior:** A red box highlights the Ctrl block, with an inset showing its internal logic.
- Author interfaces R2022b:** A purple box highlights the Interfaces window, listing APP\_Percent, TPS\_HwIO, TPS\_Percent, ThrCmd\_HwIO, and ThrCmd\_Percent.
- R2022b Author Runnables:** A green box highlights the Functions Editor window, showing a table of functions and their periods.
- Generate code and arxml:** An orange box highlights the Export Composition dialog, with the 'Export ECU extract' option checked.

Execution Order	Function Name	Software Component	Period
1	PedalSensor_D1	Sensors/PedalSensor	0.005
2	TPS_Primary_D1	Sensors/TPS_Primary	0.005
3	TPS_Secondary_D1	Sensors/TPS_Secondary	0.005
4	Monitor_D1	Sensors/Monitor	0.005
5	Ctrl_D1	Ctrl	0.005
6	Actuator_D1	Actuator	0.005

# Function Editor —— 为 SWC 添加 Runnable

The screenshot shows the MATLAB Function Editor interface for a software component diagram. The diagram consists of two software components, SWC1 and SWC2, connected by a port labeled PPort and RPort. The Functions Editor panel is open, showing a table with columns for Execution Order, Function Name, Software Component, and Period. The table is currently empty, displaying "No data to display".

Execution Order	Function Name	Software Component	Period
No data to display			

Interfaces  
Ready 150% FixedStepDiscrete

# AUTOSAR 软件架构设计 —— 接口定义

The screenshot displays the Simulink environment for AUTOSAR software architecture design. The main workspace shows a block diagram with components like Sensors, Ctrl, and Actuator. Annotations include:

- Author Components & Composition hierarchy:** A red box highlights the Sensors component and its internal sub-components.
- Simulate calls to BSW:** A blue box highlights the DEM / FIM and NvM components.
- Implement internal behavior:** A red box highlights the internal logic of the Ctrl component.
- R2022b Author Runnables:** A green box highlights the Functions Editor window.
- Generate code and arxml:** An orange box highlights the Export Composition dialog.
- Author interfaces R2022b:** A purple box highlights the Interfaces window.

The Functions Editor window shows the following table of functions:

Execution Order	Function Name	Software Component	Period
1	PedalSensor_D1	Sensors/PedalSensor	0.005
2	TPS_Primary_D1	Sensors/TPS_Primary	0.005
3	TPS_Secondary_D1	Sensors/TPS_Secondary	0.005
4	Monitor_D1	Sensors/Monitor	0.005
5	Ctrl_D1	Ctrl	0.005
6	Actuator_D1	Actuator	0.005

Author Components & Composition hierarchy

Simulate calls to BSW

Implement internal behavior

R2022b Author Runnables

Generate code and arxml

Author interfaces R2022b

# Interface Editor —— 定义 AUTOSAR 接口

The screenshot displays the MATLAB/Simulink Interface Editor for an AUTOSAR model named "ThrottlePositionControlComposition". The main workspace shows a block diagram with the following components and connections:

- ThrottlePositionSensor**: Receives `TPS_HwIO` and outputs `TPS_Percent`.
- AccelPedalPositionSensor**: Receives `APP_HwIO` and outputs `APP_Percent`.
- Controller**: Receives `TPS_Percent` and `APP_Percent` as inputs and outputs `ThrCmd_Percent`.
- Actuator**: Receives `ThrCmd_Percent` and outputs `ThrCmd_HwIO`.

The **Property Inspector** on the right shows the following details for the selected `ThrottlePositionSensor` component:

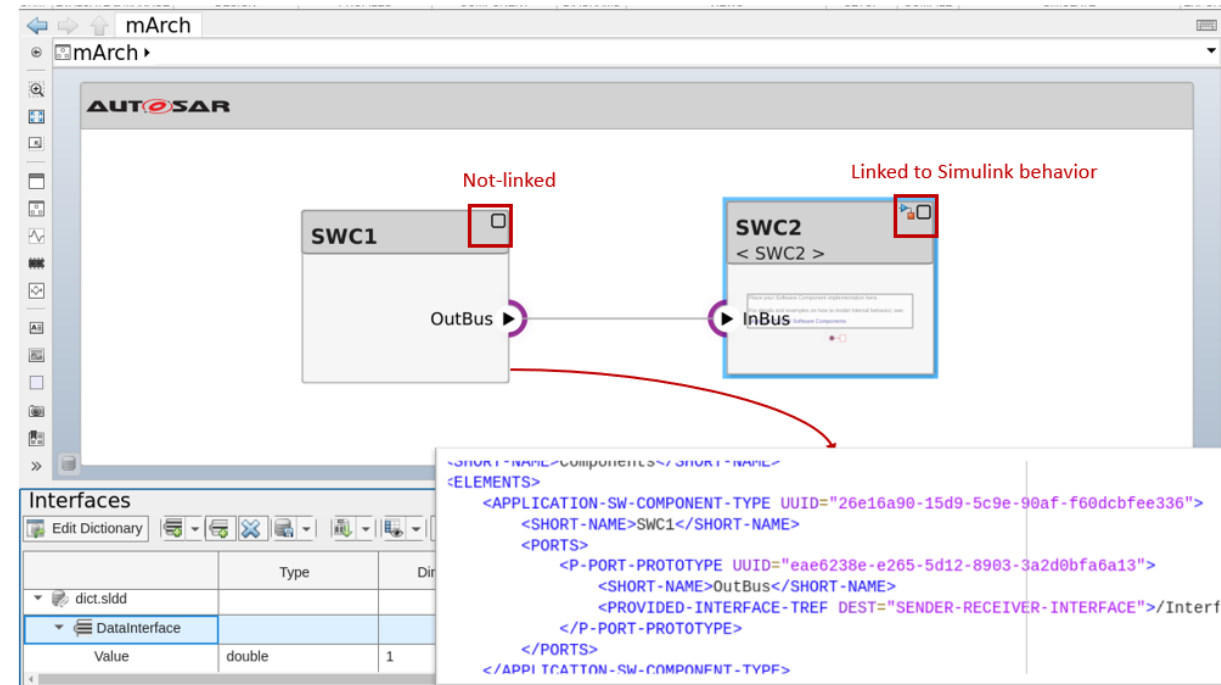
Component	
Architecture Info	
NAME	VALUE
<b>Main</b>	
Name	ThrottlePositionSensor
Kind	SensorActuator
Stereotype	Add..
<b>Parameters</b>	Select

The **Interfaces** panel at the bottom shows the current interface dictionary for "ThrottlePositionControlComposition.slx".

# 为空组件导出 ARXML 文件

Export non-linked components from AUTOSAR architecture model

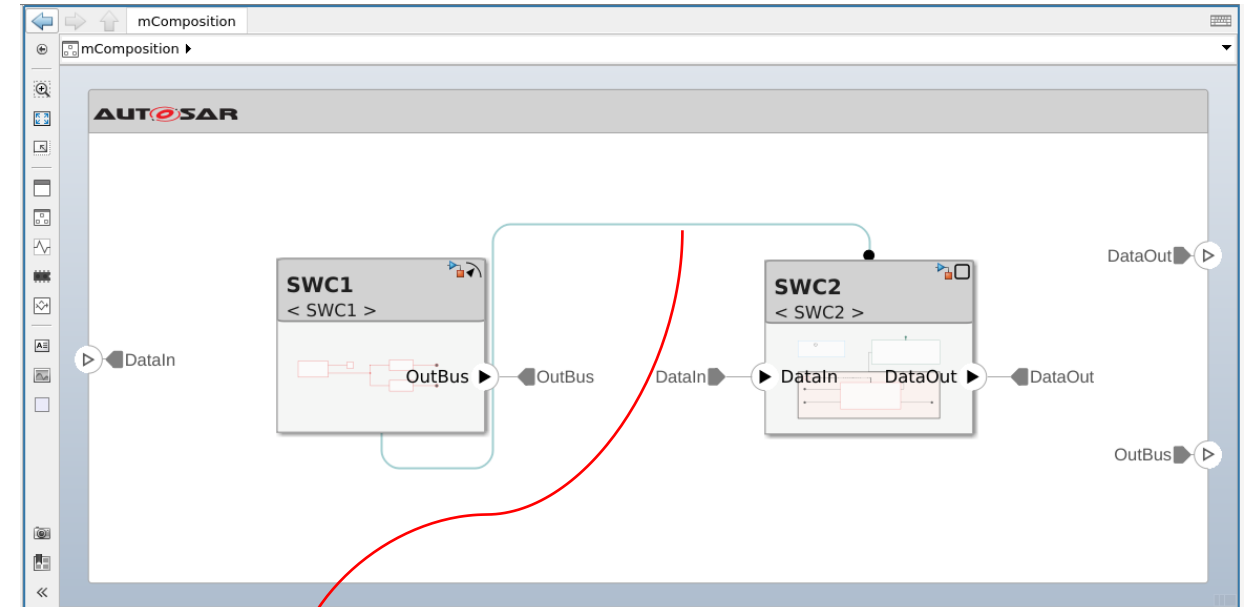
- Export component blocks even when they are not linked to Simulink behavior model to ARXML
- Export non-linked Components ports and interfaces
- Allows incremental workflows where internal behavior can be specified later



# 导出 C/S Connectors 到 ARXML

Export Client Server Connectors from AUTOSAR architecture model

- Export to ARXML the client-server connectors describing call between Function Callers and global Simulink Functions
- Use command line API with `ExportUnmodeledClientServerConnectors` argument
- Supports calls across composition hierarchy. Requires client/server interface names to match in the AUTOSAR dictionary.



```
>> archModel = autosar.arch.loadModel(modelName);
>> archModel.export('ExportUnmodeledClientServerConnectors', true);
```

```
<ASSEMBLY-SW-CONNECTOR UUID="8c7fcc77-e1d1-5d5c-2354-59627e3c3c20">
  <SHORT-NAME>SWC2_SrvPort_SWC1_readData_client</SHORT-NAME>
  <PROVIDER-IREF>
    <CONTEXT-COMPONENT-REF DEST="SW-COMPONENT-PROTOTYPE"/>/Components/mComposition/SWC2</C
    <TARGET-P-PORT-REF DEST="P-PORT-PROTOTYPE"/>/Components/ASWC/SrvPort</TARGET-P-PORT-RE
  </PROVIDER-IREF>
  <REQUESTER-IREF>
    <CONTEXT-COMPONENT-REF DEST="SW-COMPONENT-PROTOTYPE"/>/Components/mComposition/SWC1</C
    <TARGET-R-PORT-REF DEST="R-PORT-PROTOTYPE"/>/Components/Component1/readData_client</TA
  </REQUESTER-IREF>
</ASSEMBLY-SW-CONNECTOR>
```

# 导出 C/S Connectors 到 ARXML (2)

```

1 % Open example model autosar_tpc_composition for reference
2 open_system('autosar_tpc_composition')
3 archModel = autosar.arch.loadModel('autosar_tpc_composition');
4 archModel.export('PackageCodeAndARXML', 'R2023a_cnctor_autosar_tpc_composit:
5 'ExportECUExtract', true, ...
6 'ExportUnmodeledClientServerConnectors', true);

```

The image shows two side-by-side windows of an ARXML file, illustrating the result of exporting unmodeled client-server connectors. The left window shows the original ARXML with a shaded region. The right window shows the modified ARXML with a new connector added in red.

**Left Window (Original ARXML):**

```

</PROVIDER-IREF>
<REQUESTER-IREF>
  <CONTEXT-COMPONENT-REF DEST="SW-COMPONENT-PRO
  <TARGET-R-PORT-REF DEST="R-PORT-PROTOTYPE"/>/C
</REQUESTER-IREF>
</ASSEMBLY-SW-CONNECTOR>
<DELEGATION-SW-CONNECTOR UUID="cbd69324-71c8-55dc-330
  <SHORT-NAME>Actuator_ThrCmd_HwIO_ThrCmd_HwIO</SHO
  <INNER-PORT-IREF>
    <P-PORT-IN-COMPOSITION-INSTANCE-REF>
      <CONTEXT-COMPONENT-REF DEST="SW-COMPONENT
      <TARGET-P-PORT-REF DEST="P-PORT-PROTOTYPE
    </P-PORT-IN-COMPOSITION-INSTANCE-REF>
  </INNER-PORT-IREF>
  <OUTER-PORT-REF DEST="P-PORT-PROTOTYPE"/>/Componen
</DELEGATION-SW-CONNECTOR>
<DELEGATION-SW-CONNECTOR UUID="9811e4cf-14ea-52e8-634
  <SHORT-NAME>TPS_Percent_Ctr1_TPS_Percent</SHORT-N

```

**Right Window (Modified ARXML):**

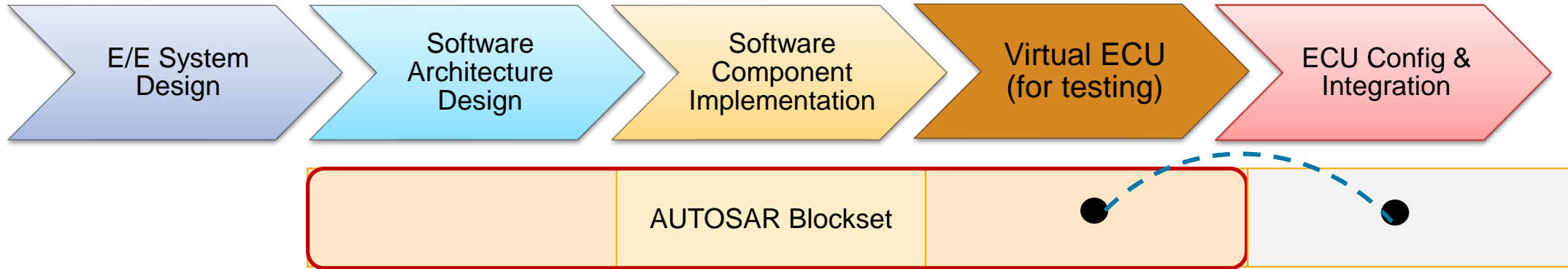
```

</PROVIDER-IREF>
<REQUESTER-IREF>
  <CONTEXT-COMPONENT-REF DEST="SW-COMPONENT-PRO
  <TARGET-R-PORT-REF DEST="R-PORT-PROTOTYPE"/>/C
</REQUESTER-IREF>
</ASSEMBLY-SW-CONNECTOR>
<ASSEMBLY-SW-CONNECTOR UUID="ce5602fe-25ea-5426-a9f2-
  <SHORT-NAME>Ctrl_my_func_Actuator_my_func_client<
  <PROVIDER-IREF>
    <CONTEXT-COMPONENT-REF DEST="SW-COMPONENT-PRO
    <TARGET-P-PORT-REF DEST="P-PORT-PROTOTYPE"/>/C
  </PROVIDER-IREF>
  <REQUESTER-IREF>
    <CONTEXT-COMPONENT-REF DEST="SW-COMPONENT-PRO
    <TARGET-R-PORT-REF DEST="R-PORT-PROTOTYPE"/>/C
  </REQUESTER-IREF>
</ASSEMBLY-SW-CONNECTOR>
<DELEGATION-SW-CONNECTOR UUID="99a1f02f-6de7-5ab0-fe5
  <SHORT-NAME>Actuator_ThrCmd_HwIO_ThrCmd_HwIO</SHO
  <INNER-PORT-IREF>
    <P-PORT-IN-COMPOSITION-INSTANCE-REF>
      <CONTEXT-COMPONENT-REF DEST="SW-COMPONENT
      <TARGET-P-PORT-REF DEST="P-PORT-PROTOTYPE
    </P-PORT-IN-COMPOSITION-INSTANCE-REF>
  </INNER-PORT-IREF>
  <OUTER-PORT-REF DEST="P-PORT-PROTOTYPE"/>/Componen
</DELEGATION-SW-CONNECTOR>
<DELEGATION-SW-CONNECTOR UUID="f088e0ab-e817-5fa8-d57
  <SHORT-NAME>TPS_Percent_Ctr1_TPS_Percent</SHORT-N

```

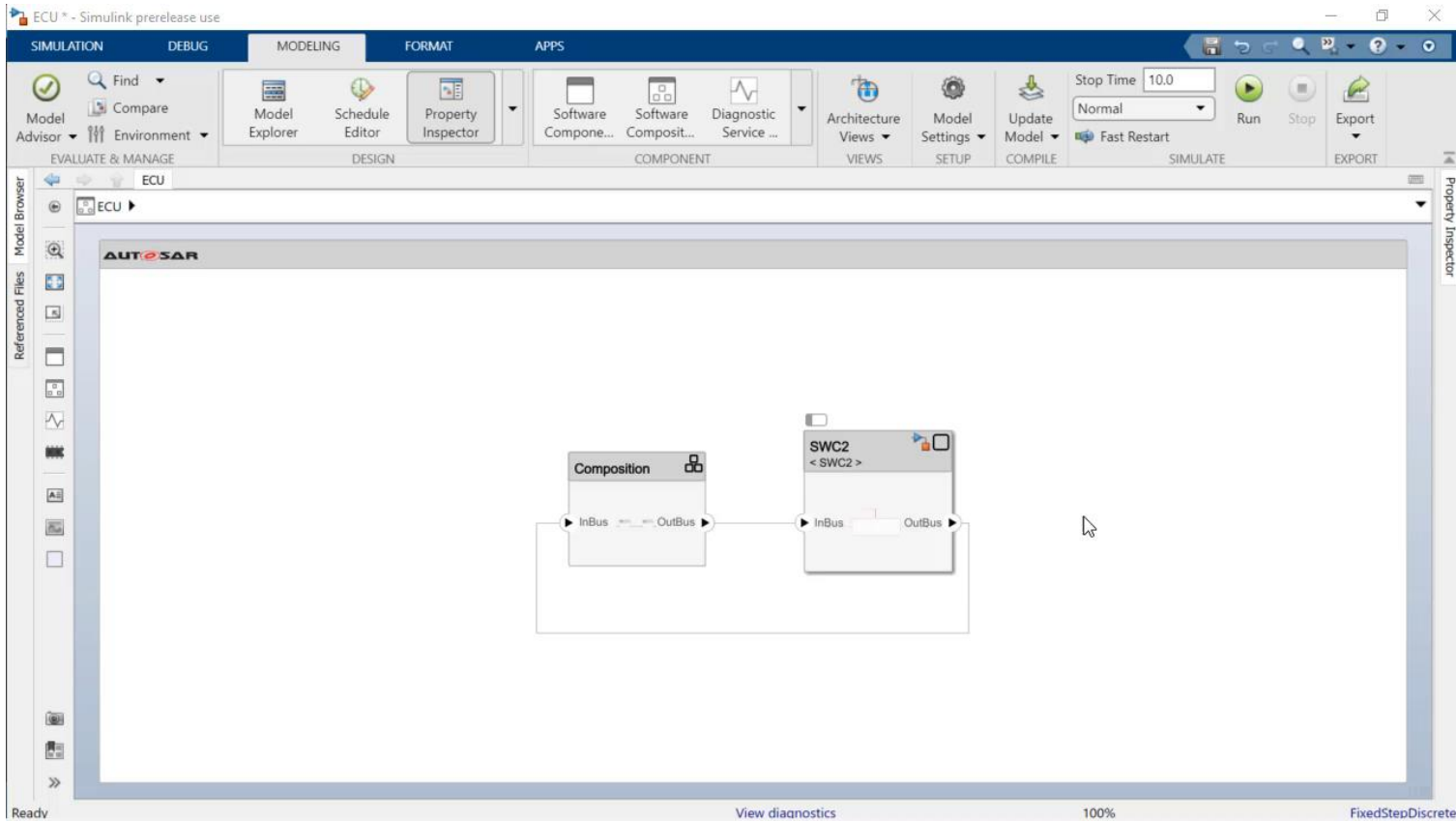


## 导出 ECU extract 与 timing ARXML



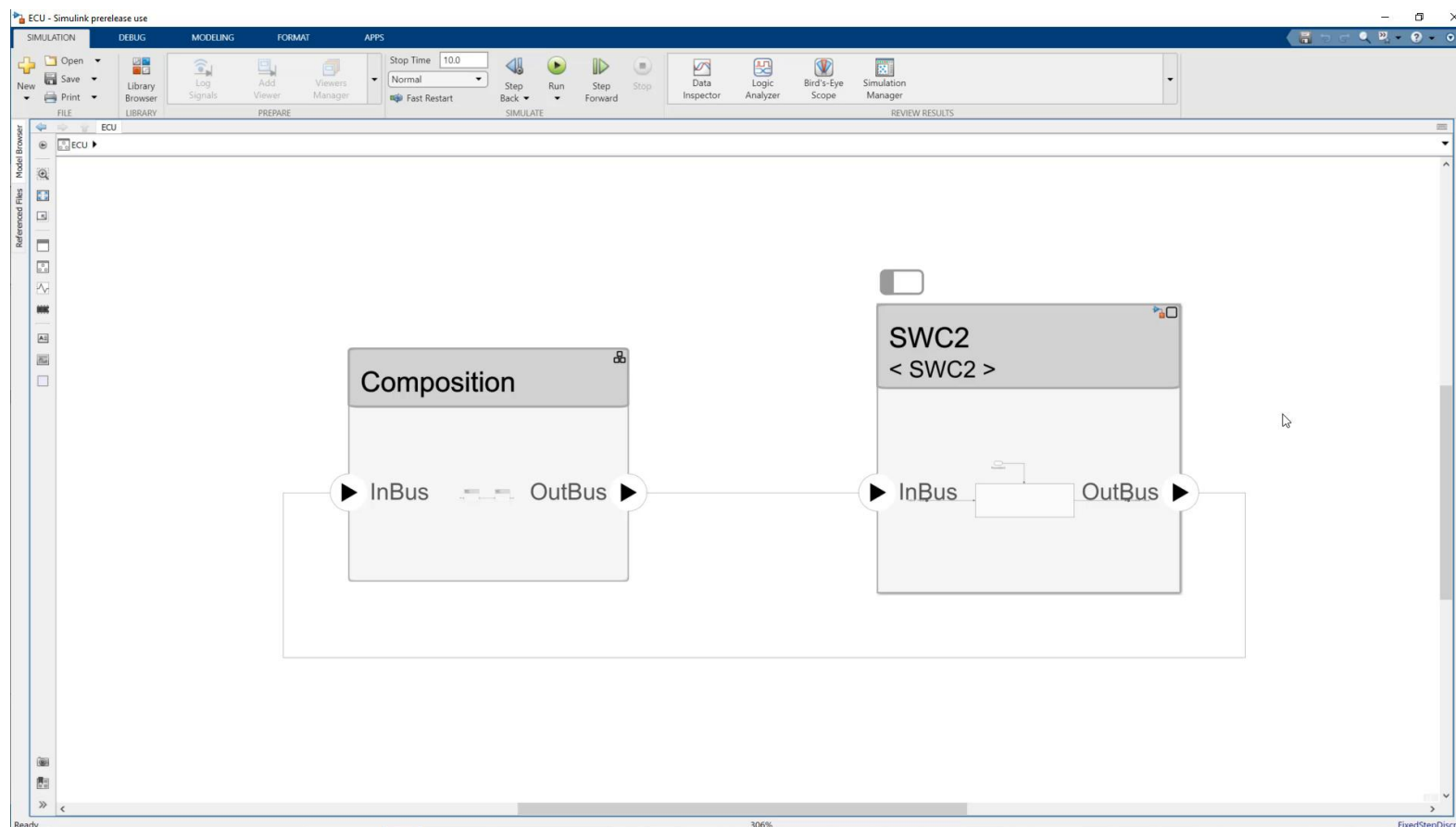
- Exporting an ECU extract – sets you up for AUTOSAR ECU configuration
- Exporting a timing model – guide runnable to task mapping

# 导出 ECU extract ARXML 文件示例



- ✓ Export flattened topology
- ✓ Automatic mapping of SWCs to an ECU
- ✓ Sets you up for AUTOSAR ECU configuration

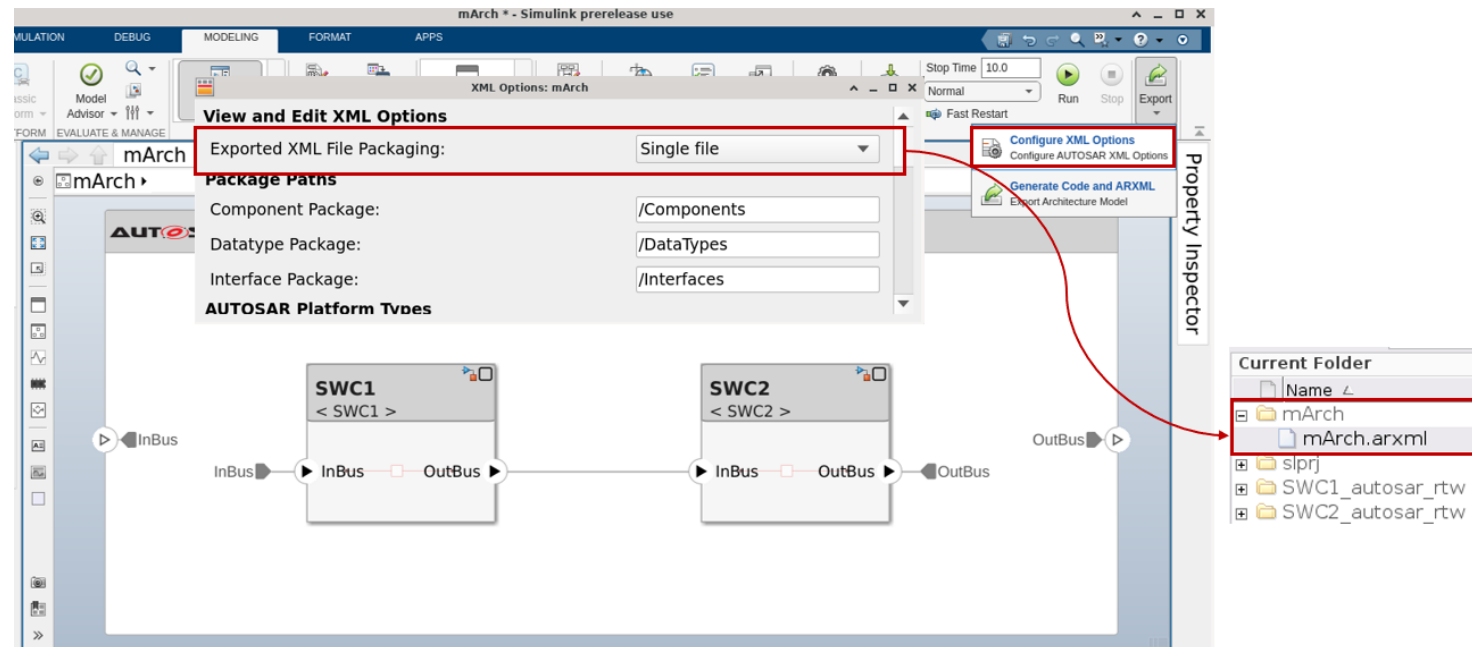
## 导出 timing ARXML 文件示例



- ✓ Exporting a timing model
- ✓ Guide runnable to task mapping
- ✓ Preserve simulated behaviour

# 把 SW-Composition 导出为单个 ARXML 文件

- Control file packaging for exported ARXML from AUTOSAR architecture model
- Support both SingleFile and Modular ARXML file packaging
- Use XmlOptions in AUTOSAR architecture model toolstrip to specify packaging



# 导入/导出/编辑 ASIL 属性

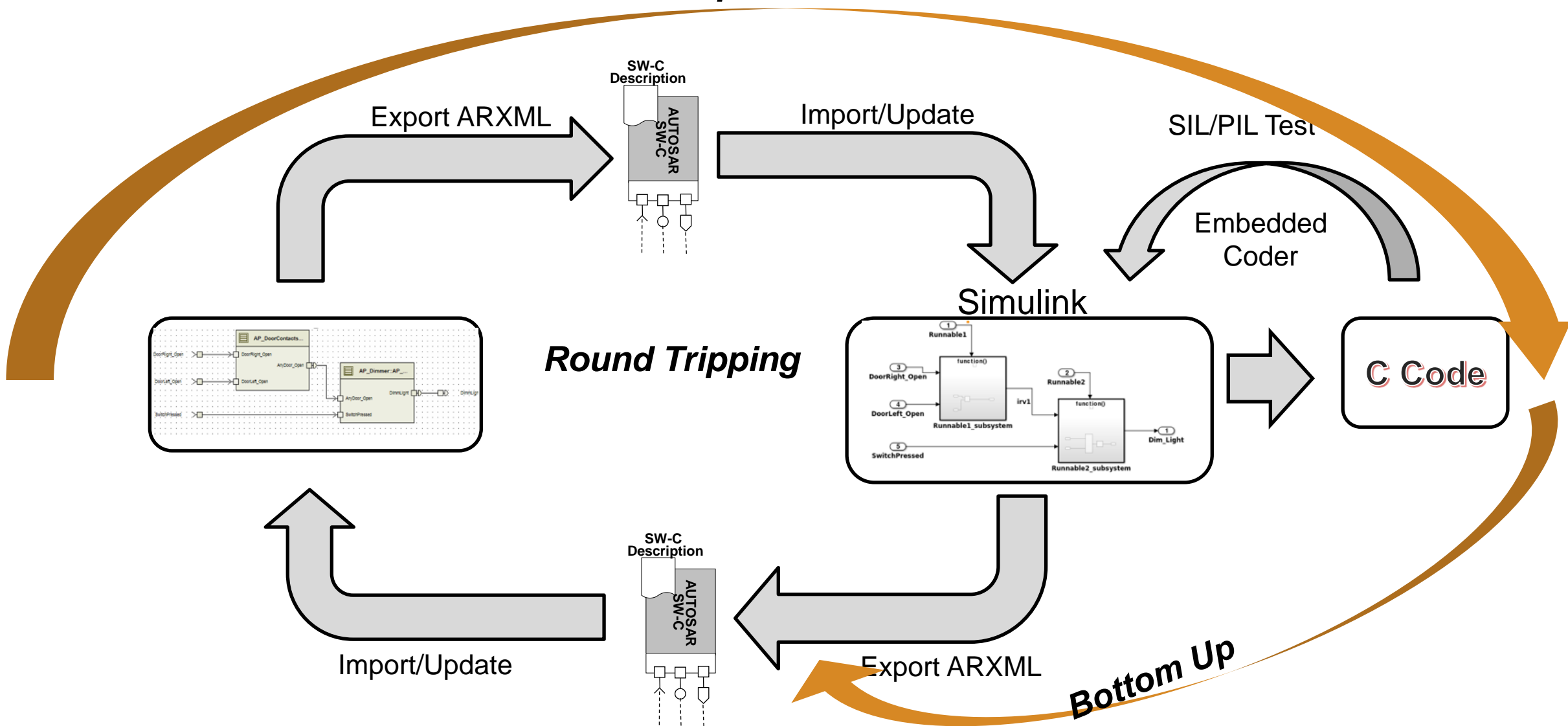
Export and Import ASIL information from AUTOSAR architecture model

- Specify ASIL safety level (QM, A, B, C, D) for Composition blocks, Component blocks and Ports
- Validation will flag any inconsistencies when outport has higher safety level than owning atomic component
- Information is exported to AUTOSAR AdminData in arxml per AUTOSAR\_TPS\_SafetyExtensions.pdf
- Import from ARXML to AUTOSAR architecture model will also import ASIL information

The screenshot displays the AUTOSAR architecture model editor. The main workspace shows two software components, SWC1 and SWC2, connected by an interface with 'OutBus' and 'InBus' ports. The 'Property Inspector' window on the right shows the 'SafetyLevel' property for SWC2 set to 'D'. Below the diagram, a snippet of ARXML code is shown, with a red box highlighting the ASIL-related XML tags: `<SDG GID='SAFEX'> <SD GID='ASIL'>D</SD> </SDG>`. A red arrow points from the 'D' value in the Property Inspector to the corresponding XML tag in the code snippet.

# 支持 AUTOSAR 3 种 workflow 开发模式

*Top down*



# AUTOSAR 架构设计 workflow —— ARXML 导出/导入 workflow

2. 导入 ARXML 文件到 System Composer

```
% 打开示例
% openExample('autosarblockset/ImportAUTOSARCompositionIntoArchitectureModelExample', 'workdir', pwd);

% open('ImportAUTOSARCompositionIntoArchitectureModelExample.mlx');

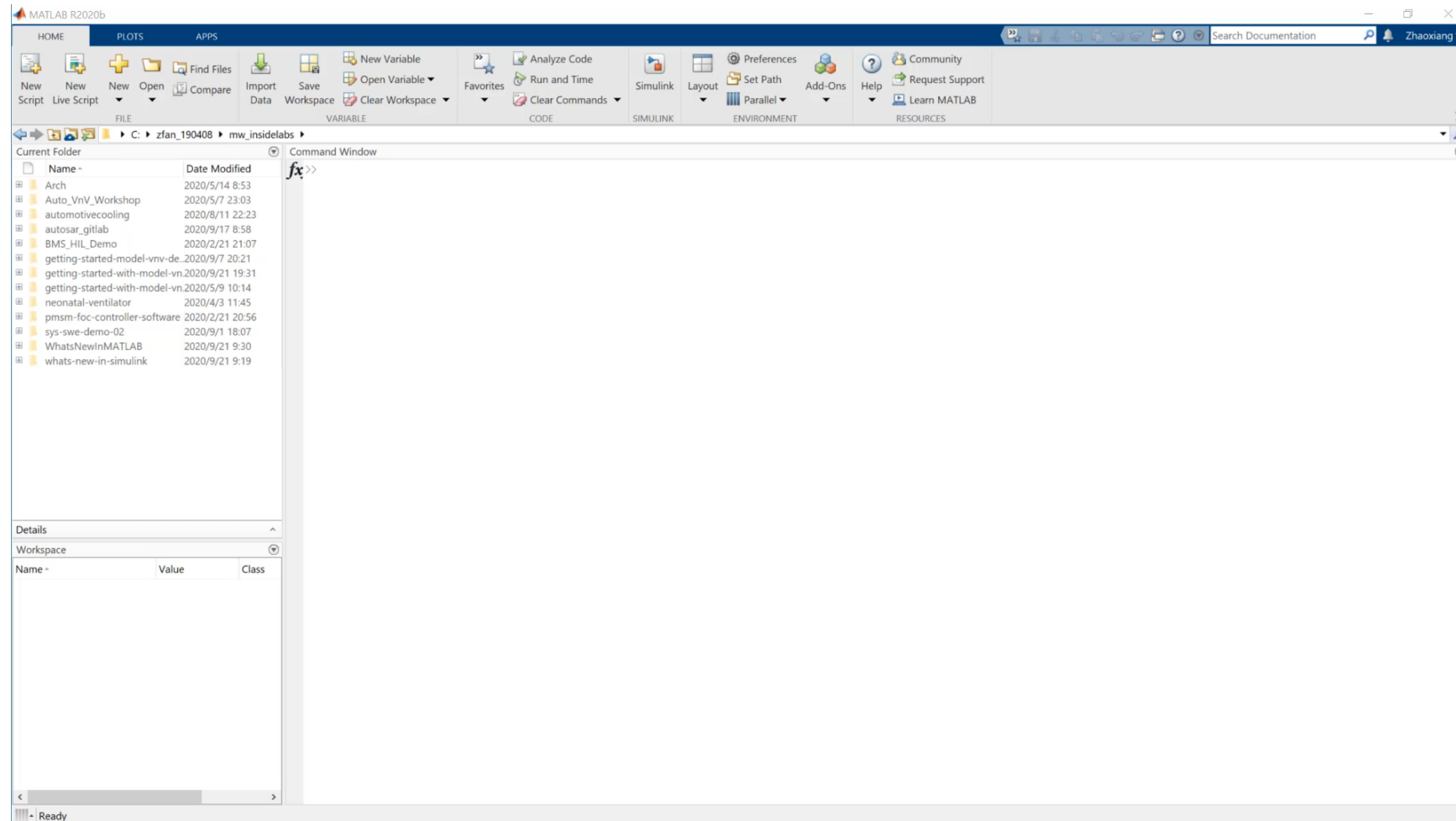
% Create AUTOSAR architecture model
modelName = "myArchModel";
archModel = autosar.arch.createModel(modelName)

% Import composition from file ThrottlePositionControlComposition.arxml
importerObj = arxml.importer("ThrottlePositionControlComposition.arxml") % Parse ARXML
importFromARXML(archModel, importerObj, ...
    "/Company/Components/ThrottlePositionControlComposition")
```

Name	Value
archModel	1x1 Model
importerObj	1x1 importer
modelName	"myArchMod..."
out	1x1 Simulatio...

2. 导入 ARXML 文件到 System Composer

# SWC 实现 —— AUTOSAR自上而下(*Top down*)工作流演示





# SWC 实现 —— AUTOSAR自下而上(*Bottom Up*) workflow 演示

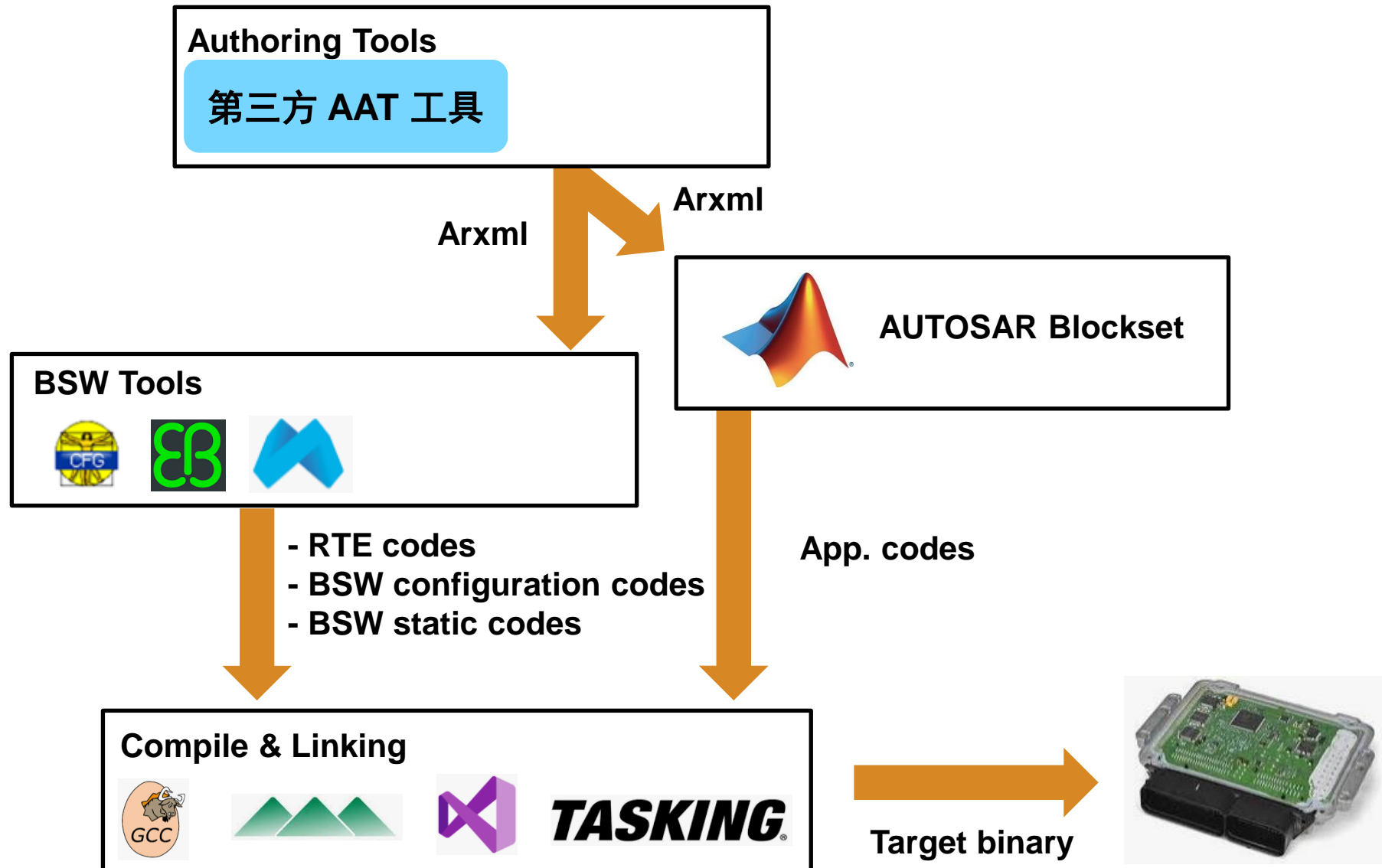
The screenshot shows the MATLAB documentation interface for the AUTOSAR Blockset. The page title is "AUTOSAR Blockset — Examples". The main content area is titled "Get Started with AUTOSAR Blockset" and features three cards with live script links:

- Create and Configure AUTOSAR Software Component**: Create an AUTOSAR software component model from an algorithm model. [Open Live Script](#)
- Create and Configure AUTOSAR Adaptive Software Component**: Create an AUTOSAR adaptive software component model from an algorithm model. [Open Live Script](#)
- Author AUTOSAR Compositions and Components in...**: Develop AUTOSAR compositions and components for the Classic Platform by using an architecture model. [Open Live Script](#)

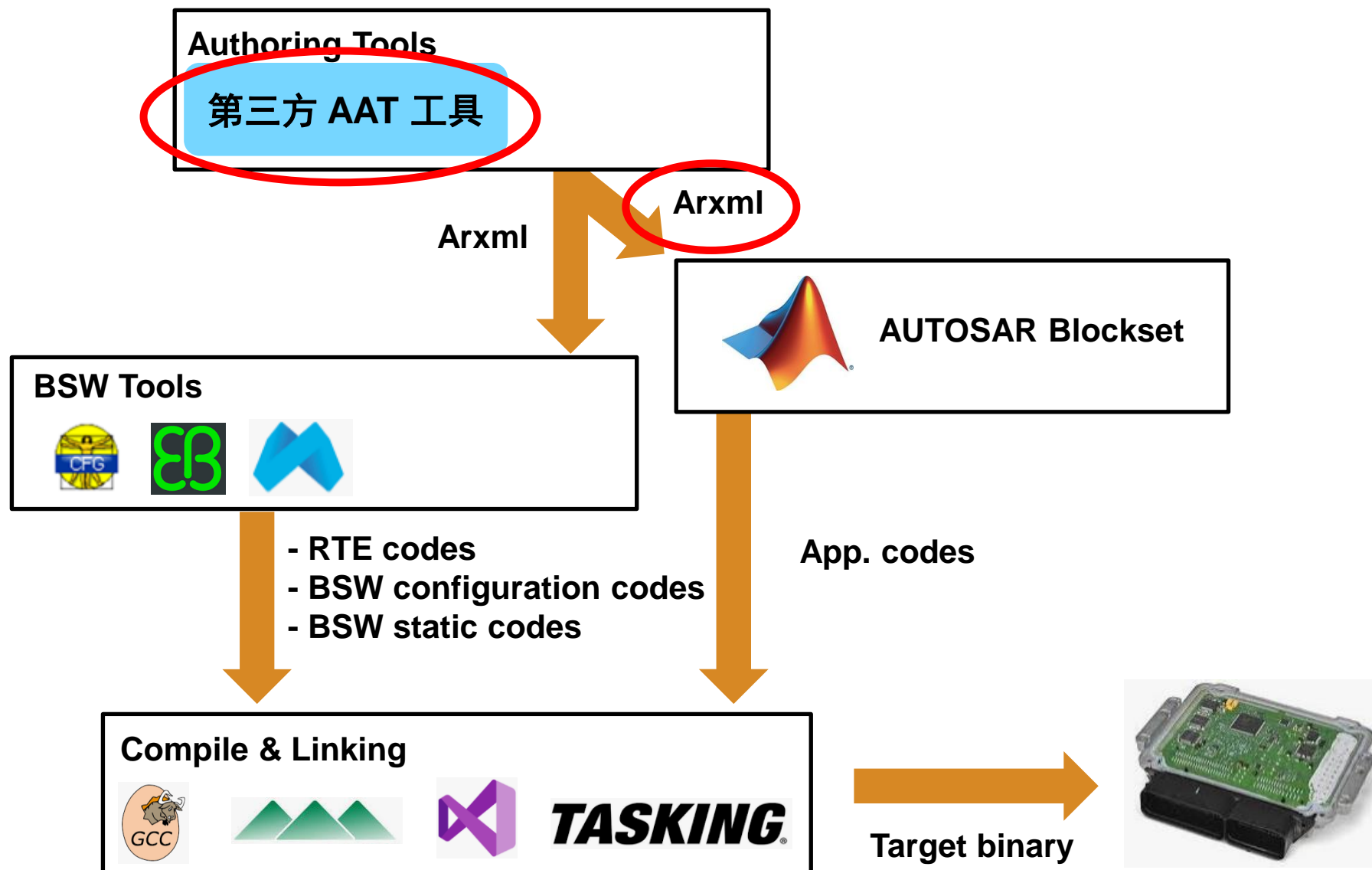
The left sidebar contains a "CONTENTS" menu with the following items:

- « Documentation Home
- « Examples
- Category
  - AUTOSAR Blockset**
    - Get Started with AUTOSAR Blockset 3
    - Software Component Modeling 9
    - Adaptive Software Component Modeling 4
    - Composition and ECU Software Simulation 4
    - Software Architecture Modeling 2
  - Bioinformatics Toolbox
  - Communications Toolbox
  - Computer Vision Toolbox
  - Control System Toolbox
  - Curve Fitting Toolbox
  - Data Acquisition Toolbox

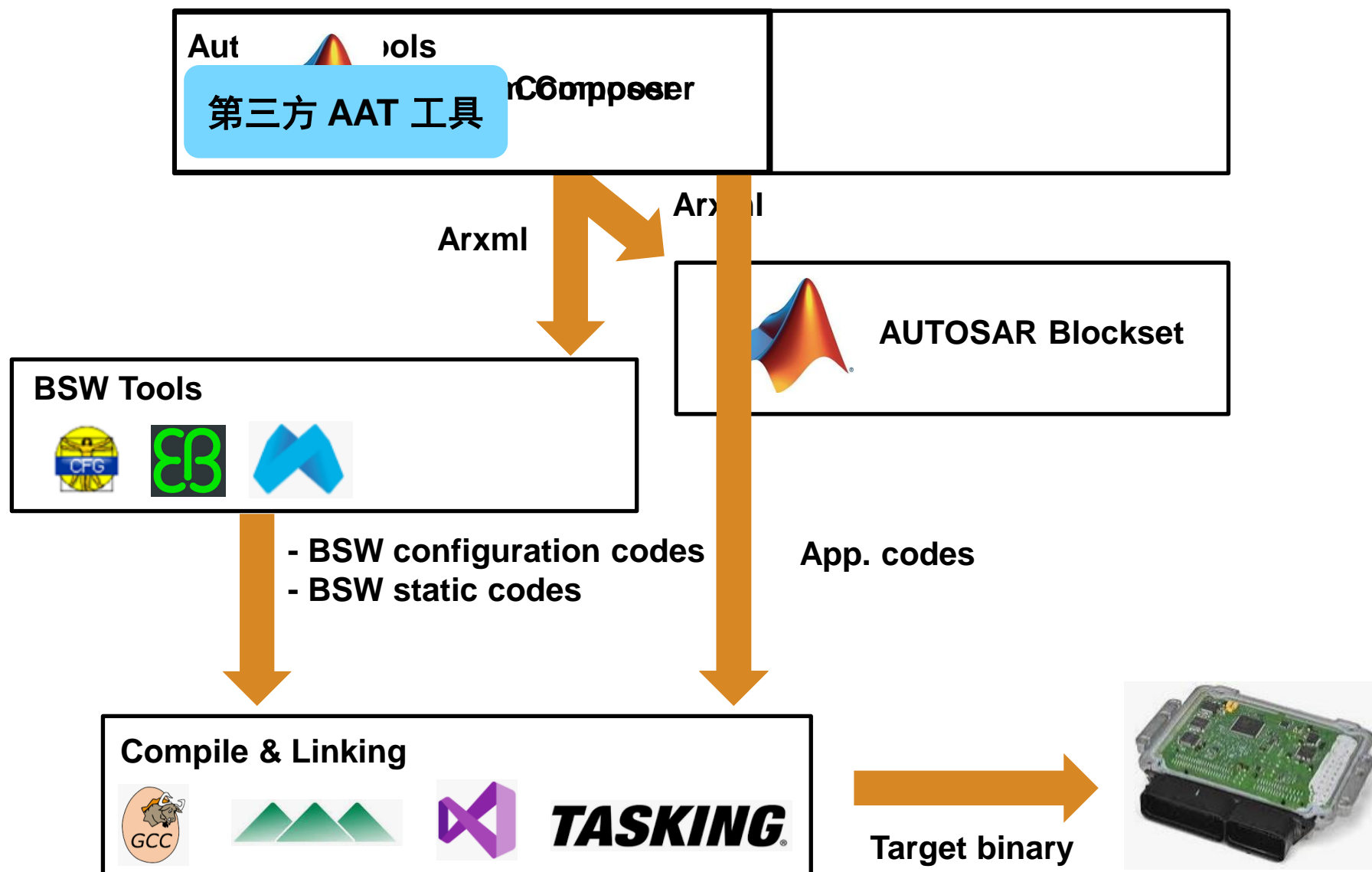
# 使用第三方 AAT 工具的 MBD Workflow (1)



## 使用第三方 AAT 工具的 MBD Workflow (2)



## 使用 System Composer 的 MBD Workflow (2)



# 基于 Excel 生成 AUTOSAR 架构导出 ARXML 文件

- 解析EXCEL数据，并基于MATLAB AUTOSAR架构设计工具System Composer自动创建AUTOSAR架构设计模型，然后导出AUTOSAR架构设计文件ARXML；

The image illustrates the process of generating ARXML files from an Excel spreadsheet using the AUTOSAR System Composer. On the left, an Excel spreadsheet contains data for various components and interfaces. The center shows the System Composer interface with a block diagram of an ECS component and its associated interfaces. On the right, a file explorer displays the resulting ARXML files.



基于MATLAB API

Name	Size	Item type	Date modified
ECS_Arch_composition.arxml	1.26 KB	ARXML File	2022/5/9 22:50
ECS_Arch_datatype.arxml	11.4 KB	ARXML File	2022/5/9 22:50
ECS_Arch_interface.arxml	32.2 KB	ARXML File	2022/5/9 22:50
ECS_Arch_timing.arxml	3.79 KB	ARXML File	2022/5/9 22:50
ECS_component.arxml	64.0 KB	ARXML File	2022/5/9 22:50
ECS_implementation.arxml	8.01 KB	ARXML File	2022/5/9 22:50
ECS_timing.arxml	12.3 KB	ARXML File	2022/5/9 22:50



# 主要内容

- AUTOSAR解决方案的进化
- System Composer实现软件架构设计
- SWC的Simulink实现
- 基于Excel的AUTOSAR架构设计

# MATLAB EXPO

## Thank you



© 2024 The MathWorks, Inc. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See [mathworks.com/trademarks](https://www.mathworks.com/trademarks) for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

