



长城汽车

Simulink加速在自适应和经典 AUTOSAR平台中的SOA软件开发

杜建福, 长城汽车



Jeff Du, GWM

2024 MathWorks
中国汽车年会

科技长城——智能化

Coffee Pilot 智慧行驶 Intelligent driving system

智驾系统迭代

3代

整体搭载车型已超过

20款

No. of car models equipped with this system exceeded 20

用户辅助驾驶里程上

亿公里

Duration of data-driven intelligent learning 1 million hours

Coffee OS 智慧空间 Intelligent cockpit system

车机快速启动时间

2.64秒

列表滑动帧率

60FPS

应用启动时间

436ms

语音唤醒时间

293ms

地图滑动响应时间

153ms

咖啡智能，作为长城汽车整车智能化技术品牌，秉持“用户主义”理念，着力为用户打造更安全、更实用、更具性价比的智能化出行体验，为整车提供全域智能化解决方案。

长城汽车第三代智能驾驶系统Coffee Pilot Ultra，首搭全新蓝山智驾版，年底将实现全国“有路就能开”，“有位就能停”。SOA作为跨域跨系统的新技术，必将推动智能汽车的快速落地。

目录

EE架构&软件架构

SOA概览

基于CP的SOA开发

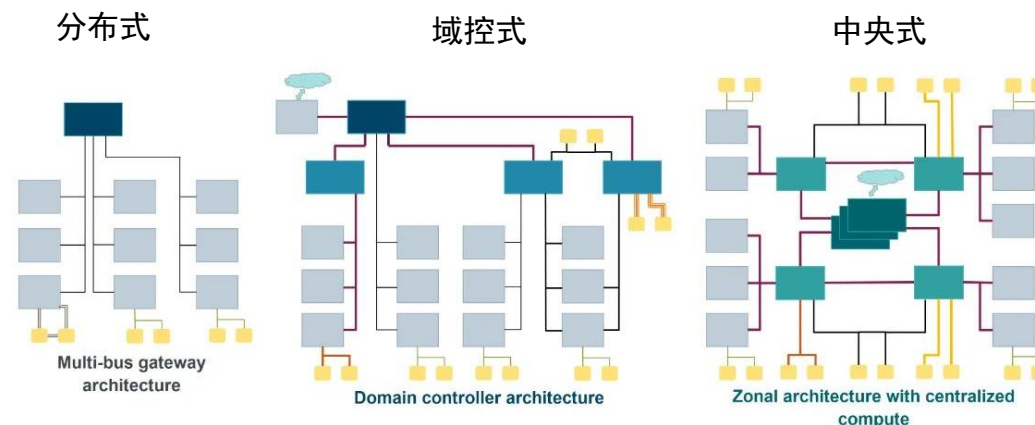
基于AP的SOA开发

总结

EE架构之变迁

分布式ECU架构

- 单个ECU控制单一或者几个功能
- ECU数量多（几十到100多）
- 算力浪费
- 软硬件耦合深
- 线束长，成本高，组装自动化水平低
- 主机厂严重依赖供应商



Source from web

域控式架构

- 多ECU整合成动力域、底盘域、车身域、座舱域、自动驾驶域五个域控制器或者车控域，座舱域，自动驾驶域3个
- ECU数量减少
- 功能软件集中到多个域控制器

中央计算架构

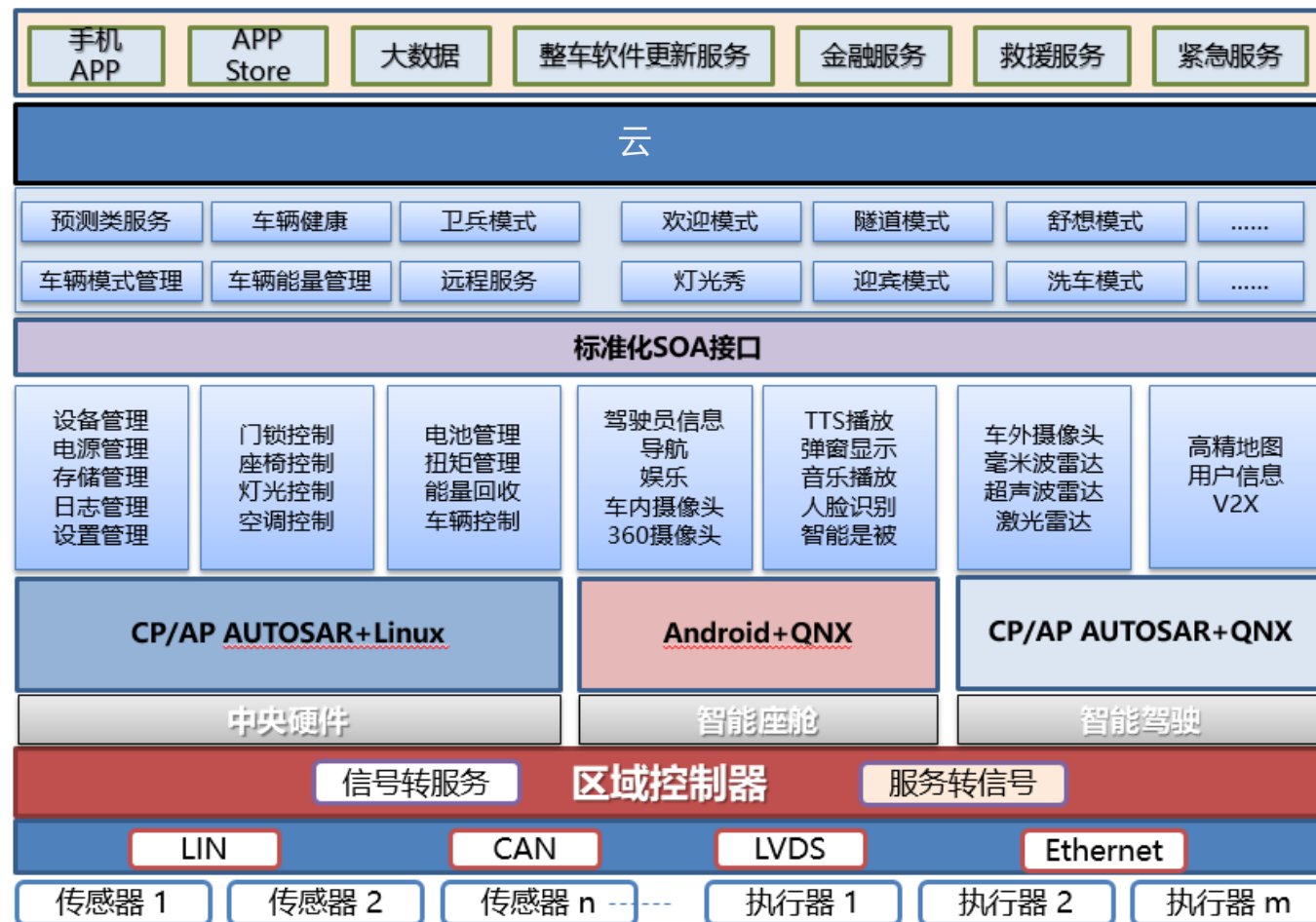
- 域控制器整合成车载电脑
- 核心功能集中到车载电脑
- 辅助功能集中到区域控制器

软件架构概念延展

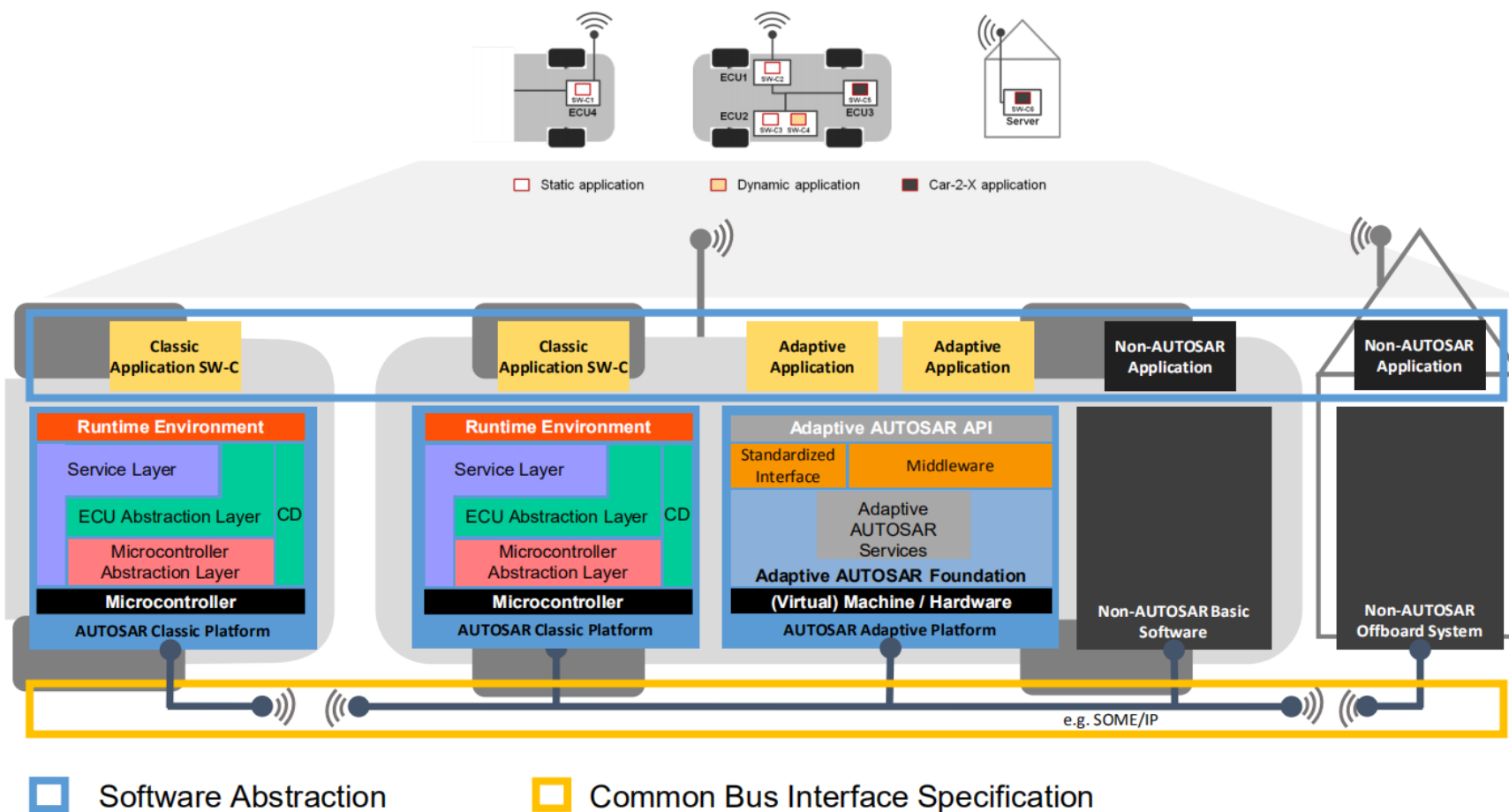
传统软件架构：
单个ECU



SOA软件架构：
车云一体



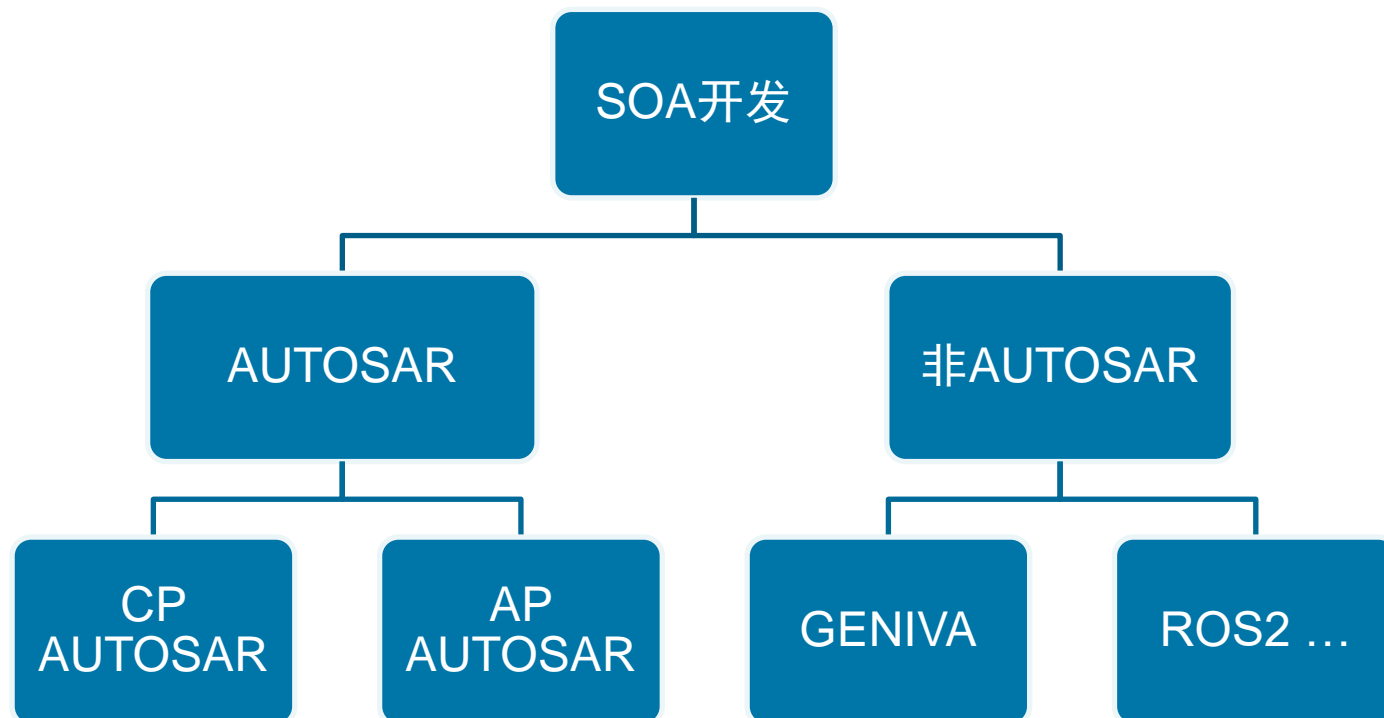
未来汽车软件架构



Source from 2016 AUTOSAR conference: *Ethernet and the AUTOSAR Adaptive Platform as basis for future E/E Architecture*

未来车辆是AP AUTOSAR, CP AUTOSAR 和非AUTOSAR 架构并存。

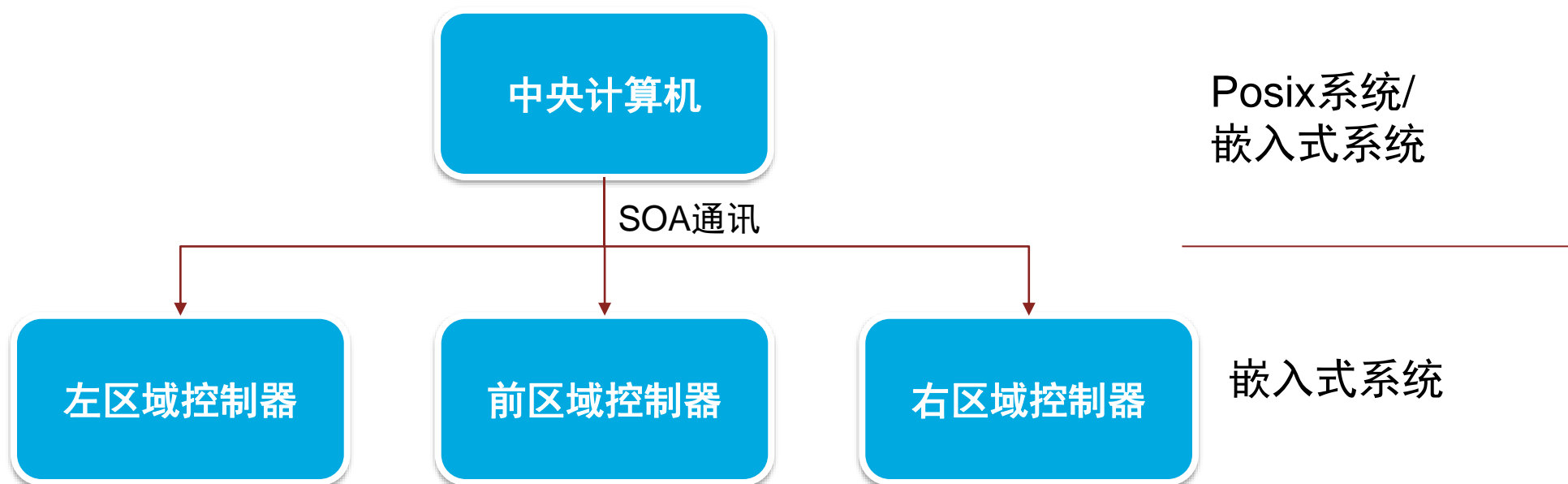
SOA概览-开发标准



SOA开发目前有基于AUTOSAR的方式，也有基于非AUTOSAR的方式。AUTOSAR是汽车行业的开放组织。其目标是定义汽车软件的标准，各家各自去实现。

SOA开发的挑战

SOA需要在支持Posix系统的A核上开发，也需要在M核（R核）或者在传统ECU上进行开发。



SOA开发的挑战 – CP & AP

| 项目 | Classic | Adaptive |
|--------|---|--|
| 使用语言 | C | C++ |
| 实时性 | 硬实时(us) | 软实时(m) |
| 适用场景 | 实时性高，功能安全要求高，算力要求低 传统ECU，如ECM、VCU、BMS、MCU等 | 算力要求高，实时性有一定要求，功能安全有一定要求的自动驾驶、辅助驾驶、车联网 |
| 功能升级 | 一般ECU开发后比较固定，支持整体FOTA升级 | 可灵活在线升级 SOTA，单个APP升级 |
| 安全等级 | 最高到ASILD | ASIL B |
| 主要通信方式 | CAN、LIN、以太网 | 以太网 |
| 操作系统 | EB tresos®, MICROSAR, RTA, KPIT | POSIX(Linux, QNX, Pike OS) |
| 开发工具链 | 成熟 | 开发中 |

软件架构

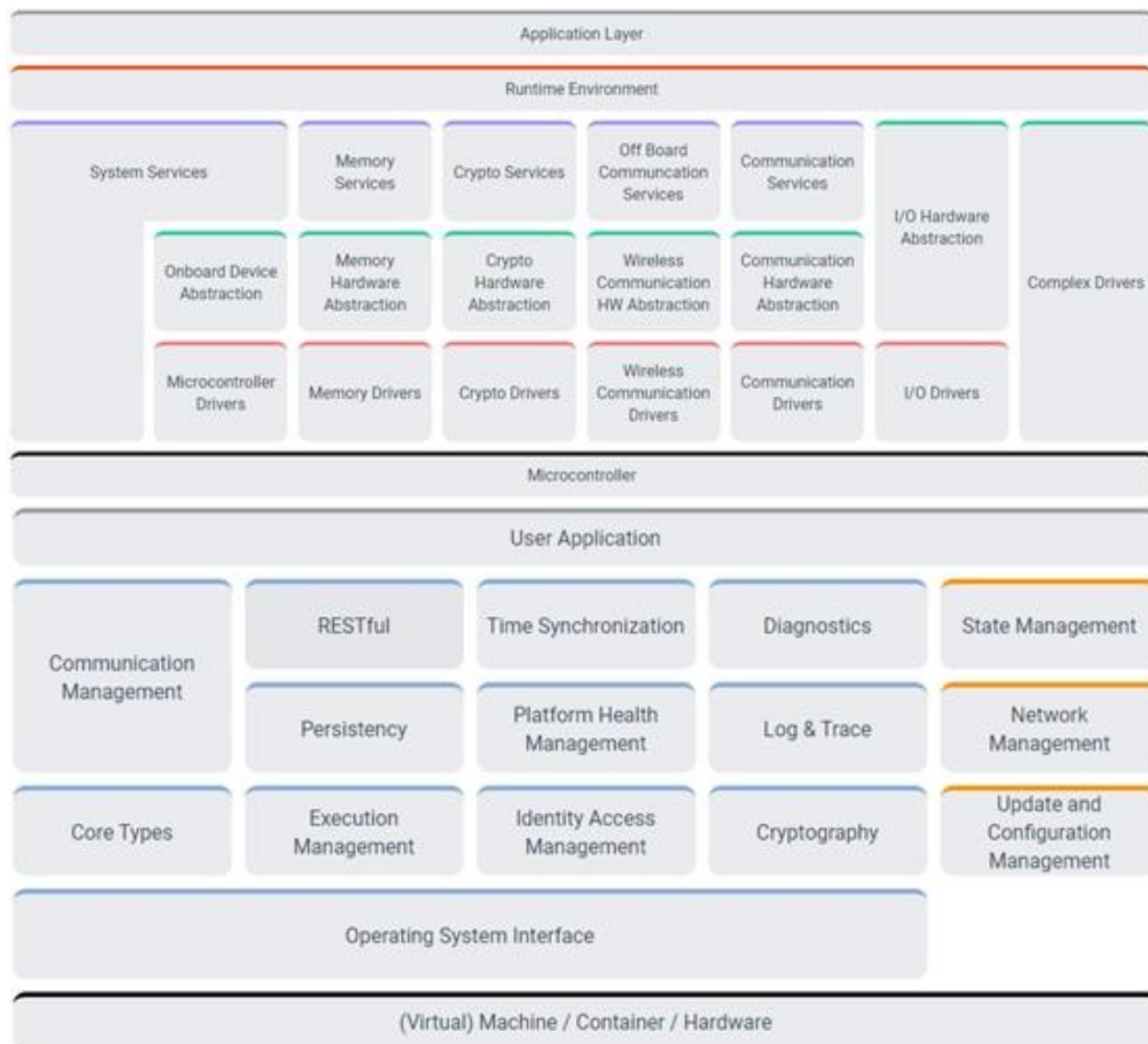
CP AUTOSAR是分层的软件架构，有较为明显得上下层关系

从下到上依次为：

- 1、微控制器层 (HW)
- 2、基础软件层 (BSW)
 - 微控制器抽象层
 - ECU抽象层
 - 服务层/复杂驱动
- 3、RTE层
- 4、Application层

AP AUTOSAR主要由两部分组成 (Foundation和Service)：

右图中，所有的模块都称为功能集群 (Functional Clusters, FC)。蓝色的FC属于Foundation的部分，橘色的部分属于Service的部分。无论是Foundation部分的FC，还是Service部分的FC，都不是上下层关系。



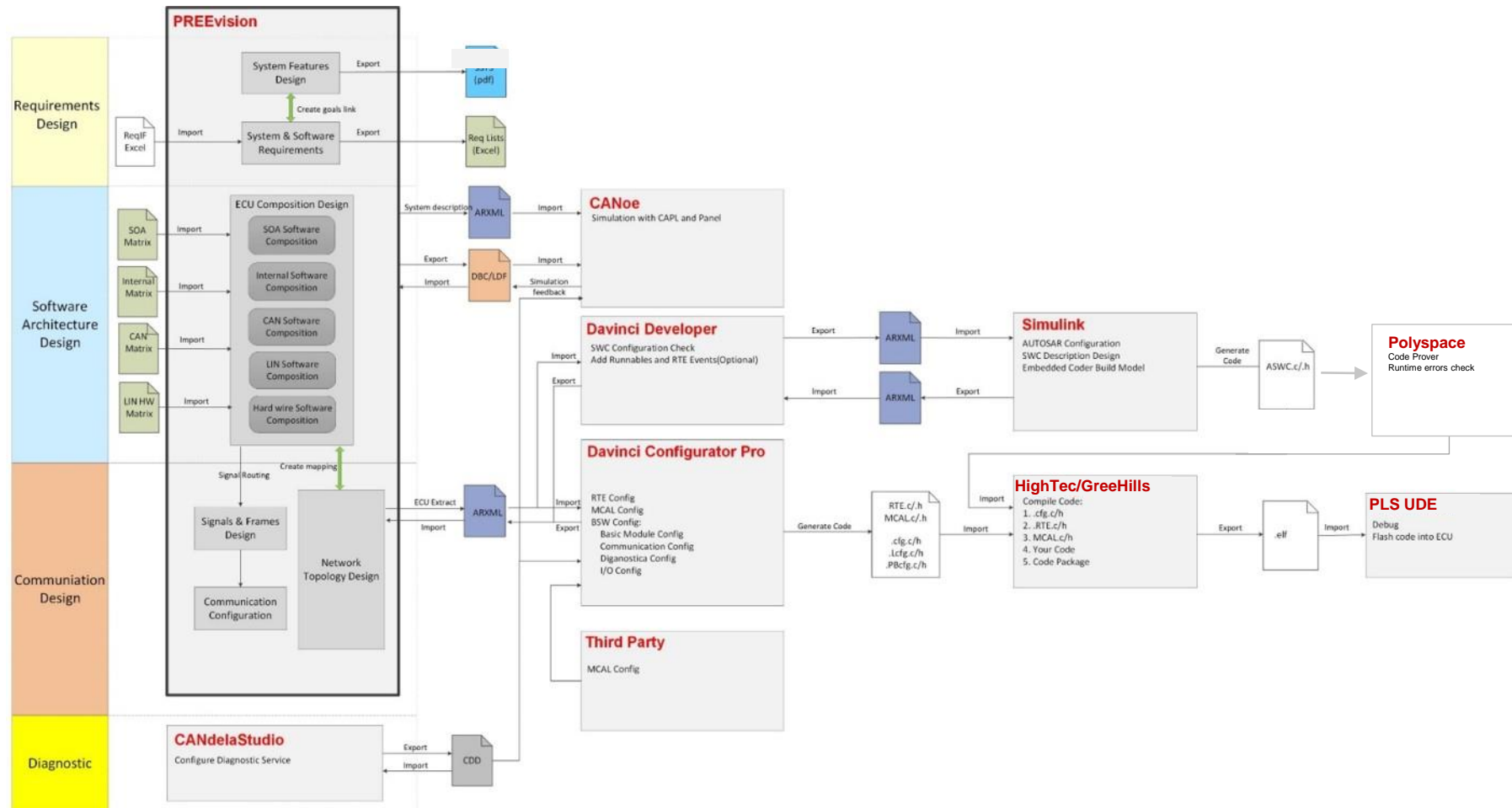
SOA开发的挑战



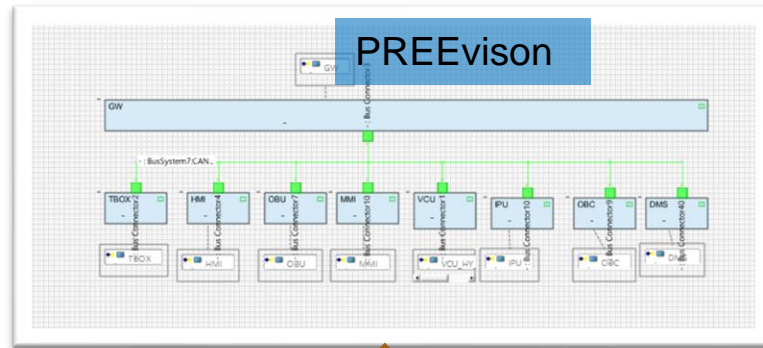
基于CP的SOA开发

- 基于MCU开发车载控制器软件，一般基于Classic Platform。CP平台也支持SOA开发，比较成熟的是基于SOMEIP的SOA开发。基于CP SOMEIP协议的SOA开发相对AP而言比较成熟。
- Vector在2018年就已经支持SOMEIP协议栈，软件架构开发有PREEvison和Davinci Developer 支持，代码配置和代码生成有Davinci Configurator，应用层开发和测试有Simulink 支持，工具链相对完善。

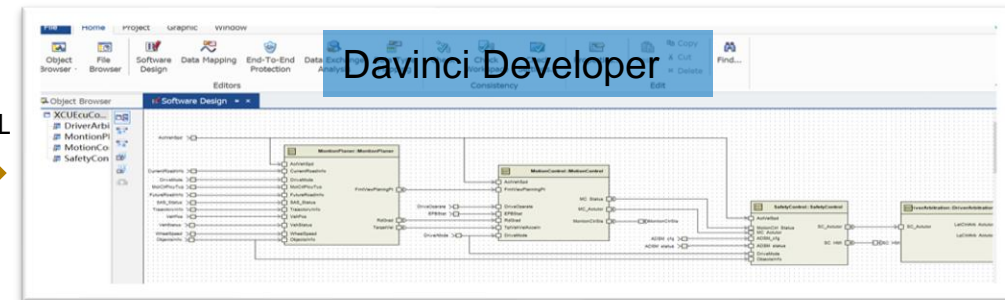
CP软件开发工具链



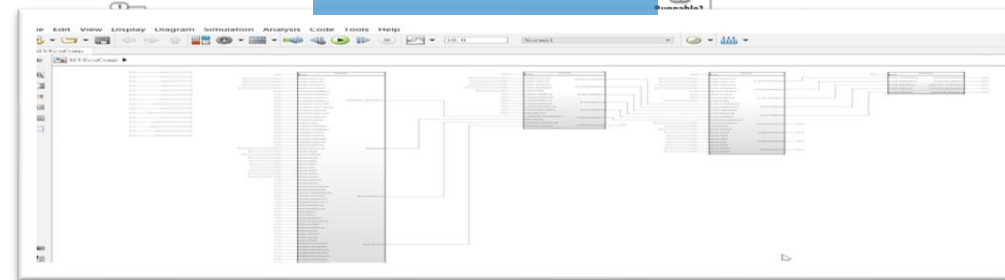
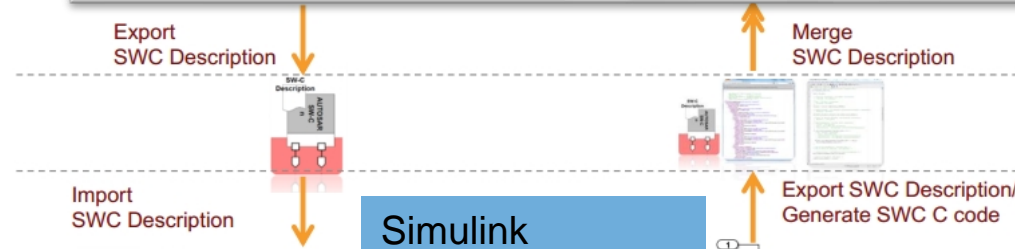
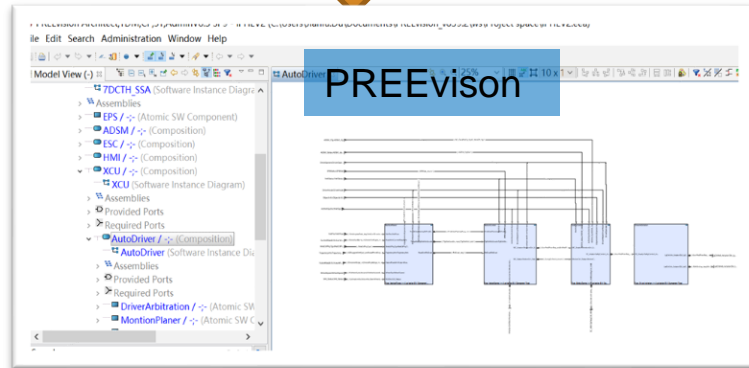
CP软件开发工具链 – 架构和建模



抽取ARXML



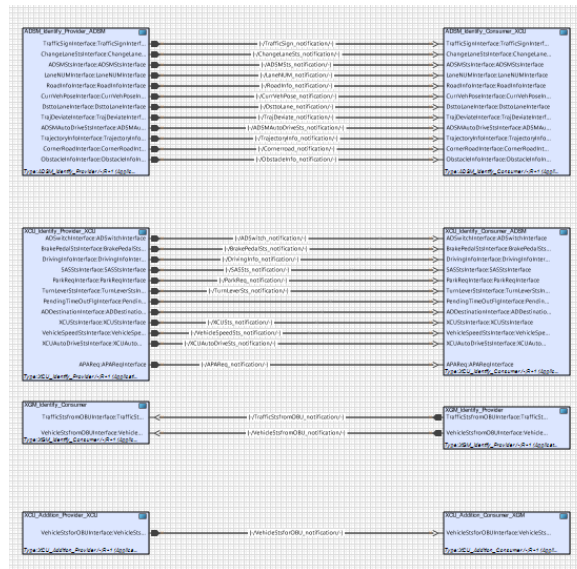
mapping



CP开发实例

SOA通信矩阵导入PREEvision生成SOA模型

| Ethernet Message | ID (HEX) | Typ | Payload (Byte) | SOME/IP Message Type (ID) | SOME/IP Message Type (name) | Transport Protocol | Data Type | notification strategy | Cyclic Time (ms) | Sender Ethernet | Receiver Ethernet | SOME/IP Service | DataElement/Argument | Signal Length(Byte) | Type | Signal description |
|------------------|----------|-----|----------------|---------------------------|-----------------------------|--------------------|-----------|-----------------------|------------------|-----------------|-------------------|-----------------|----------------------|---------------------|------|--------------------|
| x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x |
| x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x |



- SWArchitecture / -;R+1 (System Software Architecture)
 - SW Package / -;R+1 (SW Package)
 - XCUecuComp / -;R+1 (Root Composition)
 - XCUecuComp (Software Instance Diagram)
 - Assemblies
 - Provided Ports
 - Required Ports
 - ADSM_Identify_Consumer_XCU / -;R+1 (Atomic SW Component)
 - XCU_Addition_Provider_XCU / -;R+1 (Atomic SW Component)
 - XCU_Identify_Provider_XCU / -;R+1 (Atomic SW Component)
 - XGM_Identify_Consumer / -;R+1 (Atomic SW Component)
- Communication / -;R+1 (Communication)
 - Communication Package / -;R+1 (Communication Package)
 - TP / -;R+1 (Transport Layer Package)
 - So Ad Routing Groups5 (So Ad Routing Groups)
 - Conditions
 - ETH EthernetCommunicationCluster2 / -;R+1 (Ethernet Comm)
 - GeneralPurposePDUs / -;R+1 (Layout Package)
 - Signals / -;R+1 (Layout Package)
 - Transport Layer Package10 / -;R+1 (Transport Layer Packag)
 - Transformer Configuration Package5 (Transformer Configur)

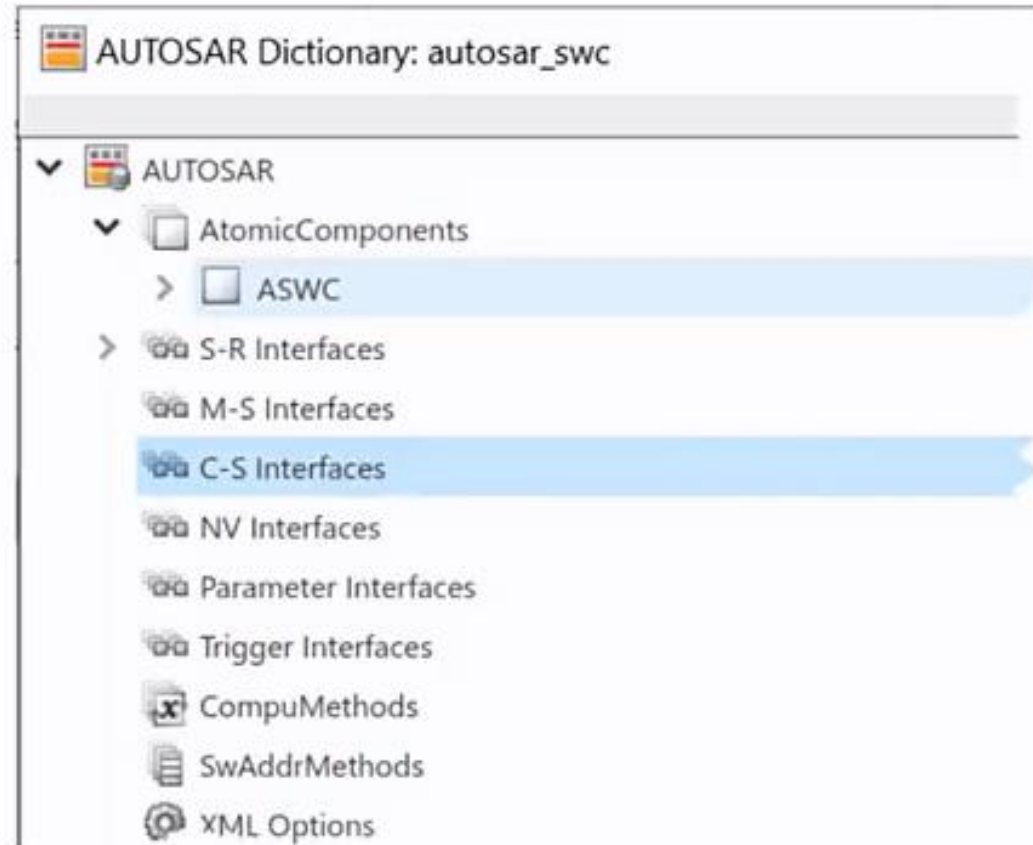
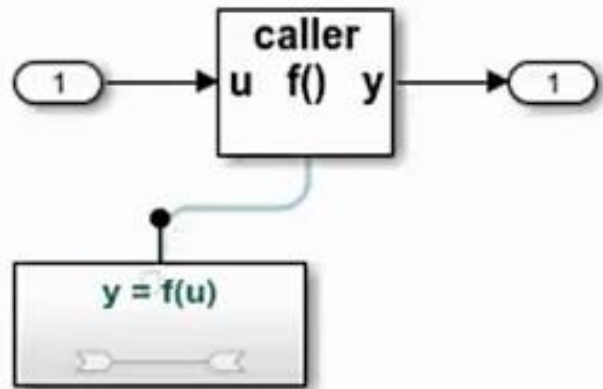
CP RTE接口类型与Simulink实现

RTE 接口

- S-R : 输入与输出

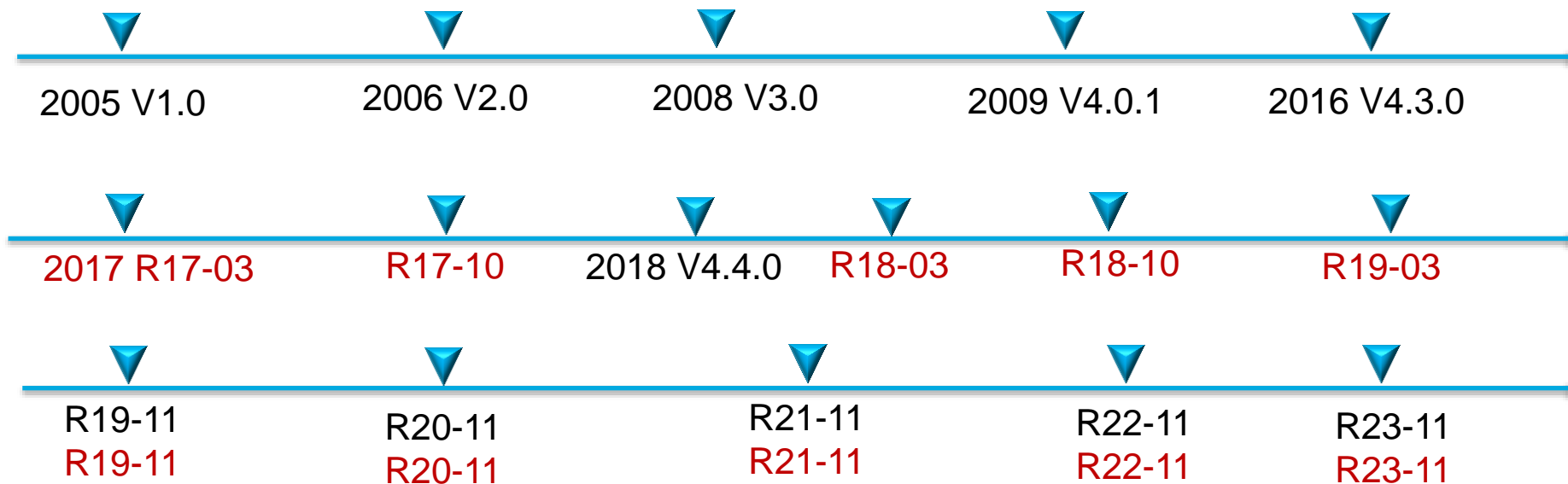


- C-S: Simulink Function 与 Function Caller



CP 平台下的SOA开发，无论是软件架构还是模型开发，工具链都相对完善。

基于AP的SOA开发



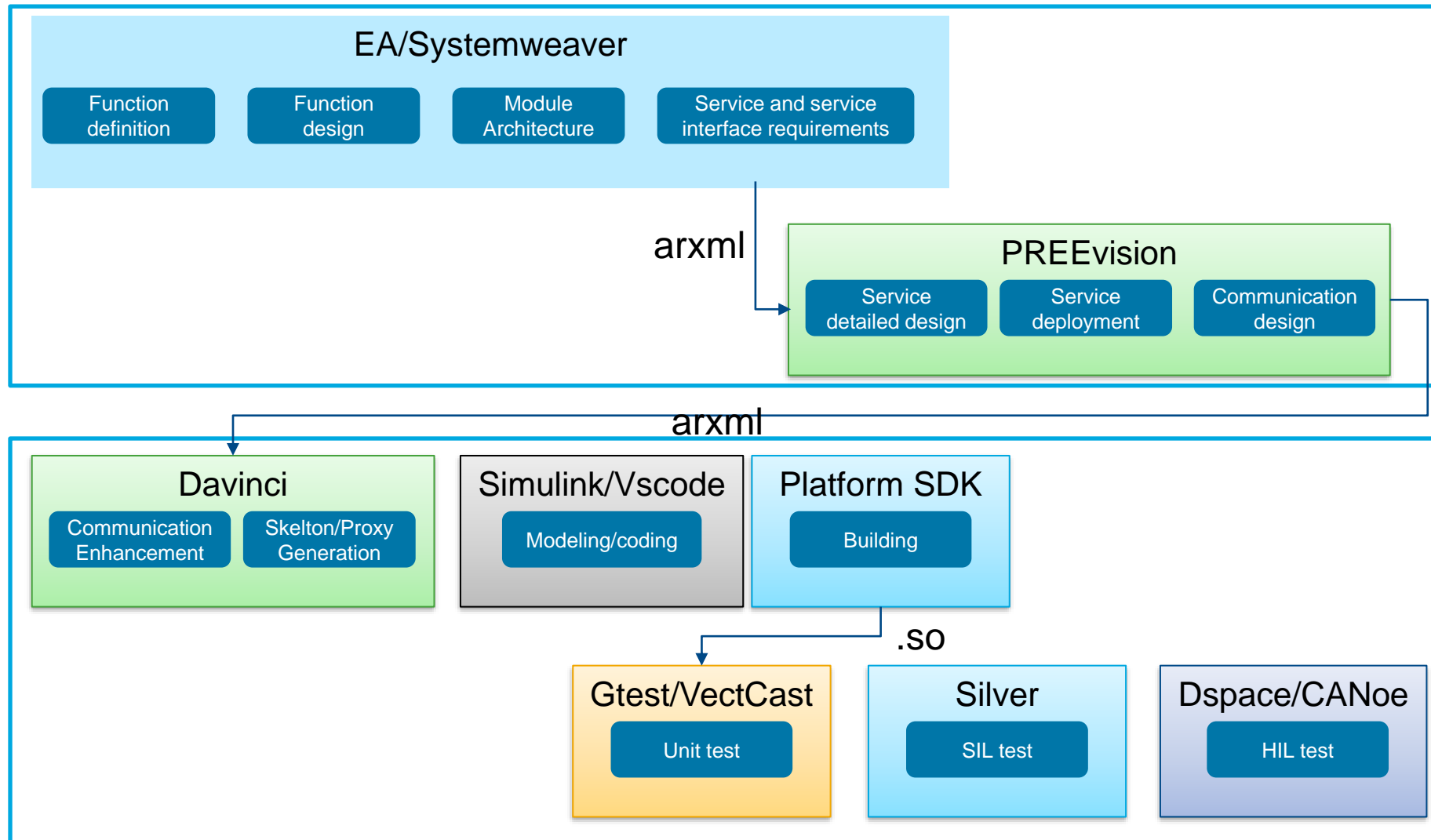
黑色字体：CP 版本发布日期 红色字体：AP版本发布日期

2005年，AUTOSAR 组织推出第一个版本 V1.0，也就是CP的第一个版本
2017年 AUTOSAR组织推出了 AP AUTOSAR版本R1703，这是AP AUTOSAR的第一个版本。

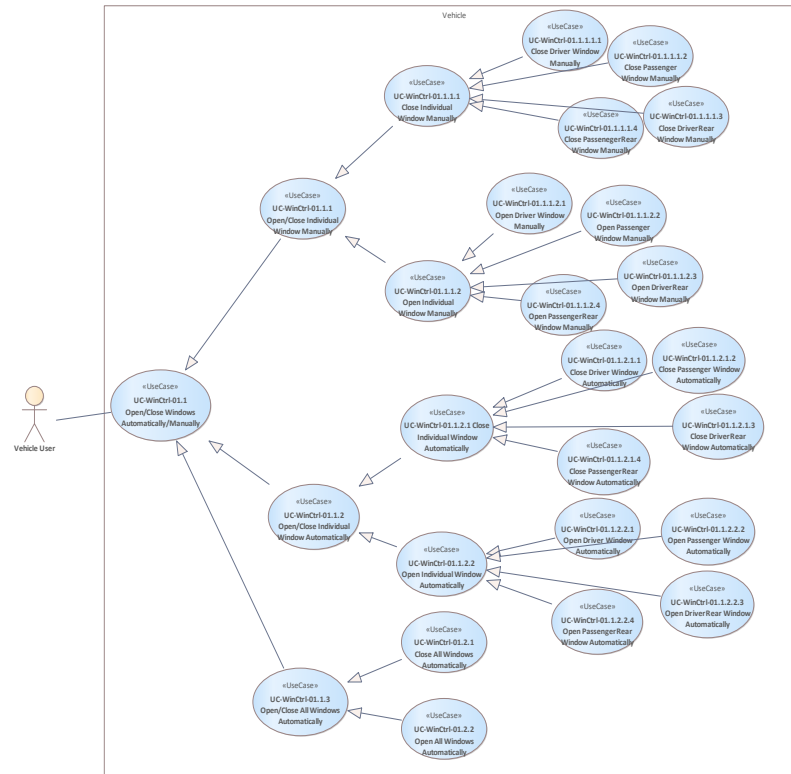
CP到现在已经有17年了，而AP才5年。

AP规范仍在开发之中。AP工具链还不成熟，各家都在摸索之中

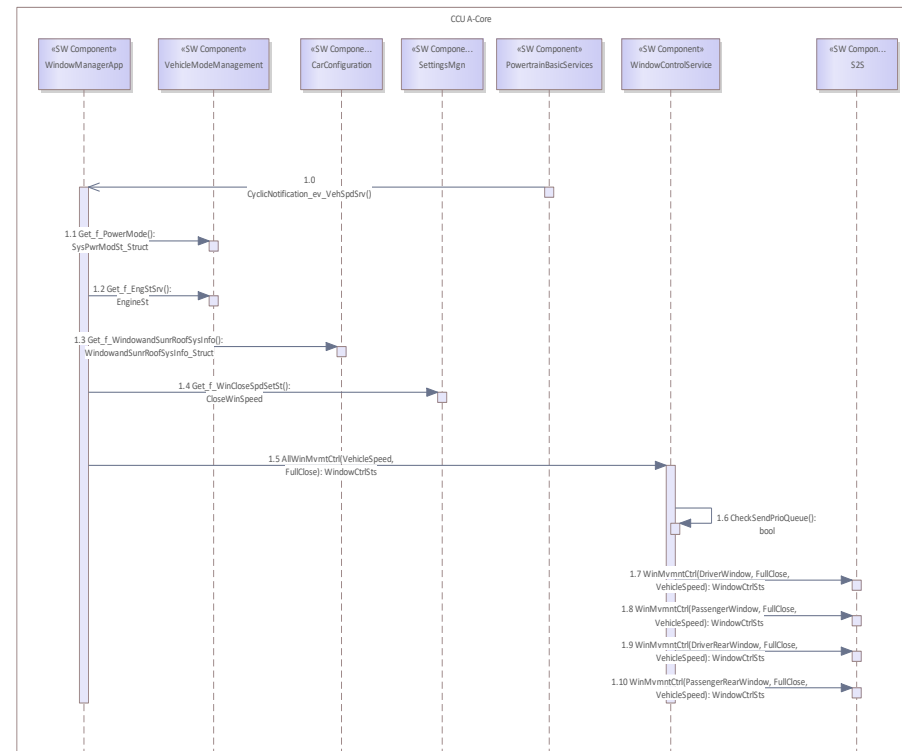
AP软件 SOA 开发工具链



基于EA进行功能定义和动态设计



功能定义



动态设计

基于PREEvison进行服务设计

| Service Interface | Used by Service | Service Interface Sub-element | Dir | Data Type |
|----------------------|--------------------|-------------------------------|-----|------------------|
| ↔ trafficInformation | TrafficInformation | onChangeTrafficStatus | | 0100 uint8 |
| ↔ navigator | Navigator | onChangeVehicleStatus | | 0100 uint8 |
| ↔ navigator | Navigator | calculateRoute | | n/a |
| | | # from | In | IMPL idtPosition |
| | | # to | In | IMPL idtPosition |
| | | # route | Out | IDT idtRoute |
| ↔ trafficInformation | TrafficInformation | getTrafficStatus | | n/a |
| | | # position | In | IMPL idtPosition |
| | | # direction | In | 0100 uint8 |
| | | # trafficStatus | Out | 0100 uint8 |
| ↔ navigator | Navigator | # currentPosition | | IMPL idtPosition |
| ↔ navigator | Navigator | # vehicleStatus | | 0100 uint8 |

服务定义

服务接口设计

服务接口部署

软件架构设计

AA (SWC) 设计

网络设计

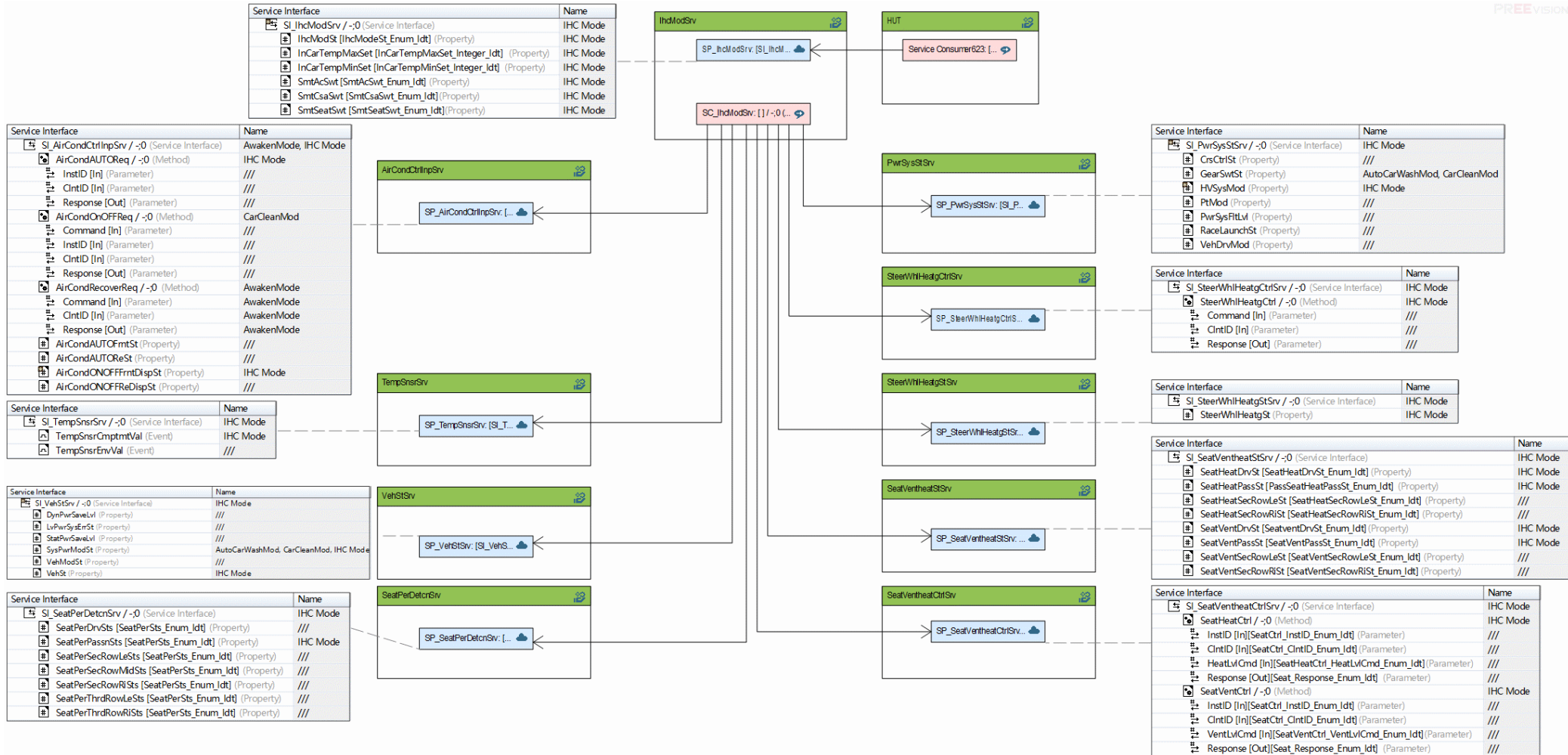
软件硬件映射

机器部署

应用部署

服务实例化

服务架构图



AP 开发实践 – Davinci Developer Adaptive

The screenshot displays the Davinci Developer Adaptive IDE interface. The main window shows the AUTOSAR Model Explorer with the following structure:

- Workspace - AUTOSAR Model Explorer - Davinci Developer Adaptive
 - LightControlSystemIntegra
 - LightControlSystemIntegrationProject
 - ServiceInterfaces
 - ServiceInterface LightSensorInterface
 - fields, namespaces, Incoming References
 - ServiceInterface LightSensorInterface {
 - namespace : lightcontrol::lightsensorservice
 - fields [
 - Field /OneCompany/AppExample/LightControl/DataTypes/LightSensorState stateOfLightSensor {
 - hasNotifier : true
 - hasGetter : true
 - hasSetter : false

The right-hand pane shows the code for the `ServiceInterface LightSensorInterface`. A tooltip is visible over the `ServiceInterface` class, listing various service interface proposals:

- ServiceInterface
- ServiceInterfaceMappingSet
- ServiceInterfacePedigree
- DdsServiceInterfaceDeployment
- IpcServiceInterfaceDeployment
- RestServiceInterface
- SomeipServiceInterfaceDeployment
- DiagnosticServiceValidationInterface
- TransformationPropsToServiceInterfaceElementMappingSet
- ProvidedServiceInstanceToSwClusterDesignPPortPrototypeMapping
- RequiredServiceInstanceToSwClusterDesignRPortPrototypeMapping
- DiagnosticServiceDataIdentifierPortMapping

The tooltip also includes a description: "Ar4x.ServiceInterface. This represents the ability to define a PortInterface that consists of a heterogeneous collection of methods, events and fields." and a note: "Showing the complete list of proposals. Press 'F2' for focus."

SOA软件的开发方式

纯手工开发

- 手工/自动编写APP框架，手工编写算法逻辑

纯Simulink开发

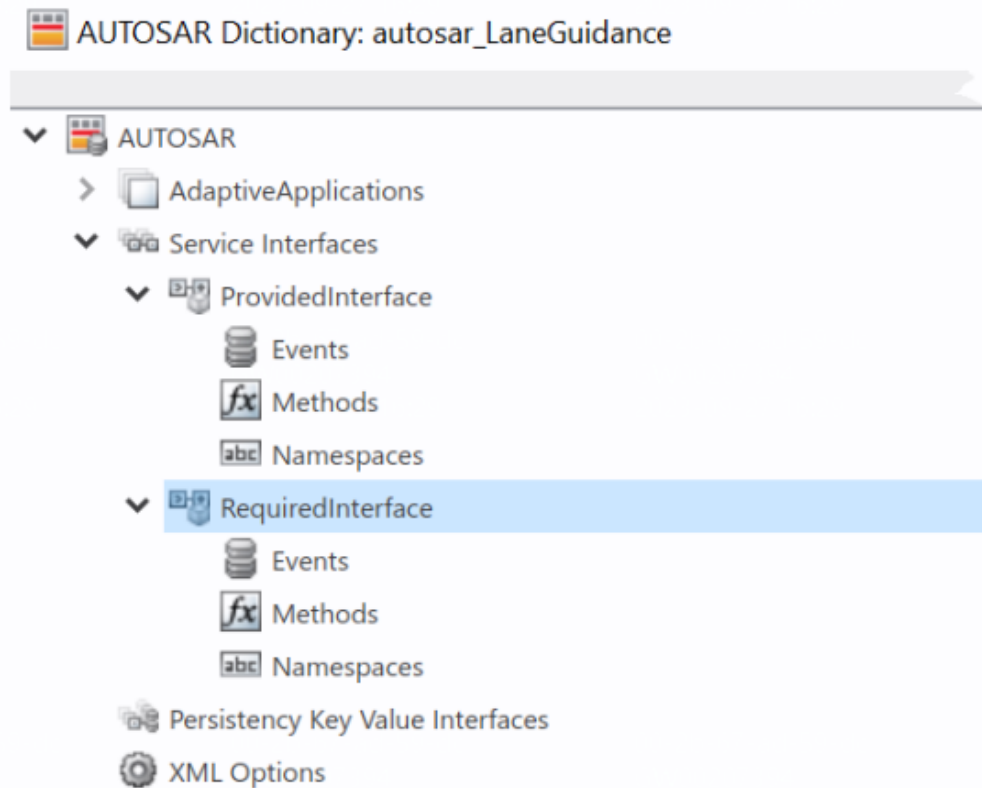
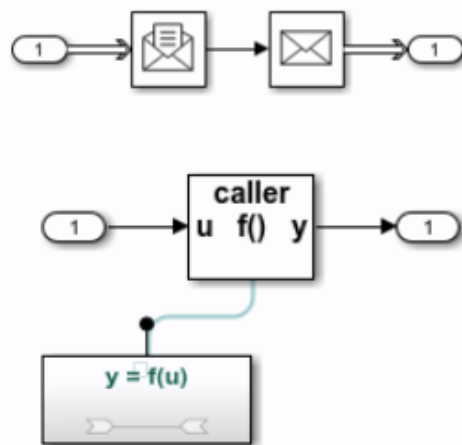
- 利用Simulink生成的框架代码和算法逻辑
- 采用adpative_autosar.tlc模板
- 直接利用Simulink生成的main函数和模型step函数

手工和Simulink结合


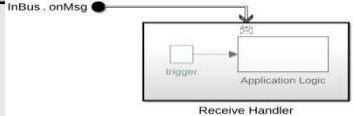
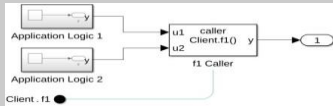
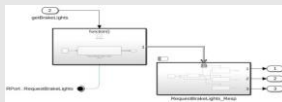



- 手工/自动编写APP框架代码
- 利用Simulink产生算法逻辑
- 采用ert.tlc模板
- 集成Simulink生成的算法代码（Skeleton和Proxy代码），调用Step函数

Adaptive AUTOSAR 接口 Simulink 实现方式

- Event: 输入与输出
- Method: Simulink Function 与 Function Caller
- Field: Event/Method



Adaptive AUTOSAR 通讯支持

| AP通信语义 | 类型 | Simulink建模语义 | 版本 |
|----------------------------|----|--|---------------|
| Events | 轮询 |  | R2019a |
| | 触发 |  | R2022a |
| Methods | 同步 |  | R2022a |
| | 异步 |  | R2022b |
| Fields | |  | R2023b |
| Persistency (ara::per) | |  | R2021b |
| Log & Trace | | 临时方案  | |
| Platform Health Management | | 未知 | |

AP 平台下的SOA开发，软件架构开发相对完善，基于模型开发工具还需要继续完善。

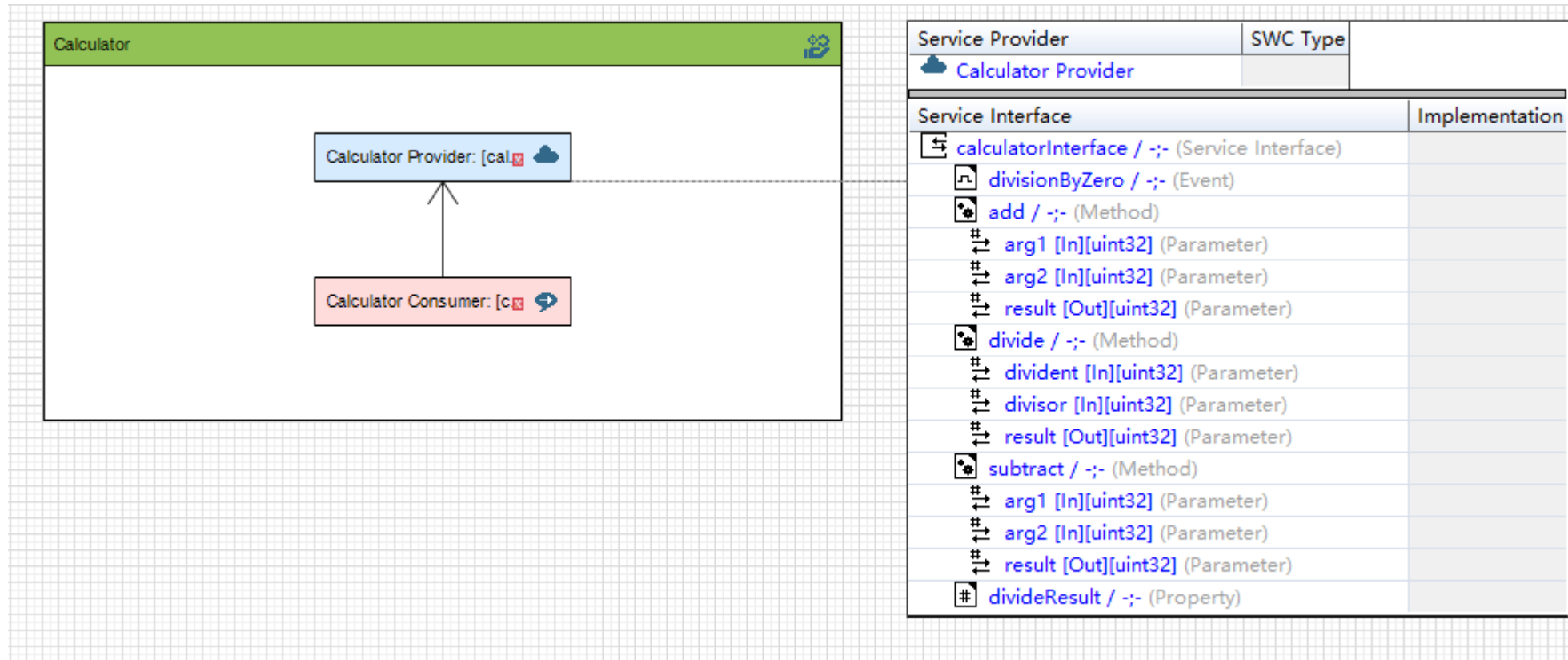
实例演示-功能说明

设计一个32位无符号除法计算器。当除数为0时，需要报除法异常错误。
除法运算结果可以标定，标定值存储在一个文件中。程序启动时读取标定值，程序关闭时，保存标定值。
程序由一个除法计算器的基础服务和一个客户端构成。

实例演示 – 服务接口设计

| A | B | C | D | E | F | G | H | I | J | K | L |
|--------------------------|----------------------|--------------------|-------------------------|--|----------------|--------------|------------------|-----------|-------|--|-----------|
| PackageName | ServiceInterfaceName | ServiceInterfaceID | ServiceInterfaceVersion | ServiceInterfaceDescription | EventgroupName | EventgroupID | TransmissionType | FieldType | RR/FF | ElementName | ElementID |
| vector.calculatorService | calculatorService | 1666 | 1 | This service provides the divider function | - | - | Method | - | RR | divide | 1603 |
| vector.calculatorService | calculatorService | 1666 | 1 | This service provides the divider function | - | - | Method | - | RR | divide | 1603 |
| vector.calculatorService | calculatorService | 1666 | 1 | This service provides the divider function | - | - | Method | - | RR | divide | 1603 |
| vector.calculatorService | calculatorService | 1666 | 1 | This service provides the divider function | divideByZeroEg | 888 | Event | - | FF | DivisionByZero | 45767 |
| vector.calculatorService | calculatorService | 1666 | 1 | This service provides the divider function | - | - | Field | Getter | RR | DivideResultCaEn(getDivideResultCaEn) | 1701 |
| vector.calculatorService | calculatorService | 1666 | 1 | This service provides the divider function | - | - | Field | Setter | RR | DivideResultCaEn(setDivideResultCaEn) | 1702 |
| vector.calculatorService | calculatorService | 1666 | 1 | This service provides the divider function | divideCaEg | 9087 | Field | Notifier | FF | DivideResultCaEn(notifyDivideResultCaEn) | 45678 |
| vector.calculatorService | calculatorService | 1666 | 1 | This service provides the divider function | - | - | Field | Getter | RR | DivideResultCaVal(getDivideResultCaVal) | 1703 |
| vector.calculatorService | calculatorService | 1666 | 1 | This service provides the divider function | - | - | Field | Setter | RR | DivideResultCaVal(setDivideResultCaVal) | 1704 |
| vector.calculatorService | calculatorService | 1666 | 1 | This service provides the divider function | divideCaEg | 9087 | Field | Notifier | FF | DivideResultCaVal(notifyDivideResultCaVal) | 45679 |

实例演示 – 服务架构设计

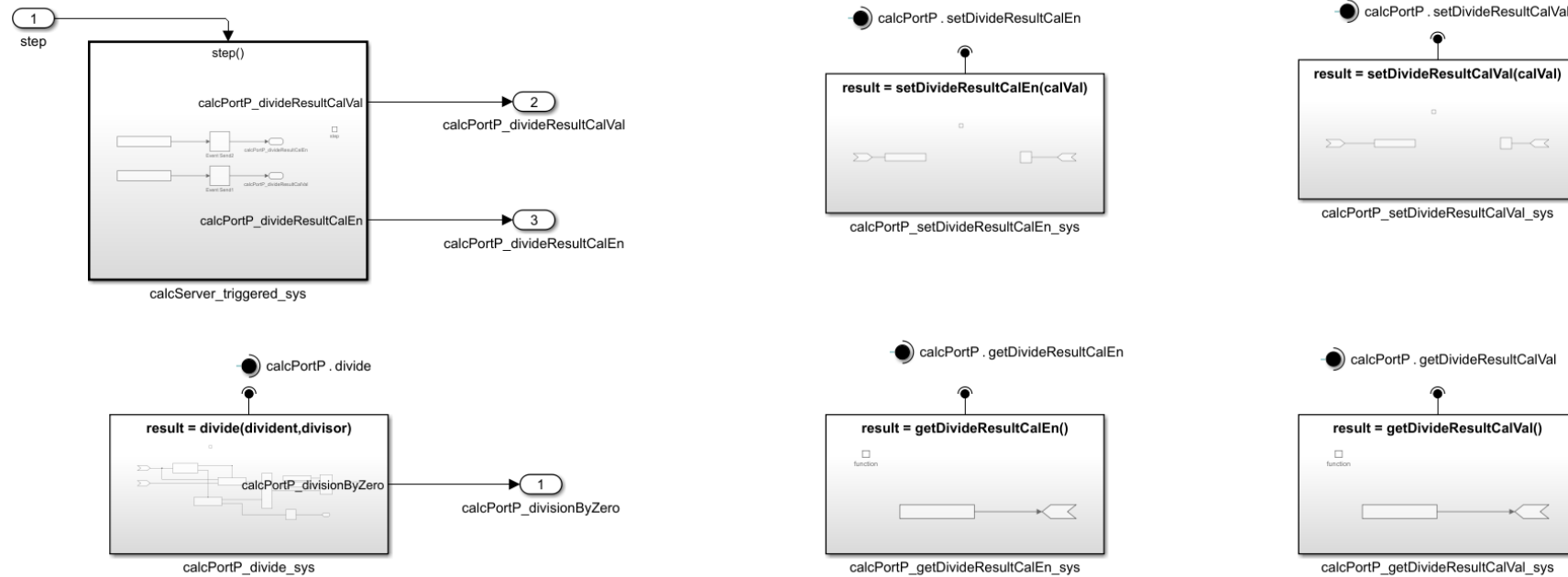


实例演示 – 创建Simulink 模型

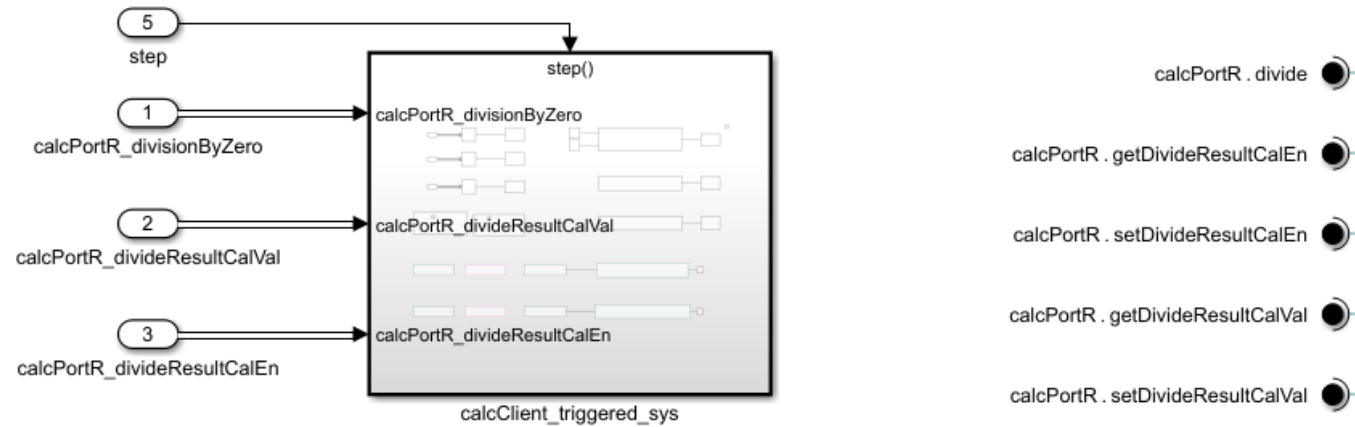
```
% Step1. 导入模型文件
ar = arxml.importer({'GWM_calculator_SI.xml','standard_datatypes.xml','GWM_calculator_server.xml', ...
'GWM_calculator_Server_machine.xml','GWM_calculator_PSI.xml','GWM_calculator_Client.xml', ...
'GWM_calculator_Client_machine.xml','GWM_calculator_CSI.xml','GWM_calculator_Per.xml'})

% Step2: 创建服务端和客户端
% 创建服务端
createComponentAsModel(ar,'/vector/calculatorService/calcServer')
% 创建客户端
createComponentAsModel(ar,'/vector/calculatorClient/calcClient')
end
```

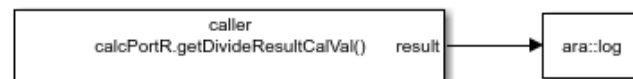
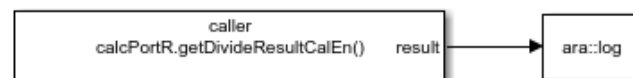
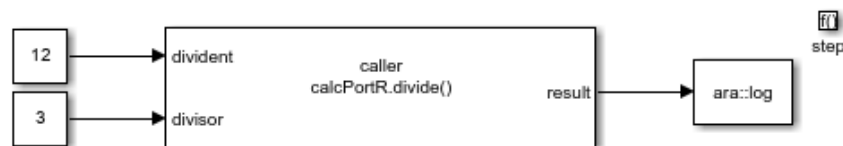
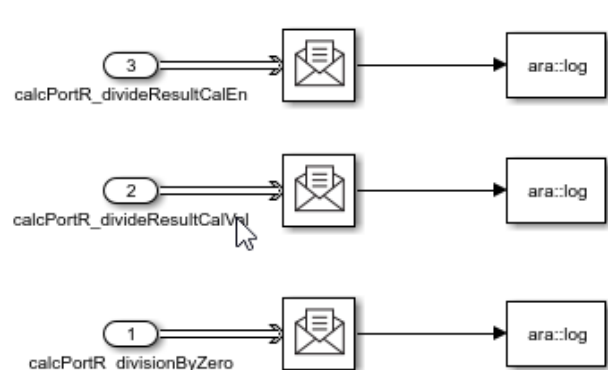
实例演示 – 算法开发（服务端）



实例演示 – 算法开发（客户端）



实例演示 – 算法开发（客户端）



实例演示 – 模型配置 – 服务端

更新Instance Identifier

The screenshot displays the MATLAB/Simulink AUTOSAR environment. The main workspace shows a Simulink model with several components, including a block labeled 'calcServer' and its associated ports and methods. A yellow warning message at the top of the workspace states: "This instance of the referenced model does not support all data visualizations. Scopes and port value labels display data for only the instance." The right-hand pane shows the 'AUTOSAR Dictionary: calcServer' with a tree view of the component's structure, including 'AdaptiveApplications', 'Service Interfaces', and 'Methods'. The 'Manifest attributes' section is expanded, showing the 'Instance Identifier' field set to '1403', which is highlighted with a red box. The bottom pane shows the 'Code Mappings - Component Interface' table, which lists the source code, ports, and methods for the component.

| Source | Port | Method |
|------------------------------------|-----------|-----------------------|
| fx calcPortP.divide | calcPortP | divide |
| fx calcPortP.getDivideResultCalEn | calcPortP | getDivideResultCalEn |
| fx calcPortP.getDivideResultCalVal | calcPortP | getDivideResultCalVal |
| fx calcPortP.setDivideResultCalEn | calcPortP | setDivideResultCalEn |
| fx calcPortP.setDivideResultCalVal | calcPortP | setDivideResultCalVal |

实例演示 – 模型配置 – 客户端

- 1) 更新服务发现策略 2) 映射 Persistence 元素

The screenshot illustrates the configuration of an AUTOSAR client model. The main workspace shows a block diagram with a 'step()' block and several 'calcPortR' blocks. The 'AUTOSAR Dictionary: calcClient' window is open, showing the 'calculatorInterface' and its methods. The 'Service Discovery Mode' dropdown is set to 'DynamicDiscovery'. The 'Code Mappings - Component Interface' window shows the mapping of persistence elements.

Code Mappings - Component Interface

| Source | Port | Event |
|--------------------------|-----------|----------------|
| calcPortR_divisionByZero | calcPortR | divisionByZero |

Code Mappings - Component Interface

| Source | Happed To |
|-------------------|---|
| dividResCalEn | Persistency > calcClient/calcClient_triggered_sys |
| dividResCalEnVar | Auto > calcClient/calcClient_triggered_sys |
| dividResCalVal | Persistency > calcClient/calcClient_triggered_sys |
| dividResCalValVar | Auto > calcClient/calcClient_triggered_sys |

Code Mappings - Component Interface

| Port | DataElement |
|------------------------------|------------------|
| prPortPrototype_KeyValueStor | divieResultCalEn |

Open in Property Inspector

实例演示 – 代码生成 - Server

The screenshot displays the Code Generation Report interface. The top bar shows navigation icons, a search field, and the current model name 'calcServer'. The left sidebar contains a 'Content' pane with a tree view of report sections and code files. The main area shows the C++ source code for 'calcServer.cpp'.

Content

- Summary
- Subsystem Report
- Code Interface Report
- Traceability Report
- Static Code Metrics Report
- Code Replacements Report
- Coder Assumptions

Code

- Model files
 - calcServer.cpp
 - calcServer.h
 - calcServer_private.h
 - calcServer_types.h
- Data files
 - calcServer_data.cpp
- Shared files
 - rtwtypes.h
- Interface files
 - GWM_calculator_Client.arxml
 - GWM_calculator_Per.arxml
 - GWM_calculator_SI.arxml
 - GWM_calculator_Server_machine.arxml
 - GWM_calculator_server.arxml
 - calcServer_ExecutionManifest.arxml
 - calcServer_ServiceInstanceManifest.arxml
 - calcServer_component.arxml
 - calcServer_datatype.arxml
 - standard_datatypes.arxml
- Other files
 - PosixExecutor.hpp
- Other files
 - calculatorinterface_common.h
 - calculatorinterface_skeleton.h
 - calculatorinterface_skeleton_impl.h
- Other files
 - main.cpp

calcServer.cpp

```
1 //
2 // calcServer.cpp
3 //
4 // Trial License - for use to evaluate programs for possible purchase as
5 // an end-user only.
6 //
7 // Code generation for model "calcServer".
8 //
9 // Model version          : 1.2
10 // Simulink Coder version : 9.9 (R2023a) 19-Nov-2022
11 // C++ source code generated on : Wed Mar 29 10:54:42 2023
12 //
13 // Target selection: autosar_adaptive.tlc
14 // Embedded hardware selection: Intel->X86-64 (Windows64)
15 // Code generation objectives: Unspecified
16 // Validation result: Not run
17
18
19 #include "calcServer.h"
20 #include "rtwtypes.h"
21
22 // Model step function
23 void calcServer::divide(uint32_T dividend, uint32_T divisor, uint32_T *result)
24 {
25     // Outputs for Function Call SubSystem: '<Root>/calcPortP_divide_sys'
26     // If: '<S1>/If' incorporates:
27     //   SignalConversion generated from: '<S1>/divisor'
28
29     if (divisor != 0U) {
30         // Outputs for IfAction SubSystem: '<S1>/If Action Subsystem' incorporates:
31         //   ActionPort: '<S7>/Action Port'
32
33         // SignalConversion generated from: '<S1>/result' incorporates:
34         //   Product: '<S7>/Divide'
35         //   SignalConversion generated from: '<S1>/divident'
36
37         *result = dividend / divisor;
38
39         // End of Outputs for SubSystem: '<S1>/If Action Subsystem'
40     } else {
41         // Outputs for IfAction SubSystem: '<S1>/If Action Subsystem1' incorporates:
42         //   ActionPort: '<S8>/Action Port'
43
44         // SignalConversion generated from: '<S1>/result' incorporates:
45         //   Constant: '<S8>/Constant'
46         //   SignalConversion generated from: '<S8>/Out1'
47     }
```

实例演示 – 代码生成 - Client

The screenshot displays the 'Code Generation Report' window. The top bar includes navigation icons, a 'Find:' search field, and a 'Match Case' checkbox. The current model is identified as 'calcClient'. The interface is divided into two main sections: 'Content' and 'Code'.

Content Section:

- Summary
- Subsystem Report
- Code Interface Report
- Traceability Report
- Static Code Metrics Report
- Code Replacements Report
- Coder Assumptions

Code Section:

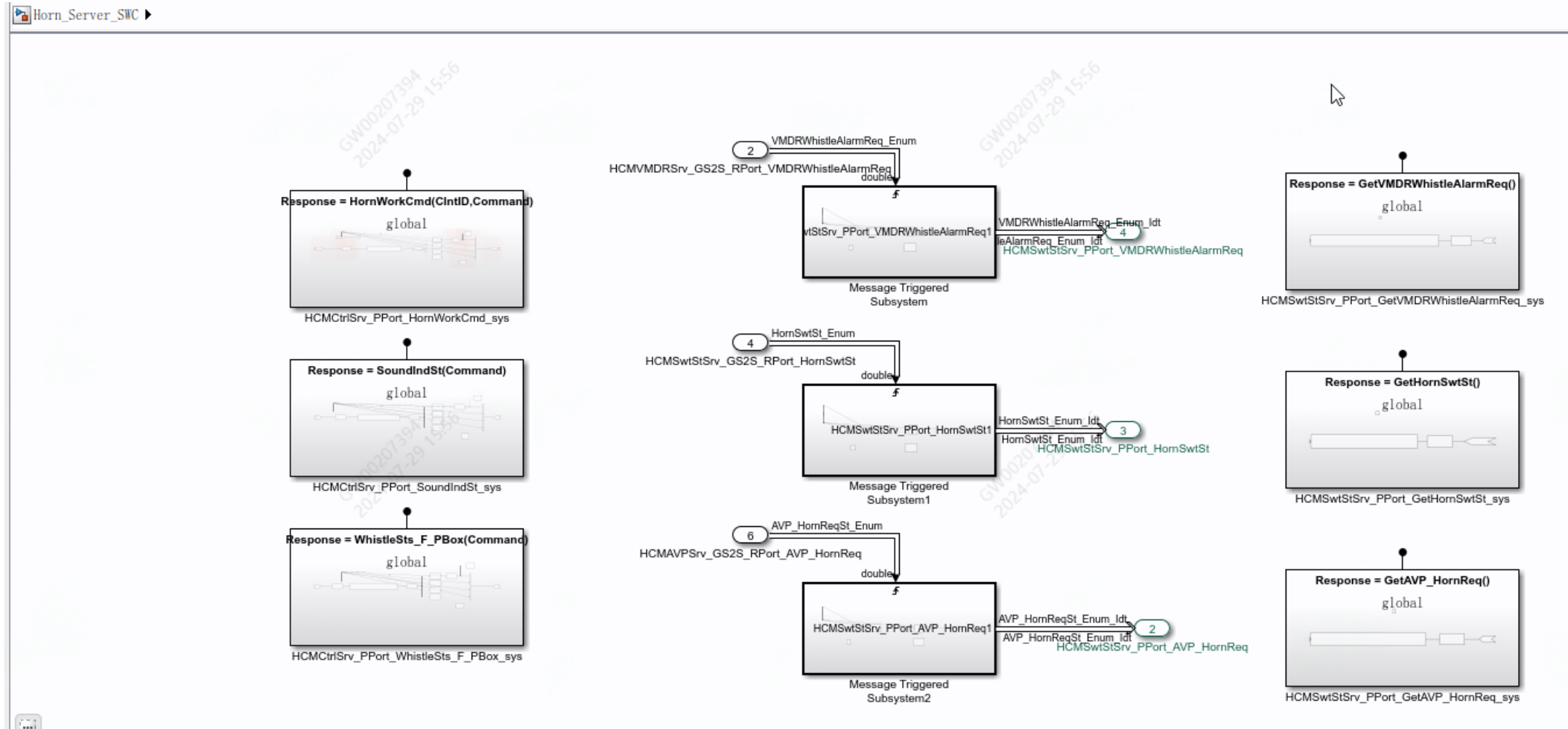
- Model files
 - calcClient.cpp** (highlighted)
 - calcClient.h
 - calcClient_private.h
 - calcClient_types.h
- Shared files
 - rtwtypes.h
- Interface files
 - GWM_calculator_Client.arxml
 - GWM_calculator_Per.arxml
 - GWM_calculator_Sl.arxml
 - GWM_calculator_Server_machine.arxml
 - GWM_calculator_server.arxml
 - calcClient_ExecutionManifest.arxml
 - calcClient_ServiceInstanceManifest.arxml
 - calcClient_component.arxml
 - calcClient_datatype.arxml
 - standard_datatypes.arxml
- Other files
 - PosixExecutor.hpp
- Other files
 - main.cpp
- ARA files
 - calculatorinterface_common.h
 - calculatorinterface_proxy.h

Code Section:

The code editor shows the content of 'calcClient.cpp'. The code includes headers, license information, and function definitions for the calculator client.

```
1 //
2 // calcClient.cpp
3 //
4 // Trial License - for use to evaluate programs for possible purchase as
5 // an end-user only.
6 //
7 // Code generation for model "calcClient".
8 //
9 // Model version          : 1.9
10 // Simulink Coder version : 9.9 (R2023a) 19-Nov-2022
11 // C++ source code generated on : Sun Apr  2 13:51:36 2023
12 //
13 // Target selection: autosar_adaptive.tlc
14 // Embedded hardware selection: Intel->x86-64 (Windows64)
15 // Code generation objectives: Unspecified
16 // Validation result: Not run
17
18
19 #include "calcClient.h"
20 #include "rtwtypes.h"
21
22 void calcClient::calcPortRSvchandler(ara::com::ServiceHandleContainer< vector::
23 calculatorService::proxy::calculatorInterfaceProxy::HandleType > svchandles,
24 const ara::com::FindServiceHandle fsHandle)
25 {
26     if (!calcPortR && (svchandles.size() > 0)) {
27         calcPortR = std::make_shared< vector::calculatorService::proxy::
28 calculatorInterfaceProxy >(*svchandles.begin());
29         calcPortR->divideResultCalEn.Subscribe(1U);
30         calcPortR->divideResultCalVal.Subscribe(1U);
31         calcPortR->divisionByZero.Subscribe(1U);
32         vector::calculatorService::proxy::calculatorInterfaceProxy::StopFindService
33 (fsHandle);
34     }
35 }
36
37 void calcClient::calcPortRdivideResultCalEnReceive(ara::com::SamplePtr< vector::
38 calculatorService::proxy::events::divideResultCalEn::SampleType const >
39 elementPtr)
40 {
41     // Receive: '<S1>/Event Receive2'
42     calcClient_B.EventReceive2 = *elementPtr;
43 }
44
45 void calcClient::calcPortRdivideResultCalValReceive(ara::com::SamplePtr< vector::
46 calculatorService::proxy::events::divideResultCalVal::SampleType const >
47 elementPtr)
```


项目实例



SOA开发中的问题

标准实施不一致问题：各家实施规范中，并没有完全遵循规范，导致架构模型生成的代码和Simulink产生的代码不完全匹配，需要手动修改

Simulink目前不支持PHM

Simulink对Log & Trace 的支持有待提高

总结

- 可以基于Simulink 开发符合CP和AP标准的SOA软件
- 利用Simulink开发SOA软件可以复用已有的软件
- 利用Simulink开发AP软件可以规避C++人员资源不足的问题
- CP环境下的开发工具链相对完善
- AP环境下的开发工具链还需要进一步完善

2024 MathWorks 中国汽车年会

Thank you

