Document Title	SWS_RTE: Complete Change Documentation 4.3.0 - 4.3.1
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	695

Document Status	Final
Part of AUTOSAR Standard	Classic Platform
Part of Standard Release	4.3.1

Table of Contents

1 SWS_RTE

1.1 Specification Item ECUC_Rte_09000

Trace References:

none

Content:

Module Name	RteRte
Module Description	Configuration of the Rte (Runtime Environment) module.
Post-Build Variant Support	true
Supported Config Variants	VARIANT-POST-BUILD, VARIANT-PRE-COMPILE

Included containers:

Included Containers		
Container Name	Multiplicity	Scope / Dependency
RteBswGeneral	1	General configuration parameters of the Bsw Scheduler section.
RteBswModuleInstance	0..*	Represents one instance of a Bsw-Module configured on one ECU.
RteGeneration	1	This container holds the parameters for the configuration of the RTE Generation.
RteImplicitCommunication	0..*	Configuration of the Implicit Communication behavior to be generated.
RteInitializationBehavior	1..*	Specifies the initialization strategy for variables allocated by RTE with the purpose to implement VariableData Prototypes. The container defines a set of Rte SectionInitializationPolicys and one Rte InitializationStrategy which is applicable for this set.
RteInitializationRunnableBatch	0..*	This container corresponds to an Rte_Init_<shortName of this container> function invoking the mapped Runnable Entities.
RteOsInteraction	1..*	Interaction of the Rte with the Os.
RtePostBuildVariantConfiguration	0..1	Specifies the PostbuildVariantSets for each of the PostBuild configurations of the RTE. The shortName of this container defines the name of the RtePostBuildVariant.

Included Containers		
Container Name	Multiplicity	Scope / Dependency
RteSwComponentInstance	0..*	Representation of one SwComponent Prototype located on the to be configured ECU. All subcontainer configuration aspects are in relation to this SwComponentPrototype. The RteSwComponentInstance can be associated with either a AtomicSw ComponentType or ParameterSw ComponentType.
RteSwComponentType	0..*	Representation of one SwComponent Type for the base of all configuration parameter which are affecting the whole type and not a specific instance.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77386: MultipleConfigurationContainer occurrences

  Problem description:

  As far as I know, MultipleConfigurationContainer type has been removed from the standard.
  However, there are some SWS documents which mention MultipleConfigurationContainer.

  COM:
  Figure 10: ComConfig: "multipleConfigurationContainer = true"
  ECUC_Com_00540: "This container is a MultipleConfigurationContainer, i.e. this container and its sub-containers exist once per configuration set."

  J1939Nm:
  ECUC_J1939Nm_00027, ECUC_J1939Nm_00028: "This container is a MultipleConfigurationContainer, i.e. this container and its sub-containers exist once per configuration set."

  PduR:
  Figure 33: ComConfig: "multipleConfigurationContainer = true", CanIfInitCfg: "multipleConfigurationContainer = true"

  Rte:
  Chapter 5.3.10: "RtePostBuildVariantConfiguration is a multipleConfigurationContainer"

  Agreed solution:

COM:
Remove multipleConfigurationContainer from (description of) ComConfig and regenerate and update chap10/Com.html and Figure 10 (The AUTOSAR COM modules Configuration Overview).

==========

Rte:
- In chapter 5.3.10 remove block "RtePostBuildVariantConfiguration is a multipleConfigurationContainer... post build configurable inside the RTE."

- In chapter 5.3.10.3 replace "And likewise for the example 2 header file the RTE generator can declare and initialize in the Rte_PBcfg.c file all possible PostBuildVariantCriterionValueSets
and the RtePostBuildVariantConfigurations using references to these variant sets." with "And likewise for the example 2 header file the RTE generator can declare and initialize in the Rte_PBcfg.c file all possible PostBuildVariantCriterionValueSets and the RtePostBuildVariantConfiguration using references to these variant sets."

- In chapter 7.4 replace "Each instance of this container specifies one PostBuild variant of the generated Rte. The shortName of the container RtePostBuildVariantConfiguration specifies the variant name." with "Each instance of RtePostBuildUsedPredefinedVariant inside this container specifies one PostBuild variant of the generated Rte. The shortName of the RtePostBuildUsedPredefinedVariant specifies the variant name."

- [ECUC_Rte_09084]: Remove "The shortName of this container defines the name of the RtePostBuildVariant."

- [ECUC_Rte_09083]: Add "The shortName of the referenced PredefinedVariant defines the name of the RtePostBuildVariant."

==========

J1939Nm:
Change the description of J1939NmConfigSet (ECUC_J1939Nm_00027) to: "This container contains the configuration parameters and sub containers of the AUTOSAR J1939Nm module."

==========

PduR:
Remove multipleConfigurationContainer from ComConfig and CanIfInitConfig on

Figure 29.

==========
–Last change on issue 77386 comment 8–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.2 Specification Item ECUC_Rte_09083

Trace References:

none

Content:

Name	RtePostBuildUsedPredefinedVariantRtePostBuildVariantConfiguration.RtePostBuildUsed PredefinedVariant		
Parent Container	RtePostBuildVariantConfiguration		
Description	Reference to the PredefinedVariant element which defines the values for PostBuildVariation Criterion elements. The shortName of the referenced PredefinedVariant defines the name of the RtePostBuild Variant.		
Multiplicity	1..*		
Type	Foreign reference to [PREDEFINED-VARIANT]		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77386: MultipleConfigurationContainer occurrences

  Problem description:

  As far as I know, MultipleConfigurationContainer type has been removed from the standard.
  However, there are some SWS documents which mention MultipleConfigura-

tionContainer.

COM:
Figure 10: ComConfig: "multipleConfigurationContainer = true"
ECUC_Com_00540: "This container is a MultipleConfigurationContainer, i.e. this container and its sub-containers exist once per configuration set."

J1939Nm:
ECUC_J1939Nm_00027, ECUC_J1939Nm_00028: "This container is a MultipleConfigurationContainer, i.e. this container and its sub-containers exist once per configuration set."

PduR:
Figure 33: ComConfig: "multipleConfigurationContainer = true", CanIfInitCfg: "multipleConfigurationContainer = true"

Rte:
Chapter 5.3.10: "RtePostBuildVariantConfiguration is a multipleConfigurationContainer"

Agreed solution:

COM:
Remove multipleConfigurationContainer from (description of) ComConfig and regenerate and update chap10/Com.html and Figure 10 (The AUTOSAR COM modules Configuration Overview).

==========

Rte:
- In chapter 5.3.10 remove block "RtePostBuildVariantConfiguration is a multipleConfigurationContainer... post build configurable inside the RTE."

- In chapter 5.3.10.3 replace "And likewise for the example 2 header file the RTE generator can declare and initialize in the Rte_PBcfg.c file all possible PostBuildVariantCriterionValueSets
and the RtePostBuildVariantConfigurations using references to these variant sets."
with "And likewise for the example 2 header file the RTE generator can declare and initialize in the Rte_PBcfg.c file all possible PostBuildVariantCriterionValueSets and the RtePostBuildVariantConfiguration using references to these variant sets."

- In chapter 7.4 replace "Each instance of this container specifies one PostBuild variant of the generated Rte. The shortName of the container RtePostBuildVariantConfiguration specifies the variant name." with "Each instance of

RtePostBuildUsedPredefinedVariant inside this container specifies one PostBuild variant of the generated Rte. The shortName of the RtePostBuildUsedPredefined-Variant specifies the variant name."

- [ECUC_Rte_09084]: Remove "The shortName of this container defines the name of the RtePostBuildVariant."

- [ECUC_Rte_09083]: Add "The shortName of the referenced PredefinedVariant defines the name of the RtePostBuildVariant."

==========

J1939Nm:
Change the description of J1939NmConfigSet (ECUC_J1939Nm_00027) to: "This container contains the configuration parameters and sub containers of the AUTOSAR J1939Nm module."

==========

PduR:
Remove multipleConfigurationContainer from ComConfig and CanIfInitConfig on Figure 29.

==========
–Last change on issue 77386 comment 8–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.3 Specification Item ECUC_Rte_09084

Trace References:

none

Content:

Container Name	RtePostBuildVariantConfigurationRtePostBuildVariantConfiguration
Description	Specifies the PostbuildVariantSets for each of the PostBuild configurations of the RTE. The shortName of this container defines the name of the RtePostBuildVariant.
Configuration Parameters	

Included parameters:

Included Parameters	
Parameter Name	SWS Item ID
RtePostBuildUsedPredefinedVariant	ECUC_Rte_09083

Included containers:

No Included Containers

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77386: MultipleConfigurationContainer occurrences

  Problem description:

  As far as I know, MultipleConfigurationContainer type has been removed from the standard.
  However, there are some SWS documents which mention MultipleConfigurationContainer.

  COM:
  Figure 10: ComConfig: "multipleConfigurationContainer = true"
  ECUC_Com_00540: "This container is a MultipleConfigurationContainer, i.e. this container and its sub-containers exist once per configuration set."

  J1939Nm:
  ECUC_J1939Nm_00027, ECUC_J1939Nm_00028: "This container is a MultipleConfigurationContainer, i.e. this container and its sub-containers exist once per configuration set."

  PduR:
  Figure 33: ComConfig: "multipleConfigurationContainer = true", CanIfInitCfg: "multipleConfigurationContainer = true"

  Rte:
  Chapter 5.3.10: "RtePostBuildVariantConfiguration is a multipleConfigurationContainer"

  Agreed solution:

  COM:
  Remove multipleConfigurationContainer from (description of) ComConfig and regenerate and update chap10/Com.html and Figure 10 (The AUTOSAR COM

modules Configuration Overview).

==========

Rte:
- In chapter 5.3.10 remove block "RtePostBuildVariantConfiguration is a multipleConfigurationContainer... post build configurable inside the RTE."

- In chapter 5.3.10.3 replace "And likewise for the example 2 header file the RTE generator can declare and initialize in the Rte_PBcfg.c file all possible PostBuildVariantCriterionValueSets
and the RtePostBuildVariantConfigurations using references to these variant sets." with "And likewise for the example 2 header file the RTE generator can declare and initialize in the Rte_PBcfg.c file all possible PostBuildVariantCriterionValueSets and the RtePostBuildVariantConfiguration using references to these variant sets."

- In chapter 7.4 replace "Each instance of this container specifies one PostBuild variant of the generated Rte. The shortName of the container RtePostBuildVariantConfiguration specifies the variant name." with "Each instance of RtePostBuildUsedPredefinedVariant inside this container specifies one PostBuild variant of the generated Rte. The shortName of the RtePostBuildUsedPredefinedVariant specifies the variant name."

- [ECUC_Rte_09084]: Remove "The shortName of this container defines the name of the RtePostBuildVariant."

- [ECUC_Rte_09083]: Add "The shortName of the referenced PredefinedVariant defines the name of the RtePostBuildVariant."

==========

J1939Nm:
Change the description of J1939NmConfigSet (ECUC_J1939Nm_00027) to: "This container contains the configuration parameters and sub containers of the AUTOSAR J1939Nm module."

==========

PduR:
Remove multipleConfigurationContainer from ComConfig and CanIfInitConfig on Figure 29.

==========
–Last change on issue 77386 comment 8–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.4 Specification Item ECUC_Rte_09085

Trace References:

none

Content:

Name	RteDevErrorDetectUninitRteGeneration.RteDevErrorDetectUninit		
Parent Container	RteGeneration		
Description	The Rte shall detect if it is started when its APIs are called, and the BSW Scheduler shall check if it is initialized when its APIs are called.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local dependency: Shall only be used when RteDevErrorDetect equals true.		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #75045: [Cleanup] Cleanup Constraint Handling in SWS items: Rollout replacement of dependencies by constraints

  Problem description:

  AUTOSAR specified in the field "Dependency" in the chapter 10 description in an informal way dependencies between different configuration parameters. On the other hand some constraints are created in single BSW modules with the same namespace as the metamodel constraints. This should be cleaned up (see presentation in the attachment).

  Agreed solution:

  Rollout to replace in each SWS the dependencies by constraints (could be done step by step, not mandatory in one release)

When all SWS specifications are cleaned up the dependency field can be removed from the database

FIM
—-
Remove dependency and in the following configuration parameters and remove the following text from the description field "At least one FiMInhSumRef or FiMInhEventRef or
FiMInhComponentRef needs to be configured." :

FiMInhEventRef [ECUC_FiM_00100]
FiMInhSumRef [ECUC_FiM_00102]
FiMInhComponentRef [ECUC_FiM_00605]

Create chapter 7.4 Configuration Constraints
Add new constraints :
SWS_FIM_CONSTR_XXX1 : For each configured FiMInhibitionConfiguration, at least one of FiMInhSumRef or FiMInhEventRef or FiMInhComponentRef shall be configured.

DEM
—-
see file under svn.:
https://svn.autosar.org/repos/work/09_WorkPackages/WP-A4/70_InternalDocuments/048_Dependency_Removal/DCM_Dem_Dependency_Removal.xlsx

DCM
—-
See file under svn :
https://svn.autosar.org/repos/work/09_WorkPackages/WP-A4/70_InternalDocuments/048_Dependency_Removal/DCM_Dem_Dependency_Removal.xlsx

Rte
—-
RteDevErrorDetectUninit: SWS_RTE_CONSTR_XXX1: In case that RteDevErrorDetectUninit is configured to true, RteDevErrorDetect shall be configured to true.
–Last change on issue 75045 comment 21–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.5 Specification Item ECUC_Rte_09094

Trace References:

none

Content:

Name	RteIocInteractionReturnValue		
Description	Defines whether the return value of RTE APIs is based on RTE-IOC interaction or RTE-COM interaction. This parameter is only applicable if the RTE utilizes IOC for inter partion communication.		
Multiplicity	1 0..1		
Type	EcucEnumerationParamDef		
Range	RTE_COM	–	
	RTE_IOC	–	
Default value	RTE_IOC		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #68369: Allow optimized inter-core communication mechanism

  Problem description:

  An internal evaluation of the IOC within Continental shows following drawbacks, among others:
  - IOC does not support multiple receivers (The 1:N communication is done with the help of the RTE by subsequently calling IOC)
  - No buffer reuse in IOC (i.e. increased resource overhead)
  - No shared remote and local usage of the IOC buffer (i.e. resource overhead with risk of coherency glitches)
  - No variable grouping of IOC read/write operations to build blocks (i.e. increased runtime overhead and risk of coherency glitches)
  - IOC uses copy operations even for read operations of primitive data (i.e. resource overhead)

- For memory partitioning the IOC is usually not needed. Instead the MPU is configured with read all policy.

The IOC seems from specification point not very well optimized with respect to resources and flexibility. It turns out the IOC is less efficient than individual vendor specific solutions.

Therefore we propose to allow the implementation of an optimized vendor specific implementation for inter-core communication. A vendor specific solution could support the individual domain specific needs regarding optimization of buffering and data protection. Also the system dynamics can be taken into account. Additionally such an implementation can be customized for a dedicated microcontroller.

The IOC module shall remain mandatory in the OS but it shall be allowed to use a different mechanism to cross core or memory partition boundaries. In case the IOC is not used, the inter-core communication could be done by a vendor specific RTE extension.

Referring the RTE and OS specification this seems not to be explicitly forbidden so far (to be proven by document owner). But both the specifications give the impression that the usage of the IOC is mandatory for inter-partition communication.

E.g.:
- AUTOSAR_SWS_OS (Rev. 4.2.1), chapter 7.10.2 IOC General purpose: To keep the RTE as hardware independent as possible, all inter OS-Application and inter core communication mechanisms and implementation variants are encapsulated in the IOC.
- AUTOSAR_SWS_RTE (Rev. 4.2.1), chapter 4.3 Communication Paradigms: For inter-Partition communication (within the same ECU) the RTE uses functionalities provided by the IOC module.

Therefore it should be explicitly mentioned that the inter-partition communication could be also realized by some optimized alternative mechanism (not using the IOC). Either directly realized by the RTE or by invoking some vendor specific module.

Agreed solution:

See attachment# 3657
–Last change on issue 68369 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.6 Specification Item ECUC_Rte_09107

Trace References:

none

Content:

Name	RteUseComShadowSignalApiRteBswGeneral.RteUseComShadowSignalApiin container RteBswGeneral		
Description	This parameter defines whether the ComShadowSignalAPIs ((Com_UpdateShadowSignal, Com_InvalidateShadowSignal, Com_ReceiveShadowSignal) are used or not. If this parameter is set to true the ShadowSignal APIs and Signal APIs (Com_SendSignal, Com_InvalidateSignal, Com_ReceiveSignal) are used. If this parameter is set to false only the Signal APIs (Com_SendSignal, Com_InvalidateSignal, Com_ReceiveSignal) are used. Tags: atp.Status=obsolete atp.StatusRevisionBegin=4.3.1		
Multiplicity	1 0..1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77468: [RTE] Remove not existing COM services

  Problem description:

  The following API's are removed from COM:
  Com_UpdateShadowSignal,
  Com_InvalidateShadowSignal,
  Com_ReceiveShadowSignal

  Thus, also these references to this COM APIs should be removed from the RTE specification.

  Agreed solution:

  === RTE ===
  1.) update [SWS_Rte_05173]
  [SWS_Rte_05173] d The RTE shall use the ComHandleId of the corresponding Com-
  GroupSignal when invoking the COM API for group signals. c

...
The input information ComHandleId of group signals is used to establish the link between
the ComGroupSignal of the COM modules configuration and the corresponding ISignal of the System Template.

2.) update text after [SWS_Rte_04527] in Chapter "4.3.1.11.2 Atomicity"

from "The RTE decomposes the composite data type into single signals as described
above and passes them to the COM module by using the COM API
call Com_SendSignal (if parameter RteUseComShadowSignalApi is FALSE) or Com_UpdateShadowSignal (if parameter RteUseComShadowSignalApi is TRUE).
...
As would be expected, the receiver side is the exact reverse of the transmission side: the RTE must first call Com_ReceiveSignalGroup precisely once for the signal group and then call Com_ReceiveSignal (if parameter RteUseComShadowSignalApi is FALSE) or Com_ReceiveShadowSignal (if parameter RteUseComShadowSignalApi
is TRUE) to extract the value of each signal within the signal group."
to
"The RTE decomposes the composite data type into single signals as described above and passes them to the COM module by using the COM API call Com_SendSignal."
...
As would be expected, the receiver side is the exact reverse of the transmission side: the RTE must first call Com_ReceiveSignalGroup precisely once for the signal group and then call Com_ReceiveSignal to extract the value of each signal within the signal group."

delete "shadow" in following sentence:
A signal group has the additional property that COM guarantees to inform the receiver
by invoking a call-back about its arrival only after all signals belonging to the signal group have been unpacked into a shadow buffer.

3.) update text after [SWS_Rte_06023]

[SWS_Rte_06023]
....
no Data Transformation: RTE invokes Com_SendSignal for each primitive element (ISignal) in the composite data type and each COM signal to which that primitive element is mapped, and Com_SendSignalGroup for each ISignalGroup

that does not require a Data Transformation to which the data element is mapped.
...
the RTE shall invoke Com_SendSignal for each ISignal to which an element in the composite data type is mapped and Com_SendSignalGroup for each ISignalGroup to which
the composite data element is mapped.

4.) Update "Chapter 4.3.1.16.1 COM"
Generally the communication of arrays in the case of inter-ECU communication must make use of the signal group mechanisms to send an array to COM.
This implies sending each array element to a buffer in COM (with Com_SendSignal() API, and in the end send the signal group (with Com_SendSignalGroup() API).

5.) update description of chapter "5.11.5.3.3 Signal Invalidation"
A trace event indicating a signal invalidation request of an
Inter-ECU signal (or of a signal in a signal group) by the RTE.
Invoked by the RTE just before Com_InvalidateSignal is invoked.

6.) update [SWS_Rte_04526]
...
the RTE shall invoke Com_SendSignal for each ISignal to which an element in the composite data type is mapped and Com_SendSignalGroup for each ISignalGroup to which the composite
data element is mapped.

=== ECUC ===

1.) Set [ECUC_Rte_09107] to obsolete
–Last change on issue 77468 comment 4–

BW-C-Level:

Application	Specification	Bus
3	4	1

1.7 Specification Item SWS_Rte_01058

Trace References:

SRS_BSW_00327

Content:

Symbolic name: |RTE_E_OK|
Value: 0
Comments: No error occurred.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77161: Tables in specification documents contain trace items

  Problem description:

  _____
  Name: Wolf-Hendrik Kaps
  Phone: 0711-811-23288
  Role: Jg-Tooling member
  _____
  Description/Motivation:

  Some specification documents contain tables which include trace items. E.g. SWS_Com, table in 7.12.1 Development Errors, SWS_Rte, Table 5.4: RTE Error and Status values.

  As discussed in tooling session we should extend ValidateARXML routine to elicit trace items inside tables.

  Further on we shall ensure that tracebles do not contain figures and tables (77206, 74860)
  –Last change on issue 77161 comment 3–

  Agreed solution:

  1. Affected Documents
  ==========

  1. Move the Traceable out of the f**ing tables (see attachment)
  2. move Tables and figures out of the Traceable

  2. Word2arxml and latex2arxml which is used by checkDocumentSource
  ==========

  let these scripts complain also requested by 77206, 74860 but summarized here

* Tables in Traceable
* Traceable in Tables
* Figures in Traceable

2.1 GST: add these contraints
==========

3. CP_Tool_Scripts
==========

* let latexinstatiaator complain about Figures in Traceable
* note: tha table in traceable violates the schema and is flagged already
* let latexinstantiator complain about traceable in tables
–Last change on issue 77161 comment 7–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.8 Specification Item SWS_Rte_01060

Trace References:

SRS_BSW_00327

Content:

Symbolic name: |RTE_E_COM_STOPPED|
Value: 128
Comments: An IPDU group was disabled while the application was waiting for the transmission acknowledgment. No value is available. This is not considered a fault, since the IPDU group is switched off on purpose.

This semantics are as follows:

- the OUT buffers of a client are not modified,
- the explicit read APIs read the last known value (or init value),
- no runnable with startOnEvent on a DataReceivedEvent for this, VariableDataPrototype is triggered,
- the buffers for implicit read access will keep the previous value.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77161: Tables in specification documents contain trace items

  Problem description:

  _____

  Name: Wolf-Hendrik Kaps
  Phone: 0711-811-23288
  Role: Jg-Tooling member
  _____

  Description/Motivation:

  Some specification documents contain tables which include trace items. E.g. SWS_Com, table in 7.12.1 Development Errors, SWS_Rte, Table 5.4: RTE Error and Status values.

  As discussed in tooling session we should extend ValidateARXML routine to elicit trace items inside tables.

  Further on we shall ensure that tracebles do not contain figures and tables (77206, 74860)
  –Last change on issue 77161 comment 3–

  Agreed solution:

  1. Affected Documents
  ==========

  1. Move the Traceable out of the f**ing tables (see attachment)
  2. move Tables and figures out of the Traceable

  2. Word2arxml and latex2arxml which is used by checkDocumentSource
  ==========

  let these scripts complain also requested by 77206, 74860 but summarized here

  * Tables in Traceable
  * Traceable in Tables
  * Figures in Traceable

  2.1 GST: add these contraints

==========

3. CP_Tool_Scripts
==========

* let latexinstatiaator complain about Figures in Traceable
* note: tha table in traceable violates the schema and is flagged already
* let latexinstantiator complain about traceable in tables
–Last change on issue 77161 comment 7–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.9 Specification Item SWS_Rte_01061

Trace References:

SRS_BSW_00327

Content:

Symbolic name: |RTE_E_NO_DATA|
Value: 131
Comments: An explicit read API call returned no data. (This is no error.)

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77161: Tables in specification documents contain trace items

  Problem description:

  _____
  Name: Wolf-Hendrik Kaps
  Phone: 0711-811-23288
  Role: Jg-Tooling member
  _____
  Description/Motivation:

  Some specification documents contain tables which include trace items. E.g. SWS_Com, table in 7.12.1 Development Errors, SWS_Rte, Table 5.4: RTE Error and Status values.

  As discussed in tooling session we should extend ValidateARXML routine to

elicit trace items inside tables.

Further on we shall ensure that tracebles do not contain figures and tables (77206, 74860)
–Last change on issue 77161 comment 3–

Agreed solution:

1. Affected Documents
==========

1. Move the Traceable out of the f**ing tables (see attachment)
2. move Tables and figures out of the Traceable

2. Word2arxml and latex2arxml which is used by checkDocumentSource
==========

let these scripts complain also requested by 77206, 74860 but summarized here

* Tables in Traceable
* Traceable in Tables
* Figures in Traceable

2.1 GST: add these contraints
==========

3. CP_Tool_Scripts
==========

* let latexinstatiaator complain about Figures in Traceable
* note: tha table in traceable violates the schema and is flagged already
* let latexinstantiator complain about traceable in tables
–Last change on issue 77161 comment 7–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.10 Specification Item SWS_Rte_01064

Trace References:

SRS_BSW_00327, SRS_Rte_00069

Content:

Symbolic name: |RTE_E_TIMEOUT|
Value: 129
Comments: A blocking API call returned due to expiry of a local timeout rather than the intended result. OUT buffers are not modified. The interpretation of this being an error depends on the application.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77161: Tables in specification documents contain trace items

  Problem description:

  _____
  Name: Wolf-Hendrik Kaps
  Phone: 0711-811-23288
  Role: Jg-Tooling member
  _____
  Description/Motivation:

  Some specification documents contain tables which include trace items. E.g. SWS_Com, table in 7.12.1 Development Errors, SWS_Rte, Table 5.4: RTE Error and Status values.

  As discussed in tooling session we should extend ValidateARXML routine to elicit trace items inside tables.

  Further on we shall ensure that tracebles do not contain figures and tables (77206, 74860)
  –Last change on issue 77161 comment 3–

  Agreed solution:

  1. Affected Documents
  ==========

  1. Move the Traceable out of the f**ing tables (see attachment)
  2. move Tables and figures out of the Traceable

2. Word2arxml and latex2arxml which is used by checkDocumentSource
==========

let these scripts complain also requested by 77206, 74860 but summarized here

* Tables in Traceable
* Traceable in Tables
* Figures in Traceable

2.1 GST: add these contraints
==========

3. CP_Tool_Scripts
==========

* let latexinstatiaator complain about Figures in Traceable
* note: tha table in traceable violates the schema and is flagged already
* let latexinstantiator complain about traceable in tables
–Last change on issue 77161 comment 7–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.11 Specification Item SWS_Rte_01065

Trace References:

SRS_BSW_00327

Content:

Symbolic name: |RTE_E_TRANSMIT_ACK|
Value: 132
Comments: Transmission acknowledgement received.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77161: Tables in specification documents contain trace items

Problem description:

Name: Wolf-Hendrik Kaps
Phone: 0711-811-23288
Role: Jg-Tooling member

Description/Motivation:

Some specification documents contain tables which include trace items. E.g. SWS_Com, table in 7.12.1 Development Errors, SWS_Rte, Table 5.4: RTE Error and Status values.

As discussed in tooling session we should extend ValidateARXML routine to elicit trace items inside tables.

Further on we shall ensure that tracebles do not contain figures and tables (77206, 74860)
–Last change on issue 77161 comment 3–

Agreed solution:

1. Affected Documents
==========

1. Move the Traceable out of the f**ing tables (see attachment)
2. move Tables and figures out of the Traceable

2. Word2arxml and latex2arxml which is used by checkDocumentSource
==========

let these scripts complain also requested by 77206, 74860 but summarized here

* Tables in Traceable
* Traceable in Tables
* Figures in Traceable

2.1 GST: add these contraints
==========

3. CP_Tool_Scripts

==========

* let latexinstatiaator complain about Figures in Traceable
* note: tha table in traceable violates the schema and is flagged already
* let latexinstantiator complain about traceable in tables
–Last change on issue 77161 comment 7–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.12 Specification Item SWS_Rte_01106

Trace References:

SRS_Rte_00094

Content:

|RTE_E_COM_STOPPED| – the RTE could not perform the operation because the COM communication service is currently not available (inter ECU communication only). RTE shall return |RTE_E_COM_STOPPED| whenthe corresponding COM :

- in case of COM the corresponding service returns |COM_SERVICE_NOT_AVAILABLE| .
- in case of LdCom the corresponding LdCom_Transmit returns |E_NOT_OK|

The buffers of the return parameters shall not be modified.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76094: [Rte][LdCom] Reaction on E_NOT_OK when calling LdCom_Transmit

  Problem description:

  It is specified, that Rte_Write and Rte_Send shall return RTE_E_OK when
  "data passed to communication service _successfully_"
  But what shall happen if LdCom_Transmit returns E_NOT_OK ?

  Agreed solution:

  On behalf of WP-A1:

  Rephrase
  "[SWS_Rte_07822] RTE_E_COM_STOPPED the RTE could

not perform the operation because the communication service is currently not available (inter ECU communication only). RTE shall return RTE_E_COM_STOPPED when:
-in case of COM the corresponding COM service returns COM_SERVICE_NOT_AVAILABLE
-in case of LdCom the corresponding LdCom_Transmit returns E_NOT_OK
."

Do the same for SWS_Rte_07823, SWS_Rte_01106, SWS_Rte_06828 and SWS_Rte_01339

SWS_Rte_01379: replace RTE_E_OK with E_OK.
–Last change on issue 76094 comment 4–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.13 Specification Item SWS_Rte_01238

Trace References:

SRS_Rte_00045, SRS_Rte_00003, SRS_Rte_00004

Content:

```
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Start
     ([const Rte_CDS_<cts>*, ]<param>)
```

Where |<api>| is the RTE API Name (Write, Call, etc.),

|<cts>| is the component type symbol of the AtomicSwComponentType and

|<ap>| the access point name (e.g. port and data element or operation name, exclusive area name, etc.). The parameters of the API shall be the same as the corresponding RTE API. As with the API itself, the instance handle is included if and only if the software component's SwcInternalBehavior.supportsMultipleInstantiation attribute is set to true and the RTE API function is per-instance. Thus the instance handle is always omitted for SWCs supporting single instantiation and also for per-SWC functions, such as Rte_CData for shared ParameterDataPrototypes, for SWCs supporting multiple instantiation. Note that |Rte_Instance| cannot be used directly, as there will be pointers to multiple components' structure types within the single VFB Tracing header file, and |Rte_Instance| would therefore be ambiguous.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76346: Component data structure type definition vs. structure declaration

  Problem description:

  In general the RTE SWS distinguishes between a structure declaration and a type declaration for a composite datatype that represents a structure.

  For a structure implementation data type SWS_Rte_07114 requires a structure declaration:

  struct _<name>_
  ...
  ;

  SWS_Rte_06812 requires the type declaration:

  typedef struct _<name>_ <name>;

  Regarding the component data structure currently there exist the following requirements only:

  [SWS_Rte_07132] The component data structure type shall be defined in the Application Header file.
  [SWS_Rte_03714] The type name of the component data structure shall be Rte_[Byps]_CDS_<cts> where <cts> is the component type symbol of the AtomicSwComponentType. ...

  Since SWS_Rte_03714 talks about the type name of the component data structure this requires a typedef statement for Rte_[Byps]_CDS_<cts> in the Application Header file:

  typedef struct ... Rte_[Byps]_CDS_<cts>;

  Currently there does not exist any requirement for a CDS structure declaration.

  The code examples in the SWS RTE are somehow in contradiction to these requirements. They use "struct

  Rte_CDS_<cts>" instead of the typedef name. See examples 5.24, 5.26, 5.28.

  This inconsistency applies to the signature of some APIs, too:

[SWS_Rte_01238]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Start([const struct Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01239]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Return([const struct Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01248]
void Rte_[<client>_]Runnable_<cts>_<reName>_Start([const RTE_CDS_<cts>*])

[SWS_Rte_01249]
void Rte_[<client>_]Runnable_<cts>_<reName>_Return([const Rte_CDS_<cts>*])

[SWS_Rte_06113]
void Rte_[<client>_]RptHook_<cts>_<var>_Transmit([const RTE_CDS_<cts>*], <type>* <buffer>)

[SWS_Rte_06114]
void Rte_[<client>_]RptHook_<cts>_<var>_Reception([const RTE_CDS_<cts>*], <type>* <buffer>)

In addition some of these APIs use RTE_CDS_ instead of Rte_CDS_.

Agreed solution:

Delete SWS_Rte_07132 and SWS_Rte_03714.

Add new requirements:

[SWS_Rte_xxxx1] The Application Header File shall include a structure declaration for the component data structure as follows:
struct Rte_[Byps_]CDS_<cts> <component data sections> ;
where <cts> is the component type symbol of the AtomicSwComponentType. [Byps_] is an optional infix used when component wrapper method for bypass support is enabled for the related software componenttype (See chapter 4.9.2).

[SWS_Rte_xxxx2] The Application Header File shall include a type declaration for the component data structure type as follows:
typedef struct Rte_[Byps_]CDS_<cts> Rte_[Byps_]CDS_<cts>;
where <cts> is the component type symbol of the AtomicSwComponentType.[Byps_] is an optional infix used when component wrapper method for bypass support is enabled for the related software componenttype (See chapter 4.9.2).

Change the API signatures to:

[SWS_Rte_01238]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Start([const Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01239]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Return([const Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01248]
void Rte_[<client>_]Runnable_<cts>_<reName>_Start([const Rte_CDS_<cts>*])

[SWS_Rte_06113]
void Rte_[<client>_]RptHook_<cts>_<var>_Transmit([const Rte_CDS_<cts>*], <type>* <buffer>)

[SWS_Rte_06114]
void Rte_[<client>_]RptHook_<cts>_<var>_Reception([const Rte_CDS_<cts>*], <type>* <buffer>)

Change examples 5.24, 5.26, 5.28 so that they contain both structure declaration and typedef for component data structure type and use the type (without struct) at the interfaces.
–Last change on issue 76346 comment 1–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.14 Specification Item SWS_Rte_01239

Trace References:

SRS_Rte_00045

Content:

```
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Return
     ([const Rte_CDS_<cts>*, ]<param>)
```

Where |<api>| is the RTE API Name (Write, Call, etc.),

|<cts>| is the component type symbol of the AtomicSwComponentType and

|<ap>| the access point name (e.g. port and data element or operation name, exclusive area name, etc.).

The parameters of the API are the same as the corresponding RTE API and contain the values of OUT and INOUT parameters on exit from the function.

As with the API itself, the instance handle is included if and only if the software component's SwcInternalBehavior.supportsMultipleInstantiation attribute is set to true and the RTE API function is per-instance. Thus the instance handle is always omitted for SWCs supporting single instantiation and also for per-SWC functions, such as Rte_CData for shared ParameterDataPrototypes, for SWCs supporting multiple instantiation. Note that |Rte_Instance| cannot be used directly, as there will be pointers to multiple components' structure types within the single VFB Tracing header file, and |Rte_Instance| would therefore be ambiguous.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76346: Component data structure type definition vs. structure declaration

  Problem description:

  In general the RTE SWS distinguishes between a structure declaration and a type declaration for a composite datatype that represents a structure.

  For a structure implementation data type SWS_Rte_07114 requires a structure declaration:

  struct _<name>_
  ...
  ;

  SWS_Rte_06812 requires the type declaration:

  typedef struct _<name>_ <name>;

  Regarding the component data structure currently there exist the following requirements only:

  [SWS_Rte_07132] The component data structure type shall be defined in the Application Header file.
  [SWS_Rte_03714] The type name of the component data structure shall be Rte_[Byps]_CDS_<cts> where <cts> is the component type symbol of the AtomicSwComponentType. ...

Since SWS_Rte_03714 talks about the type name of the component data structure this requires a typedef statement for Rte_[Byps]_CDS_<cts> in the Application Header file:

typedef struct ... Rte_[Byps]_CDS_<cts>;

Currently there does not exist any requirement for a CDS structure declaration.

The code examples in the SWS RTE are somehow in contradiction to these requirements. They use "struct

Rte_CDS_<cts>" instead of the typedef name. See examples 5.24, 5.26, 5.28.

This inconsistency applies to the signature of some APIs, too:

[SWS_Rte_01238]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Start([const struct Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01239]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Return([const struct Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01248]
void Rte_[<client>_]Runnable_<cts>_<reName>_Start([const RTE_CDS_<cts>*])

[SWS_Rte_01249]
void Rte_[<client>_]Runnable_<cts>_<reName>_Return([const Rte_CDS_<cts>*])

[SWS_Rte_06113]
void Rte_[<client>_]RptHook_<cts>_<var>_Transmit([const RTE_CDS_<cts>*], <type>* <buffer>)

[SWS_Rte_06114]
void Rte_[<client>_]RptHook_<cts>_<var>_Reception([const RTE_CDS_<cts>*], <type>* <buffer>)

In addition some of these APIs use RTE_CDS_ instead of Rte_CDS_.

Agreed solution:

Delete SWS_Rte_07132 and SWS_Rte_03714.

Add new requirements:

[SWS_Rte_xxxx1] The Application Header File shall include a structure declaration for the component data structure as follows:
struct Rte_[Byps_]CDS_<cts> <component data sections> ;
where <cts> is the component type symbol of the AtomicSwComponentType. [Byps_] is an optional infix used when component wrapper method for bypass support is enabled for the related software componenttype (See chapter 4.9.2).

[SWS_Rte_xxxx2] The Application Header File shall include a type declaration for the component data structure type as follows:
typedef struct Rte_[Byps_]CDS_<cts> Rte_[Byps_]CDS_<cts>;
where <cts> is the component type symbol of the AtomicSwComponentType.[Byps_] is an optional infix used when component wrapper method for bypass support is enabled for the related software componenttype (See chapter 4.9.2).

Change the API signatures to:

[SWS_Rte_01238]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Start([const Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01239]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Return([const Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01248]
void Rte_[<client>_]Runnable_<cts>_<reName>_Start([const Rte_CDS_<cts>*])

[SWS_Rte_06113]
void Rte_[<client>_]RptHook_<cts>_<var>_Transmit([const Rte_CDS_<cts>*], <type>* <buffer>)

[SWS_Rte_06114]
void Rte_[<client>_]RptHook_<cts>_<var>_Reception([const Rte_CDS_<cts>*], <type>* <buffer>)

Change examples 5.24, 5.26, 5.28 so that they contain both structure declaration and typedef for component data structure type and use the type (without struct) at the interfaces.
–Last change on issue 76346 comment 1–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.15 Specification Item SWS_Rte_01248

Trace References:

SRS_Rte_00045

Content:

```
void Rte_[<client>_]Runnable_<cts>_<reName>_Start
     ([const Rte_CDS_<cts>*])
```

Where |<cts>| is the component type symbol of the AtomicSwComponentType

and |reName| the runnable entity name.

The instance handle is included if and only if the software component's SwcInternalBehavior.supportsMultipleInstantiation attribute is set to true. Note that |Rte_Instance| cannot be used directly, as there will be pointers to multiple components' structure types within the single VFB Tracing header file, and |Rte_Instance| would therefore be ambiguous.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76346: Component data structure type definition vs. structure declaration

  Problem description:

  In general the RTE SWS distinguishes between a structure declaration and a type declaration for a composite datatype that represents a structure.

  For a structure implementation data type SWS_Rte_07114 requires a structure declaration:

  struct _<name>_
  ...
  ;

  SWS_Rte_06812 requires the type declaration:

  typedef struct _<name>_ <name>;

Regarding the component data structure currently there exist the following requirements only:

[SWS_Rte_07132] The component data structure type shall be defined in the Application Header file.
[SWS_Rte_03714] The type name of the component data structure shall be Rte_[Byps]_CDS_<cts> where <cts> is the component type symbol of the AtomicSwComponentType. ...

Since SWS_Rte_03714 talks about the type name of the component data structure this requires a typedef statement for Rte_[Byps]_CDS_<cts> in the Application Header file:

typedef struct ... Rte_[Byps]_CDS_<cts>;

Currently there does not exist any requirement for a CDS structure declaration.

The code examples in the SWS RTE are somehow in contradiction to these requirements. They use "struct

Rte_CDS_<cts>" instead of the typedef name. See examples 5.24, 5.26, 5.28.

This inconsistency applies to the signature of some APIs, too:

[SWS_Rte_01238]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Start([const struct Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01239]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Return([const struct Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01248]
void Rte_[<client>_]Runnable_<cts>_<reName>_Start([const RTE_CDS_<cts>*])

[SWS_Rte_01249]
void Rte_[<client>_]Runnable_<cts>_<reName>_Return([const Rte_CDS_<cts>*])

[SWS_Rte_06113]
void Rte_[<client>_]RptHook_<cts>_<var>_Transmit([const RTE_CDS_<cts>*], <type>* <buffer>)

[SWS_Rte_06114]
void Rte_[<client>_]RptHook_<cts>_<var>_Reception([const RTE_CDS_<cts>*], <type>* <buffer>)

In addition some of these APIs use RTE_CDS_ instead of Rte_CDS_.

Agreed solution:

Delete SWS_Rte_07132 and SWS_Rte_03714.

Add new requirements:

[SWS_Rte_xxxx1] The Application Header File shall include a structure declaration for the component data structure as follows:
struct Rte_[Byps_]CDS_<cts> <component data sections> ;
where <cts> is the component type symbol of the AtomicSwComponentType. [Byps_] is an optional infix used when component wrapper method for bypass support is enabled for the related software componenttype (See chapter 4.9.2).

[SWS_Rte_xxxx2] The Application Header File shall include a type declaration for the component data structure type as follows:
typedef struct Rte_[Byps_]CDS_<cts> Rte_[Byps_]CDS_<cts>;
where <cts> is the component type symbol of the AtomicSwComponentType.[Byps_] is an optional infix used when component wrapper method for bypass support is enabled for the related software componenttype (See chapter 4.9.2).

Change the API signatures to:

[SWS_Rte_01238]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Start([const Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01239]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Return([const Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01248]
void Rte_[<client>_]Runnable_<cts>_<reName>_Start([const Rte_CDS_<cts>*])

[SWS_Rte_06113]
void Rte_[<client>_]RptHook_<cts>_<var>_Transmit([const Rte_CDS_<cts>*], <type>* <buffer>)

[SWS_Rte_06114]

void Rte_[<client>_]RptHook_<cts>_<var>_Reception([const Rte_CDS_<cts>*], <type>* <buffer>)

Change examples 5.24, 5.26, 5.28 so that they contain both structure declaration and typedef for component data structure type and use the type (without struct) at the interfaces.
–Last change on issue 76346 comment 1–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.16 Specification Item SWS_Rte_01317

Trace References:

SRS_BSW_00327

Content:

Symbolic name: |RTE_E_LIMIT|
Value: 130
Comments: A internal RTE limit has been exceeded. Request could not be handled. OUT buffers are not modified.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77161: Tables in specification documents contain trace items

  Problem description:

  Name: Wolf-Hendrik Kaps
  Phone: 0711-811-23288
  Role: Jg-Tooling member

  Description/Motivation:

  Some specification documents contain tables which include trace items. E.g. SWS_Com, table in 7.12.1 Development Errors, SWS_Rte, Table 5.4: RTE Error and Status values.

  As discussed in tooling session we should extend ValidateARXML routine to elicit trace items inside tables.

Further on we shall ensure that tracebles do not contain figures and tables (77206, 74860)
–Last change on issue 77161 comment 3–

Agreed solution:

1. Affected Documents
==========

1. Move the Traceable out of the f**ing tables (see attachment)
2. move Tables and figures out of the Traceable

2. Word2arxml and latex2arxml which is used by checkDocumentSource
==========

let these scripts complain also requested by 77206, 74860 but summarized here

* Tables in Traceable
* Traceable in Tables
* Figures in Traceable

2.1 GST: add these contraints
==========

3. CP_Tool_Scripts
==========

* let latexinstatiaator complain about Figures in Traceable
* note: tha table in traceable violates the schema and is flagged already
* let latexinstantiator complain about traceable in tables
–Last change on issue 77161 comment 7–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.17 Specification Item SWS_Rte_01339

Trace References:

SRS_Rte_00094

Content:

|RTE_E_COM_STOPPED| – the RTE could not perform the operation because the COM communication service is currently not available (inter ECU communication only). RTE shall return |RTE_E_COM_STOPPED| whenthe corresponding COM :

- in case of COM the corresponding service returns |COM_SERVICE_NOT_AVAILABLE| .
- in case of LdCom the corresponding LdCom_Transmit returns |E_NOT_OK|

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76094: [Rte][LdCom] Reaction on E_NOT_OK when calling LdCom_Transmit

  Problem description:

  It is specified, that Rte_Write and Rte_Send shall return RTE_E_OK when
  "data passed to communication service _successfully_"
  But what shall happen if LdCom_Transmit returns E_NOT_OK ?

  Agreed solution:

  On behalf of WP-A1:

  Rephrase
  "[SWS_Rte_07822] RTE_E_COM_STOPPED the RTE could
  not perform the operation because the communication service is currently
  not available (inter ECU communication only). RTE shall return
  RTE_E_COM_STOPPED when:
  -in case of COM the corresponding COM service returns COM_SERVICE_NOT_AVAILABLE
  -in case of LdCom the corresponding LdCom_Transmit returns E_NOT_OK
  ."

  Do the same for SWS_Rte_07823, SWS_Rte_01106, SWS_Rte_06828 and SWS_Rte_01339

  SWS_Rte_01379: replace RTE_E_OK with E_OK.
  –Last change on issue 76094 comment 4–

  BW-C-Level:

Application	Specification	Bus
1	1	1

1.18 Specification Item SWS_Rte_01379

Trace References:

SRS_Rte_00246

Content:

The RTE shall lock the signal buffer after it initiated a Tp Transmission (LdCom_Transmit returned |RTE_E_OK|).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76094: [Rte][LdCom] Reaction on E_NOT_OK when calling LdCom_Transmit

  Problem description:

  It is specified, that Rte_Write and Rte_Send shall return RTE_E_OK when
  "data passed to communication service _successfully_"
  But what shall happen if LdCom_Transmit returns E_NOT_OK ?

  Agreed solution:

  On behalf of WP-A1:

  Rephrase
  "[SWS_Rte_07822] RTE_E_COM_STOPPED the RTE could
  not perform the operation because the communication service is currently
  not available (inter ECU communication only). RTE shall return
  RTE_E_COM_STOPPED when:
  -in case of COM the corresponding COM service returns COM_SERVICE_NOT_AVAILABLE
  -in case of LdCom the corresponding LdCom_Transmit returns E_NOT_OK
  ."

  Do the same for SWS_Rte_07823, SWS_Rte_01106, SWS_Rte_06828 and SWS_Rte_01339

  SWS_Rte_01379: replace RTE_E_OK with E_OK.
  –Last change on issue 76094 comment 4–

  BW-C-Level:

Application	Specification	Bus
1	1	1

1.19 Specification Item SWS_Rte_01389

Trace References:

SRS_Rte_00246

Content:

Symbolic name: |RTE_E_COM_BUSY|
Value: 141
Comments: The transmission/reception could not be performed due to another transmission/reception currently ongoing for the same signal.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77161: Tables in specification documents contain trace items

  Problem description:

  ———————————————
  Name: Wolf-Hendrik Kaps
  Phone: 0711-811-23288
  Role: Jg-Tooling member
  ———————————————
  Description/Motivation:

  Some specification documents contain tables which include trace items. E.g. SWS_Com, table in 7.12.1 Development Errors, SWS_Rte, Table 5.4: RTE Error and Status values.

  As discussed in tooling session we should extend ValidateARXML routine to elicit trace items inside tables.

  Further on we shall ensure that tracebles do not contain figures and tables (77206, 74860)
  –Last change on issue 77161 comment 3–

  Agreed solution:

  1. Affected Documents
  ==========

1. Move the Traceable out of the f**ing tables (see attachment)
2. move Tables and figures out of the Traceable

2. Word2arxml and latex2arxml which is used by checkDocumentSource
==========

let these scripts complain also requested by 77206, 74860 but summarized here

* Tables in Traceable
* Traceable in Tables
* Figures in Traceable

2.1 GST: add these contraints
==========

3. CP_Tool_Scripts
==========

* let latexinstatiaator complain about Figures in Traceable
* note: tha table in traceable violates the schema and is flagged already
* let latexinstantiator complain about traceable in tables
–Last change on issue 77161 comment 7–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.20 Specification Item SWS_Rte_02310

Trace References:

SRS_BSW_00305, SRS_Rte_00011, SRS_Rte_00167

Content:

The Application Header File shall include a structure declaration for the component data structure as follows:

```
struct Rte_[Byps_]CDS_<cts> { <component data sections> };
```

where |<cts>| is the component type symbol of the AtomicSwComponentType. |[Byps_]| is an optional infix used when component wrapper method for bypass support is enabled for the related software componenttype (See chapter [REF rte_fs_component_wrapper_method]).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76346: Component data structure type definition vs. structure declaration

  Problem description:

  In general the RTE SWS distinguishes between a structure declaration and a type declaration for a composite datatype that represents a structure.

  For a structure implementation data type SWS_Rte_07114 requires a structure declaration:

  struct _<name>_
  ...
  ;

  SWS_Rte_06812 requires the type declaration:

  typedef struct _<name>_ <name>;

  Regarding the component data structure currently there exist the following requirements only:

  [SWS_Rte_07132] The component data structure type shall be defined in the Application Header file.
  [SWS_Rte_03714] The type name of the component data structure shall be Rte_[Byps]_CDS_<cts> where <cts> is the component type symbol of the AtomicSwComponentType. ...

  Since SWS_Rte_03714 talks about the type name of the component data structure this requires a typedef statement for Rte_[Byps]_CDS_<cts> in the Application Header file:

  typedef struct ... Rte_[Byps]_CDS_<cts>;

  Currently there does not exist any requirement for a CDS structure declaration.

  The code examples in the SWS RTE are somehow in contradiction to these

requirements. They use "struct

Rte_CDS_<cts>" instead of the typedef name. See examples 5.24, 5.26, 5.28.

This inconsistency applies to the signature of some APIs, too:

[SWS_Rte_01238]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Start([const struct Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01239]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Return([const struct Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01248]
void Rte_[<client>_]Runnable_<cts>_<reName>_Start([const RTE_CDS_<cts>*])

[SWS_Rte_01249]
void Rte_[<client>_]Runnable_<cts>_<reName>_Return([const Rte_CDS_<cts>*])

[SWS_Rte_06113]
void Rte_[<client>_]RptHook_<cts>_<var>_Transmit([const RTE_CDS_<cts>*], <type>* <buffer>)

[SWS_Rte_06114]
void Rte_[<client>_]RptHook_<cts>_<var>_Reception([const RTE_CDS_<cts>*], <type>* <buffer>)

In addition some of these APIs use RTE_CDS_ instead of Rte_CDS_.

Agreed solution:

Delete SWS_Rte_07132 and SWS_Rte_03714.

Add new requirements:

[SWS_Rte_xxxx1] The Application Header File shall include a structure declaration for the component data structure as follows:
struct Rte_[Byps_]CDS_<cts> <component data sections> ;
where <cts> is the component type symbol of the AtomicSwComponentType. [Byps_] is an optional infix used when component wrapper method for bypass support is enabled for the related software componenttype (See chapter 4.9.2).

[SWS_Rte_xxxx2] The Application Header File shall include a type declaration

for the component data structure type as follows:
typedef struct Rte_[Byps_]CDS_<cts> Rte_[Byps_]CDS_<cts>;
where <cts> is the component type symbol of the AtomicSwComponentType.[Byps_] is an optional infix used when component wrapper method for bypass support is enabled for the related software componenttype (See chapter 4.9.2).

Change the API signatures to:

[SWS_Rte_01238]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Start([const Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01239]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Return([const Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01248]
void Rte_[<client>_]Runnable_<cts>_<reName>_Start([const Rte_CDS_<cts>*])

[SWS_Rte_06113]
void Rte_[<client>_]RptHook_<cts>_<var>_Transmit([const Rte_CDS_<cts>*], <type>* <buffer>)

[SWS_Rte_06114]
void Rte_[<client>_]RptHook_<cts>_<var>_Reception([const Rte_CDS_<cts>*], <type>* <buffer>)

Change examples 5.24, 5.26, 5.28 so that they contain both structure declaration and typedef for component data structure type and use the type (without struct) at the interfaces.
–Last change on issue 76346 comment 1–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.21 Specification Item SWS_Rte_02311

Trace References:

SRS_BSW_00305, SRS_Rte_00011, SRS_Rte_00167

Content:

The Application Header File shall include a type declaration for the component data structure type as follows:

```
typedef struct Rte_[Byps_]CDS_<cts> Rte_[Byps_]CDS_<cts>;
```

where |<cts>| is the component type symbol of the AtomicSwComponentType.|[Byps_]| is an optional infix used when component wrapper method for bypass support is enabled for the related software componenttype (See chapter [REF rte_fs_component_wrapper_method]).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76346: Component data structure type definition vs. structure declaration

  Problem description:

  In general the RTE SWS distinguishes between a structure declaration and a type declaration for a composite datatype that represents a structure.

  For a structure implementation data type SWS_Rte_07114 requires a structure declaration:

  struct _<name>_
  ...
  ;

  SWS_Rte_06812 requires the type declaration:

  typedef struct _<name>_ <name>;

  Regarding the component data structure currently there exist the following requirements only:

  [SWS_Rte_07132] The component data structure type shall be defined in the Application Header file.
  [SWS_Rte_03714] The type name of the component data structure shall be Rte_[Byps]_CDS_<cts> where <cts> is the component type symbol of the AtomicSwComponentType. ...

  Since SWS_Rte_03714 talks about the type name of the component data structure this requires a typedef statement for Rte_[Byps]_CDS_<cts> in the Application Header file:

typedef struct ... Rte_[Byps]_CDS_<cts>;

Currently there does not exist any requirement for a CDS structure declaration.

The code examples in the SWS RTE are somehow in contradiction to these requirements. They use "struct

Rte_CDS_<cts>" instead of the typedef name. See examples 5.24, 5.26, 5.28.

This inconsistency applies to the signature of some APIs, too:

[SWS_Rte_01238]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Start([const struct Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01239]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Return([const struct Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01248]
void Rte_[<client>_]Runnable_<cts>_<reName>_Start([const RTE_CDS_<cts>*])

[SWS_Rte_01249]
void Rte_[<client>_]Runnable_<cts>_<reName>_Return([const Rte_CDS_<cts>*])

[SWS_Rte_06113]
void Rte_[<client>_]RptHook_<cts>_<var>_Transmit([const RTE_CDS_<cts>*], <type>* <buffer>)

[SWS_Rte_06114]
void Rte_[<client>_]RptHook_<cts>_<var>_Reception([const RTE_CDS_<cts>*], <type>* <buffer>)

In addition some of these APIs use RTE_CDS_ instead of Rte_CDS_.

Agreed solution:

Delete SWS_Rte_07132 and SWS_Rte_03714.

Add new requirements:

[SWS_Rte_xxxx1] The Application Header File shall include a structure declaration for the component data structure as follows:

struct Rte_[Byps_]CDS_<cts> <component data sections> ;
where <cts> is the component type symbol of the AtomicSwComponentType. [Byps_] is an optional infix used when component wrapper method for bypass support is enabled for the related software componenttype (See chapter 4.9.2).

[SWS_Rte_xxxx2] The Application Header File shall include a type declaration for the component data structure type as follows:
typedef struct Rte_[Byps_]CDS_<cts> Rte_[Byps_]CDS_<cts>;
where <cts> is the component type symbol of the AtomicSwComponentType.[Byps_] is an optional infix used when component wrapper method for bypass support is enabled for the related software componenttype (See chapter 4.9.2).

Change the API signatures to:

[SWS_Rte_01238]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Start([const Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01239]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Return([const Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01248]
void Rte_[<client>_]Runnable_<cts>_<reName>_Start([const Rte_CDS_<cts>*])

[SWS_Rte_06113]
void Rte_[<client>_]RptHook_<cts>_<var>_Transmit([const Rte_CDS_<cts>*], <type>* <buffer>)

[SWS_Rte_06114]
void Rte_[<client>_]RptHook_<cts>_<var>_Reception([const Rte_CDS_<cts>*], <type>* <buffer>)

Change examples 5.24, 5.26, 5.28 so that they contain both structure declaration and typedef for component data structure type and use the type (without struct) at the interfaces.
–Last change on issue 76346 comment 1–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.22 Specification Item SWS_Rte_02571

Trace References:

SRS_BSW_00327, SRS_Rte_00107, SRS_Rte_00110, SRS_Rte_00094

Content:

Symbolic name: |RTE_E_LOST_DATA|
Value: 64
Comments: An API call for reading received data with event semantics indicates that some incoming data has been lost due to an overflow of the receive queue or due to an error of the underlying communication stack.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77161: Tables in specification documents contain trace items

  Problem description:

  _____
  Name: Wolf-Hendrik Kaps
  Phone: 0711-811-23288
  Role: Jg-Tooling member
  _____
  Description/Motivation:

  Some specification documents contain tables which include trace items. E.g. SWS_Com, table in 7.12.1 Development Errors, SWS_Rte, Table 5.4: RTE Error and Status values.

  As discussed in tooling session we should extend ValidateARXML routine to elicit trace items inside tables.

  Further on we shall ensure that tracebles do not contain figures and tables (77206, 74860)
  –Last change on issue 77161 comment 3–

  Agreed solution:

  1. Affected Documents
  ==========

  1. Move the Traceable out of the f**ing tables (see attachment)
  2. move Tables and figures out of the Traceable

2. Word2arxml and latex2arxml which is used by checkDocumentSource
==========

let these scripts complain also requested by 77206, 74860 but summarized here

* Tables in Traceable
* Traceable in Tables
* Figures in Traceable

2.1 GST: add these contraints
==========

3. CP_Tool_Scripts
==========

* let latexinstatiaator complain about Figures in Traceable
* note: tha table in traceable violates the schema and is flagged already
* let latexinstantiator complain about traceable in tables
–Last change on issue 77161 comment 7–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.23 Specification Item SWS_Rte_02579

Trace References:

SRS_Rte_00029, SRS_Rte_00082, SRS_Rte_00018

Content:

The RTE Generator shall reject configurations where there is inter-ECU client-server communication from several client-ECUs using the same SystemSignals and no MetaData is configured for distinction.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #69594: [SWS_Rte_05111] is not detectable by the RTE generator

Problem description:

I believe that requirement [SWS_Rte_05111] is not detectable by the RTE generator.
My interpretation of the requirement is that the RTE shall reject n:1 communication from one ECU to a server unless a client identifier is used.
On client-ECU it is not possible to detect if the Clients are communicating with the same server.
On server-ECU it is not possible to detect which Client-ECU is the origin of the client.

Agreed solution:

Remove SWS_Rte_05111.
Remove SWS_Rte_02579.
Add a reference to the constraint 3264 at the end of the preceding paragraph.
"...if the client identifier is used to distinguish the different clients (see constr_3264)."
–Last change on issue 69594 comment 16–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.24 Specification Item SWS_Rte_02594

Trace References:

SRS_BSW_00327, SRS_Rte_00078

Content:

Symbolic name: |RTE_E_INVALID|
Value: 1
Comments: Generic application error indicated by signal invalidation in sender receiver communication with data semantics on the receiver side.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77161: Tables in specification documents contain trace items

  Problem description:

  Name: Wolf-Hendrik Kaps
  Phone: 0711-811-23288
  Role: Jg-Tooling member

Description/Motivation:

Some specification documents contain tables which include trace items. E.g. SWS_Com, table in 7.12.1 Development Errors, SWS_Rte, Table 5.4: RTE Error and Status values.

As discussed in tooling session we should extend ValidateARXML routine to elicit trace items inside tables.

Further on we shall ensure that tracebles do not contain figures and tables (77206, 74860)
–Last change on issue 77161 comment 3–

Agreed solution:

1. Affected Documents
==========

1. Move the Traceable out of the f**ing tables (see attachment)
2. move Tables and figures out of the Traceable

2. Word2arxml and latex2arxml which is used by checkDocumentSource
==========

let these scripts complain also requested by 77206, 74860 but summarized here

* Tables in Traceable
* Traceable in Tables
* Figures in Traceable

2.1 GST: add these contraints
==========

3. CP_Tool_Scripts
==========

* let latexinstatiaator complain about Figures in Traceable
* note: tha table in traceable violates the schema and is flagged already
* let latexinstantiator complain about traceable in tables
–Last change on issue 77161 comment 7–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.25 Specification Item SWS_Rte_02702

Trace References:

SRS_BSW_00327, SRS_Rte_00078

Content:

Symbolic name: |RTE_E_MAX_AGE_EXCEEDED|
Value: 64
Comments: An API call for reading received data with data semantics indicates that the available data has exceeded the aliveTimeout limit. A COM signal outdated callback will result in this error.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77161: Tables in specification documents contain trace items

  Problem description:

  Name: Wolf-Hendrik Kaps
  Phone: 0711-811-23288
  Role: Jg-Tooling member

  Description/Motivation:

  Some specification documents contain tables which include trace items. E.g. SWS_Com, table in 7.12.1 Development Errors, SWS_Rte, Table 5.4: RTE Error and Status values.

  As discussed in tooling session we should extend ValidateARXML routine to elicit trace items inside tables.

  Further on we shall ensure that tracebles do not contain figures and tables (77206, 74860)
  –Last change on issue 77161 comment 3–

  Agreed solution:

1. Affected Documents
==========

1. Move the Traceable out of the f**ing tables (see attachment)
2. move Tables and figures out of the Traceable

2. Word2arxml and latex2arxml which is used by checkDocumentSource
==========

let these scripts complain also requested by 77206, 74860 but summarized here

* Tables in Traceable
* Traceable in Tables
* Figures in Traceable

2.1 GST: add these contraints
==========

3. CP_Tool_Scripts
==========

* let latexinstatiaator complain about Figures in Traceable
* note: tha table in traceable violates the schema and is flagged already
* let latexinstantiator complain about traceable in tables
–Last change on issue 77161 comment 7–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.26 Specification Item SWS_Rte_02706

Trace References:

SRS_Rte_00143, SRS_Rte_00018

Content:

RTE shall reject configurations that contain OperationInvokedEvents with a mode disabling dependency. the configurations violating constr_1523.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76989: Missing constraints for mode disabling dependencies and OperationInvokedEvents / BswOperationInvokedEvents

  Problem description:

  The SWS RTE excludes
  the combination of mode disabling dependencies and OperationInvokedEvents ([SWS_Rte_02706])

  Reading the rational, the same combination shall be excluded for BswOperationInvokedEvents

  Agreed solution:

  SWC-T
  [constr_xxxx] No mode disabling for OperationInvokedEvents d An OperationInvokedEvent shall not have a reference to a ModeDeclaration in the role disabledMode. c()

  Rationale
  The RTE does not support the disabling of server Runnables by modes. Instead, the server shall respond with an explicit error code if the execution of the server operation is not possible in specific side conditions.

  BSWMD-T
  [constr_yyyy] No mode disabling for BswOperationInvokedEvents d An BswOperationInvokedEvents shall not have a reference to a ModeDeclaration in the role disabledInMode. c()

  SWS RTE:
  adjust
  [SWS_Rte_02706] The RTE shall reject the configurations violating constr_xxxx
  add
  [SWS_Rte_nnnn1] d The RTE shall reject the configurations violating constr_yyyy
  –Last change on issue 76989 comment 3–

  BW-C-Level:

Application	Specification	Bus
4	4	1

1.27 Specification Item SWS_Rte_02739

Trace References:

SRS_BSW_00327

Content:

Symbolic name: |RTE_E_IN_EXCLUSIVE_AREA|
Value: 135
Comments: The error is returned by a blocking API and indicates that the runnable could not enter a wait state. This could be for example because one ExecutableEntity of the current task's call stack has entered an ExclusiveArea.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77161: Tables in specification documents contain trace items

  Problem description:

  _____

  Name: Wolf-Hendrik Kaps
  Phone: 0711-811-23288
  Role: Jg-Tooling member

  _____

  Description/Motivation:

  Some specification documents contain tables which include trace items. E.g. SWS_Com, table in 7.12.1 Development Errors, SWS_Rte, Table 5.4: RTE Error and Status values.

  As discussed in tooling session we should extend ValidateARXML routine to elicit trace items inside tables.

  Further on we shall ensure that tracebles do not contain figures and tables (77206, 74860)
  –Last change on issue 77161 comment 3–

  Agreed solution:

  1. Affected Documents
  ==========

  1. Move the Traceable out of the f**ing tables (see attachment)
  2. move Tables and figures out of the Traceable

2. Word2arxml and latex2arxml which is used by checkDocumentSource
==========

let these scripts complain also requested by 77206, 74860 but summarized here

* Tables in Traceable
* Traceable in Tables
* Figures in Traceable

2.1 GST: add these contraints
==========

3. CP_Tool_Scripts
==========

* let latexinstatiaator complain about Figures in Traceable
* note: tha table in traceable violates the schema and is flagged already
* let latexinstantiator complain about traceable in tables
–Last change on issue 77161 comment 7–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.28 Specification Item SWS_Rte_02747

Trace References:

SRS_BSW_00327

Content:

Symbolic name: |SCHM_E_IN_EXCLUSIVE_AREA|
Value: 135
Comments: The error is returned by a blocking API and indicates that the schedulable entity could not enter a wait state, because one ExecutableEntity of the current task's call stack has entered an ExclusiveArea.

Note: There are no blocking SchM APIs and therefore this value cannot be returned. It is defined here for future use and for consistency with SWS_Rte_02739. Both error values have to be identical.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77161: Tables in specification documents contain trace items

  Problem description:

  _____
  Name: Wolf-Hendrik Kaps
  Phone: 0711-811-23288
  Role: Jg-Tooling member
  _____
  Description/Motivation:

  Some specification documents contain tables which include trace items. E.g. SWS_Com, table in 7.12.1 Development Errors, SWS_Rte, Table 5.4: RTE Error and Status values.

  As discussed in tooling session we should extend ValidateARXML routine to elicit trace items inside tables.

  Further on we shall ensure that tracebles do not contain figures and tables (77206, 74860)
  –Last change on issue 77161 comment 3–

  Agreed solution:

  1. Affected Documents
  ==========

  1. Move the Traceable out of the f**ing tables (see attachment)
  2. move Tables and figures out of the Traceable

  2. Word2arxml and latex2arxml which is used by checkDocumentSource
  ==========

  let these scripts complain also requested by 77206, 74860 but summarized here

  * Tables in Traceable

* Traceable in Tables
* Figures in Traceable

2.1 GST: add these contraints
==========

3. CP_Tool_Scripts
==========

* let latexinstatiaator complain about Figures in Traceable
* note: tha table in traceable violates the schema and is flagged already
* let latexinstantiator complain about traceable in tables
–Last change on issue 77161 comment 7–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.29 Specification Item SWS_Rte_02757

Trace References:

SRS_BSW_00327

Content:

Symbolic name: |RTE_E_SEG_FAULT|
Value: 136
Comments: The error can be returned by an RTE API, if the parameters contain a direct or indirect reference to memory that is not accessible from the callers partition.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77161: Tables in specification documents contain trace items

  Problem description:

  Name: Wolf-Hendrik Kaps
  Phone: 0711-811-23288
  Role: Jg-Tooling member

  Description/Motivation:

Some specification documents contain tables which include trace items. E.g. SWS_Com, table in 7.12.1 Development Errors, SWS_Rte, Table 5.4: RTE Error and Status values.

As discussed in tooling session we should extend ValidateARXML routine to elicit trace items inside tables.

Further on we shall ensure that tracebles do not contain figures and tables (77206, 74860)
–Last change on issue 77161 comment 3–

Agreed solution:

1. Affected Documents
==========

1. Move the Traceable out of the f**ing tables (see attachment)
2. move Tables and figures out of the Traceable

2. Word2arxml and latex2arxml which is used by checkDocumentSource
==========

let these scripts complain also requested by 77206, 74860 but summarized here

* Tables in Traceable
* Traceable in Tables
* Figures in Traceable

2.1 GST: add these contraints
==========

3. CP_Tool_Scripts
==========

* let latexinstatiaator complain about Figures in Traceable
* note: tha table in traceable violates the schema and is flagged already
* let latexinstantiator complain about traceable in tables
–Last change on issue 77161 comment 7–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.30 Specification Item SWS_Rte_03608

Trace References:

SRS_Rte_00249

Content:

If there is a PortAPIOption with the attribute PortAPIOption.errorHandling set to DataTransformationErrorHandlingEnum.transformerErrorHandling referencing a PortPrototype to which no data transformation applies, the |Rte_TransformerClass| shall be set to |RTE_TRANSFORMER_UNSPECIFIED| and |Rte_TransformerErrorCode| to |E_OK|.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77115: transformerError parameter when no dataTransformation applies

  Problem description:

  According to RTE SWS the The optional OUT parameter
  transformerError of the APIs (Rte_Write, Rte_IStatus, Rte_Send, Rte_Receive, Rte_Call, Rte_Result, etc.) shall be generated if the PortPrototype of port <p> is referenced by a PortAPIOption
  which has the attribute errorHandling set to
  transformerErrorHandling.

  Furthermore chapter "4.10.5 Error Handling" clarifies which errors shall be returned by the RTE when transformers are executed.
  But IMHO it shall also be clarified and specified which error shall be returned by the RTE when the PortAPIOption
  which has the attribute errorHandling set to
  "transformerErrorHandling" and there is no data transformation - i.e. no execution of transformers - no for this particular port.

  This is to ensure the portability of SWCs without redesigning them.

  Agreed solution:

  add below requirement in chapter 4.10.5 between [SWS_Rte_08539] and [SWS_Rte_08540]
  [SWS_Rte_0xxxx]If there is a PortAPIOption with the attribute errorHandling set to transformerErrorHandling referencing a PortPrototype to which

no data transformation applies the Rte_TransformerClass shall be set to RTE_TRANSFORMER_UNSPECIFIED and Rte_TransformerErrorCode to E_OK.

Rationale: the generation condition of the optional OUT parameter transformerError only depends on the attribute errorHandling. Nevertheless it is possible to integrate such SW-Cs supporting transformerErrorHandling without any transformers. And in this case the data transformation is always logically assumed to be successful.
–Last change on issue 77115 comment 4–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.31 Specification Item SWS_Rte_03609

Trace References:

SRS_Rte_00055, SRS_Rte_00164

Content:

For each Array Implementation Data Type which last ImplementationDataTypeElement is of category STRUCTURE, the RTE Types Header File shall include the corresponding type declaration as:

|typedef| |struct| | <elements> | |<name>|;

where |<elements>| is the record element specification and

|<name>| is the Implementation Data Type Element shortName of the Array Implementation Data Type.

For each record element defined by one ImplementationDataTypeElement one record element specification |<elements>| is defined. The record element specifications are ordered according the order of the related ImplementationDataTypeElements in the input configuration. Sequent record elements are separated with a semicolon.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77453: "[SWS_Rte_07072]" and "[SWS_Rte_06706]/[SWS_Rte_06707]" lead to uncompilable code

  Problem description:

According to [SWS_Rte_06706] and [SWS_Rte_06707] a typedef shall be generated for each ArrayImplementationDataType which last ImplementationDataTypeElement is of category STRUCTURE,UNION.

[SWS_Rte_06706] d For each Array Implementation Data Type which last ImplementationDataTypeElement is of category STRUCTURE, the RTE Types Header File shall include the corresponding type declaration as:
typedef struct <elements> <name>[<size 1>][<size 2>]...
[<size n>];
where <elements> is the record element specification and
<name> is the Implementation Data Type symbol of the Array Implementation Data Type.
...
...

Assuming an ImplementationDataType TypeArrayOfStruct of category ARRAY with its ImplementationDataTypeElement ArrayOfStruct_ElementType of category STRUCTURE with an ArraySize of 2. Furthermore this ImplementationDataTypeElement defines additional ImplementationDataTypeElements element1_boolean, element2_sint32 of category TYPE_REFERENCE.

From this definition following should be generated in Rte_Type.h, which is IMHO correct according to SWS_Rte_06706

typedef struct
boolean element1_boolean;
sint32 element2_sint32;
TypeArrayOfStruct[2];

Let's consider a DataPrototype "myArPimOfTypeArrayOfStruct" - defined e.g. in role of ArTypedPerInstanceMemory - associated to this ImplementationDataType "TypeArrayOfStruct".

This will lead to following generation of the RTE API in the Application header file.

define Rte_Pim_myArPimOfTypeArrayOfStruct() ((P2VAR(struct
boolean element1_boolean;
sint32 element2_sint32;
, AUTOMATIC, RTE_APPL_DATA)) & Rte_ArPim_rba_Swc1_myArPimOfTypeArrayOfStruct
)

According to "[SWS_Rte_07072]" which states "If the DataPrototype is associ-

ated to an Array Implementation
Data Type the RTE API shall return an array expression (that is a pointer to the array base type) pointing to variable holding the value of the DataPrototype for which the API provides access.
..." the generated code is correct, but leads to compiler errors.

IMHO following requirements need to be updated [SWS_Rte_06706]/[SWS_Rte_06707] to enable the generation of the typedef in case the ImplementationDataTypeElement is of category STRUCTURE/UNION. Furthermore [SWS_Rte_07072] needs to be updated and it shall be clearly stated that the "array base type" shall be the ImplementationDataTypeElement.ShortName if it is of category STRUCTURE/UNION.

Agreed solution:

1.) add new requirement [SWS_Rte_0xxx0] before [SWS_Rte_06706]
[SWS_Rte_0xxx0] d For each Array Implementation Data Type which last ImplementationDataTypeElement is of
category STRUCTURE, the RTE Types Header File shall include the corresponding type declaration as:
typedef struct <elements> <name>; c(SRS_Rte_00055, SRS_Rte_00164)

where <elements> is the record element specification and
<name> is the Implementation Data Type Element shortName of the Array Implementation Data Type.

For each record element defined by one ImplementationDataTypeElement one record element specification <elements> is defined. The record element specifications
are ordered according the order of the related ImplementationDataTypeElements in the input configuration.
Sequent record elements are separated with a semicolon.
The definition of the record element specification is defined in section 5.3.4.6.

2.) update [SWS_Rte_06706]
[SWS_Rte_06706] d For each Array Implementation Data Type which last ImplementationDataTypeElement is of
category STRUCTURE, the RTE Types Header File shall include the corresponding type declaration as:

typedef <type> <name>[<size 1>][<size 2>]...[<size n>];

where <type> is the Implementation Data Type Element shortName,

<name> is the Implementation Data Type symbol of the Array Implementation Data Type,
[<size x>] is the arraySize of the Arrays ImplementationDataTypeElement.
For each array dimension defined by one Arrays ImplementationDataTypeElement one array dimension definition [<size x>] is defined.
The array dimension definitions [<size 1>], [<size 2>] ... [<size n>] ordered from the root to the last ImplementationDataTypeElement belonging to the array definition. c(SRS_Rte_00055, SRS_Rte_00164)

3.) add new requirement [SWS_Rte_0xxx1] before [SWS_Rte_06707]
[SWS_Rte_0xxx1] d For each Array Implementation Data Type which last ImplementationDataTypeElement is of
category UNION, the RTE Types Header File shall include the corresponding type declaration as:
typedef union <elements> <name>;

where <elements> is the union element specification and
<name> is the Implementation Data Type Element shortName of the Array Implementation Data Type.

For each union element defined by one ImplementationDataTypeElement one union element specification <elements> is defined. The union element specifications
are ordered according the order of the related ImplementationDataTypeElements in the input configuration.
Sequent union elements are separated with a semicolon.
The definition of the union element specification is defined in section 5.3.4.6.

4.) update [SWS_Rte_06707]
[SWS_Rte_06707] d For each Array Implementation Data Type which last ImplementationDataTypeElement is of
category UNION, the RTE Types Header File shall include the corresponding type declaration as:

typedef <type> <name>[<size 1>][<size 2>]...[<size n>];

where <type> is the Implementation Data Type Element shortName,
<name> is the Implementation Data Type symbol of the Array Implementation Data Type,
[<size x>] is the arraySize of the Arrays ImplementationDataTypeElement.
For each array dimension defined by one Arrays ImplementationDataTypeElement one array dimension definition [<size x>] is defined.
The array dimension definitions [<size 1>], [<size 2>] ... [<size n>] ordered

from the root to the last ImplementationDataTypeElement belonging to the array definition. c(SRS_Rte_00055, SRS_Rte_00164)

5.) update [SWS_Rte_07072]
[SWS_Rte_07072] d If the DataPrototype is associated to an Array Implementation Data Type the RTE API shall return an array expression (that is a pointer
to the array base type) pointing to variable holding the value of the DataPrototype for which the API provides access. If the leaf ImplementationDataTypeElement
is typed by a SwBaseType the array type name shall be equal to the nativeDeclaration
attribute of the SwBaseType. If the leaf ImplementationDataTypeElement is typed by an
ImplementationDataType the type name shall be equal to the shortName of this ImplementationDataType.
If the leaf ImplementationDataTypeElement is of category STRUCTURE or UNION the type name shall be equal to
the shortName of this ImplementationDataTypeElement. c(SRS_Rte_00059)
–Last change on issue 77453 comment 2–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.32 Specification Item SWS_Rte_03610

Trace References:

SRS_Rte_00055, SRS_Rte_00164

Content:

For each Array Implementation Data Type which last ImplementationDataTypeElement is of category UNION, the RTE Types Header File shall include the corresponding type declaration as:

|typedef| |union| | <elements> | |<name>|;
where |<elements>| is the union element specification and |<name>| is the Implementation Data Type Element shortName of the Array Implementation Data Type.

For each union element defined by one ImplementationDataTypeElement one union element specification |<elements>| is defined. The union element specifications are ordered according the order of the related ImplementationDataTypeElements in the input configuration. Sequent union elements are separated with a semicolon.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77453: "[SWS_Rte_07072]" and "[SWS_Rte_06706]/[SWS_Rte_06707]" lead to uncompilable code

  Problem description:

  According to [SWS_Rte_06706] and [SWS_Rte_06707] a typedef shall be generated for each ArrayImplementationDataType which last ImplementationDataTypeElement is of category STRUCTURE,UNION.

  [SWS_Rte_06706] d For each Array Implementation Data Type which last ImplementationDataTypeElement is of category STRUCTURE, the RTE Types Header File shall include the corresponding type declaration as:
  typedef struct <elements> <name>[<size 1>][<size 2>]...
  [<size n>];
  where <elements> is the record element specification and
  <name> is the Implementation Data Type symbol of the Array Implementation Data Type.
  ...
  ...

  Assuming an ImplementationDataType TypeArrayOfStruct of category ARRAY with its ImplementationDataTypeElement ArrayOfStruct_ElementType of category STRUCTURE with an ArraySize of 2. Furthermore this ImplementationDataTypeElement defines additional ImplementationDataTypeElements element1_boolean, element2_sint32 of category TYPE_REFERENCE.

  From this definition following should be generated in Rte_Type.h, which is IMHO correct according to SWS_Rte_06706

  typedef struct
  boolean element1_boolean;
  sint32 element2_sint32;
  TypeArrayOfStruct[2];

  Let's consider a DataPrototype "myArPimOfTypeArrayOfStruct" - defined e.g. in role of ArTypedPerInstanceMemory - associated to this ImplementationDataType "TypeArrayOfStruct".

  This will lead to following generation of the RTE API in the Application header file.

  # define Rte_Pim_myArPimOfTypeArrayOfStruct() ((P2VAR(struct

boolean element1_boolean;
sint32 element2_sint32;
, AUTOMATIC, RTE_APPL_DATA)) & Rte_ArPim_rba_Swc1_myArPimOfTypeArrayOfStruct
)

According to "[SWS_Rte_07072]" which states "If the DataPrototype is associated to an Array Implementation
Data Type the RTE API shall return an array expression (that is a pointer to the array base type) pointing to variable holding the value of the DataPrototype for which the API provides access.
..." the generated code is correct, but leads to compiler errors.

IMHO following requirements need to be updated [SWS_Rte_06706]/[SWS_Rte_06707] to enable the generation of the typedef in case the ImplementationDataTypeElement is of category STRUCTURE/UNION. Furthermore [SWS_Rte_07072] needs to be updated and it shall be clearly stated that the "array base type" shall be the ImplementationDataTypeElement.ShortName if it is of category STRUCTURE/UNION.

Agreed solution:

1.) add new requirement [SWS_Rte_0xxx0] before [SWS_Rte_06706]
[SWS_Rte_0xxx0] d For each Array Implementation Data Type which last ImplementationDataTypeElement is of
category STRUCTURE, the RTE Types Header File shall include the corresponding type declaration as:
typedef struct <elements> <name>; c(SRS_Rte_00055, SRS_Rte_00164)

where <elements> is the record element specification and
<name> is the Implementation Data Type Element shortName of the Array Implementation Data Type.

For each record element defined by one ImplementationDataTypeElement one record element specification <elements> is defined. The record element specifications
are ordered according the order of the related ImplementationDataTypeElements in the input configuration.
Sequent record elements are separated with a semicolon.
The definition of the record element specification is defined in section 5.3.4.6.

2.) update [SWS_Rte_06706]
[SWS_Rte_06706] d For each Array Implementation Data Type which last ImplementationDataTypeElement is of

category STRUCTURE, the RTE Types Header File shall include the corresponding type declaration as:

typedef <type> <name>[<size 1>][<size 2>]...[<size n>];

where <type> is the Implementation Data Type Element shortName,
<name> is the Implementation Data Type symbol of the Array Implementation Data Type,
[<size x>] is the arraySize of the Arrays ImplementationDataTypeElement.
For each array dimension defined by one Arrays ImplementationDataTypeElement one array dimension definition [<size x>] is defined.
The array dimension definitions [<size 1>], [<size 2>] ... [<size n>] ordered from the root to the last ImplementationDataTypeElement belonging to the array definition. c(SRS_Rte_00055, SRS_Rte_00164)

3.) add new requirement [SWS_Rte_0xxx1] before [SWS_Rte_06707]
[SWS_Rte_0xxx1] d For each Array Implementation Data Type which last ImplementationDataTypeElement is of
category UNION, the RTE Types Header File shall include the corresponding type declaration as:
typedef union <elements> <name>;

where <elements> is the union element specification and
<name> is the Implementation Data Type Element shortName of the Array Implementation Data Type.

For each union element defined by one ImplementationDataTypeElement one union element specification <elements> is defined. The union element specifications
are ordered according the order of the related ImplementationDataTypeElements in the input configuration.
Sequent union elements are separated with a semicolon.
The definition of the union element specification is defined in section 5.3.4.6.

4.) update [SWS_Rte_06707]
[SWS_Rte_06707] d For each Array Implementation Data Type which last ImplementationDataTypeElement is of
category UNION, the RTE Types Header File shall include the corresponding type declaration as:

typedef <type> <name>[<size 1>][<size 2>]...[<size n>];

where <type> is the Implementation Data Type Element shortName,

<name> is the Implementation Data Type symbol of the Array Implementation Data Type,
[<size x>] is the arraySize of the Arrays ImplementationDataTypeElement.
For each array dimension defined by one Arrays ImplementationDataTypeElement one array dimension definition [<size x>] is defined.
The array dimension definitions [<size 1>], [<size 2>] ... [<size n>] ordered from the root to the last ImplementationDataTypeElement belonging to the array definition. c(SRS_Rte_00055, SRS_Rte_00164)

5.) update [SWS_Rte_07072]
[SWS_Rte_07072] d If the DataPrototype is associated to an Array Implementation Data Type the RTE API shall return an array expression (that is a pointer to the array base type) pointing to variable holding the value of the DataPrototype for which the API provides access. If the leaf ImplementationDataTypeElement is typed by a SwBaseType the array type name shall be equal to the nativeDeclaration attribute of the SwBaseType. If the leaf ImplementationDataTypeElement is typed by an ImplementationDataType the type name shall be equal to the shortName of this ImplementationDataType.
If the leaf ImplementationDataTypeElement is of category STRUCTURE or UNION the type name shall be equal to the shortName of this ImplementationDataTypeElement. c(SRS_Rte_00059)
–Last change on issue 77453 comment 2–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.33 Specification Item SWS_Rte_03714

Trace References:

SRS_BSW_00305

Content:

The type name of the component data structure shall be |Rte_[Byps]_CDS_<cts>| where |<cts>| is the component type symbol of the AtomicSwComponentType. |[Byps_]| is an optional infix used when component wrapper method for bypass support is enabled for the related software component type (See chapter REF rte_fs_component_wrapper_method).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76346: Component data structure type definition vs. structure declaration

  Problem description:

  In general the RTE SWS distinguishes between a structure declaration and a type declaration for a composite datatype that represents a structure.

  For a structure implementation data type SWS_Rte_07114 requires a structure declaration:

  struct _<name>_
  ...
  ;

  SWS_Rte_06812 requires the type declaration:

  typedef struct _<name>_ <name>;

  Regarding the component data structure currently there exist the following requirements only:

  [SWS_Rte_07132] The component data structure type shall be defined in the Application Header file.
  [SWS_Rte_03714] The type name of the component data structure shall be Rte_[Byps]_CDS_<cts> where <cts> is the component type symbol of the AtomicSwComponentType. ...

  Since SWS_Rte_03714 talks about the type name of the component data structure this requires a typedef statement for Rte_[Byps]_CDS_<cts> in the Application Header file:

  typedef struct ... Rte_[Byps]_CDS_<cts>;

  Currently there does not exist any requirement for a CDS structure declaration.

  The code examples in the SWS RTE are somehow in contradiction to these requirements. They use "struct

  Rte_CDS_<cts>" instead of the typedef name. See examples 5.24, 5.26, 5.28.

  This inconsistency applies to the signature of some APIs, too:

[SWS_Rte_01238]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Start([const struct Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01239]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Return([const struct Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01248]
void Rte_[<client>_]Runnable_<cts>_<reName>_Start([const RTE_CDS_<cts>*])

[SWS_Rte_01249]
void Rte_[<client>_]Runnable_<cts>_<reName>_Return([const Rte_CDS_<cts>*])

[SWS_Rte_06113]
void Rte_[<client>_]RptHook_<cts>_<var>_Transmit([const RTE_CDS_<cts>*], <type>* <buffer>)

[SWS_Rte_06114]
void Rte_[<client>_]RptHook_<cts>_<var>_Reception([const RTE_CDS_<cts>*], <type>* <buffer>)

In addition some of these APIs use RTE_CDS_ instead of Rte_CDS_.

Agreed solution:

Delete SWS_Rte_07132 and SWS_Rte_03714.

Add new requirements:

[SWS_Rte_xxxx1] The Application Header File shall include a structure declaration for the component data structure as follows:
struct Rte_[Byps_]CDS_<cts> <component data sections> ;
where <cts> is the component type symbol of the AtomicSwComponentType. [Byps_] is an optional infix used when component wrapper method for bypass support is enabled for the related software componenttype (See chapter 4.9.2).

[SWS_Rte_xxxx2] The Application Header File shall include a type declaration for the component data structure type as follows:
typedef struct Rte_[Byps_]CDS_<cts> Rte_[Byps_]CDS_<cts>;
where <cts> is the component type symbol of the AtomicSwComponentType.[Byps_] is an optional infix used when component wrapper method for bypass support is enabled for the related software componenttype (See chapter 4.9.2).

Change the API signatures to:

[SWS_Rte_01238]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Start([const Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01239]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Return([const Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01248]
void Rte_[<client>_]Runnable_<cts>_<reName>_Start([const Rte_CDS_<cts>*])

[SWS_Rte_06113]
void Rte_[<client>_]RptHook_<cts>_<var>_Transmit([const Rte_CDS_<cts>*], <type>* <buffer>)

[SWS_Rte_06114]
void Rte_[<client>_]RptHook_<cts>_<var>_Reception([const Rte_CDS_<cts>*], <type>* <buffer>)

Change examples 5.24, 5.26, 5.28 so that they contain both structure declaration and typedef for component data structure type and use the type (without struct) at the interfaces.
–Last change on issue 76346 comment 1–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.34 Specification Item SWS_Rte_03809

Trace References:

SRS_Rte_00167

Content:

The Application Types Header File shall include the definitions of all constants of ImplementationDataTypes and ApplicationDataTypes for each ImplementationDataType/ApplicationDataTypes used (referencedSee SWS_Rte_08802 for the meaning of the term "used") by this software component.

This includes constants for CompuMethods referenced by ImplementationDataTypeElements of ImplementationDataTypes directly referenced by the software component and constants for CompuMethods of ImplementationDataTypes which are referenced indirectly via ImplementationDataTypes / ImplementationDataTypeElements of category TYPE_REFERENCE.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #73034: Clarification required regarding applicability of SWS_Rte_03809 - should labels be generated for mapped ImplementationDataTypes?

  Problem description:

  SWS_Rte_03809 states that, for a given ApplicationSoftwareComponent, the constants associated with all 'used' AutosarDataTypes must be included in the Application Types Header. It is noted that AutosarDataTypes referenced from relevant DataPrototypes and AutosarDataTypes referenced from IncludedDataTypeSets are all 'used' in this sense and further that sub-element data types of 'used' complex data types are also 'used', as are data types that are aliased by 'used' type reference data types.

  What is not entirely clear is whether an ImplementationDataType that is mapped by the component to a 'used' ApplicationDataType is to be considered 'used' if it is not directly referenced from a DataPrototype or IncludedDataTypeSet itself.

  On the one hand, this case is not explicitly included, although it is not clear whether the list is intended to be exhaustive. Indeed it does not mention element types of 'used' complex ApplicationDataTypes although it would seem strange if they were not to be included analogously to the ImplementationDataType case that is explicitly included.

  On the other hand, the example given in the definition of 'AI' within Table 5.39 in the SoftwareComponentTemplate (AR4.2.2) suggests that the labels from both types may be used. Indeed, when a DataPrototype references an ApplicationDataType then the corresponding APIs use the mapped ImplementationDataType so it is reasonable to expect the software component code to use its labels, as well as the aliases from the ApplicationDataType. It could also be argued that the ImplementationDataType /is/ referenced by the component, via the referenced DataTypeMappingSet.

  Issue # 72232 has clarified that in the case of labels associated with both an ApplicationDataType and its mapped ImplementationDataType an IncludedDataTypeSet referencing only the ApplicationDataType gives its literalPrefix (by SWS_Rte_03810) to the definitions for the ApplicationDataType only but it is not

entirely clear whether the mapped ImplementationDataType should be considered ‘used’ at all in this case. Since there is no API or other code artefact resulting from an IncludedDataTypeSet it seems reasonable that different behaviour may be required in this respect compared to the case where the ApplicationDataType is referenced by a DataPrototype.

Having the labels defined for both types ensures that whichever labels the component code uses will be available but the downside is that conflicts between labels of different ImplementationDataTypes may occur and require entries in IncludedDataTypeSets to resolve even if the labels are not actually being used in the code. A mismatch of interpretations in either direction can be worked around using IncludedDataTypeSets but a common understanding is required to reduce this workload on integrators.

Agreed solution:

— SWS RTE —
[SWS_Rte_03809] d The Application Types Header File shall include the definitions of all constants of ImplementationDataTypes and ApplicationDataTypes for each ImplementationDataType/ApplicationDataTypes used (See [SWS_Rte_0xxx1] for the meaning of the term "used") by this software component.

[SWS_Rte_03983] d The The Module Interlink Types Header File shall include the definitions of all constants of ImplementationDataTypes and Application-DataTypes for each ImplementationDataType/ApplicationDataTypes used (See [SWS_Rte_0xxx2] for the meaning of the term "used") by this Basic Software module.

Add [SWS_Rte_0xxx1] in chapter 5.5.5

[SWS_Rte_0xxx1] The meaning of the term "used" with respect to Autosar-DataTypes [
An AutosarDataType is used if it meets any one of the following conditions:

- it is referenced by a DataPrototype in the InternalBehavior, or
- it is referenced by a VariationPointProxy in the InternalBehavior, or
- it is referenced by a DataPrototype in a PortInterface referenced by a PortPrototype, or
- it is referenced by an IncludedDataTypeSet in the InternalBehavior, or
- it is the ImplementationDataType mapped to an ApplicationDataType (i.e. via the DataTypeMappingSet) that is used in one of the above ways, or
- it is an ImplementationDataTypeElement of a complex ImplementationDataType that is used in one of the above ways, or
- it is referenced as the target type of an ImplementationDataType or Implementa-

tionDataTypeElement of category TYPE_REFERENCE that is used in one of the above ways, or
- it is an ApplicationDataType referenced as the type of a sub-element of a complex ApplicationDataType that is used in one of the above ways.
]
Please note that in contrast to the TYPE_REFERENCE case, when an ImplementationDataType of category DATA_REFERENCE is "used" the target ImplementationDataType it references is not considered used, unless it is independently used in its own right.
-
Add [SWS_Rte_0xxx2] in chapter 6.4.3
[SWS_Rte_0xxx2] The meaning of the term "used" with respect to AutosarDataTypes [
An AutosarDataType is used if it meets any one of the following conditions:

- it is referenced by a DataPrototype in the InternalBehavior, or
- it is referenced by a VariationPointProxy in the internalBehavior, or
- it is referenced by a DataPrototype referenced by a providedData or requiredData, or
- it is referenced by an IncludedDataTypeSet in the InternalBehavior, or
- it is the ImplementationDataType mapped to an ApplicationDataType (i.e. via the DataTypeMappingSet) that is used in one of the above ways, or
- it is an ImplementationDataTypeElement of a complex ImplementationDataType that is used in one of the above ways, or
- it is referenced as the target type of an ImplementationDataType or ImplementationDataTypeElement of category TYPE_REFERENCE that is used in one of the above ways, or
- it is an ApplicationDataType referenced as the type of a sub-element of a complex ApplicationDataType that is used in one of the above ways.
]
Please note that in contrast to the TYPE_REFERENCE case, when an ImplementationDataType of category DATA_REFERENCE is "used" the target ImplementationDataType it references is not considered used, unless it is independently used in its own right.
–Last change on issue 73034 comment 14–

BW-C-Level:

Application	Specification	Bus
3	3	1

1.35 Specification Item SWS_Rte_03812

Trace References:

SRS_Rte_00051, SRS_Rte_00032

Content:

Entries in the Exclusive-area API section shall be sorted alphabetically (ASCII / ISO 8859-1 code in ascending order).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76347: Explicit specification of alphabetical ordering

  Problem description:

  Requirements SWS_Rte_06061, SWS_Rte_03812, and SWS_Rte_07228 do not mention the exact criteria for alphabetical ordering.

  Agreed solution:

  Requirements SWS_Rte_06061, SWS_Rte_03812, and SWS_Rte_07228:
  Add the parenthesis phrase "(ASCII / ISO 8859-1 code in ascending order)" after "alphabetic ordering".
  –Last change on issue 76347 comment 1–

  BW-C-Level:

Application	Specification	Bus
1	4	1

1.36 Specification Item SWS_Rte_03853

Trace References:

SRS_Rte_00094, SRS_Rte_00210

Content:

|RTE_E_TIMEOUT| – Any mode user partition is stopped or restarting or has been restarted while the mode switch was requested.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #73463: Clarification regarding requirements for mode acknowledgement [SWS_Rte_03853][SWS_Rte_07055][SWS_Rte_02679]

Problem description:

The mentioned requirements assumes that there may only be mode-users in a single partition. Since 4.2.x multiple partitions could exist with mode-users. So these requirements need to be updated. I assume RTE_E_TIMEOUT should be returned when any partition with mode-users of the mode-machine is stopped or restarting as well as the RunnableEntity should be activated when any mode-user partition is stopped or restarted.

Propose to change the requirements [SWS_Rte_03853][SWS_Rte_07055][SWS_Rte_02679] according to the following:

[SWS_Rte_03853] RTE_E_TIMEOUT Any mode-user partition is stopped or restarting or has been restarted
while the mode switch was requested.

[SWS_Rte_07055] SCHM_E_TIMEOUT The configured timeout
exceeds before the mode transition was completed.
OR:
Any mode-user partition is stopped or restarting or
has been restarted while the mode switch was requested.

[SWS_Rte_02679] If acknowledgment is enabled for a provided ModeDeclarationGroupPrototype and a ModeSwitchedAckEvent references a RunnableEntity as well as the ModeDeclarationGroupPrototype, the RunnableEntity shall be activated when the mode switch acknowledgment occurs or when the RTE detects that any partition to which the mode users are mapped was stopped or restarted.

Agreed solution:

[SWS_Rte_03853] RTE_E_TIMEOUT Any mode-user partition is stopped or restarting or has been restarted
while the mode switch was requested.

[SWS_Rte_07055] SCHM_E_TIMEOUT The configured timeout exceeds before the mode transition was completed.
OR:
Any mode-user partition is stopped or restarting or has been restarted while the mode switch was requested.
–Last change on issue 73463 comment 2–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.37 Specification Item SWS_Rte_03869

Trace References:

SRS_Rte_00143, SRS_Rte_00018

Content:

RTE shall reject the configurations violating constr_4098.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76989: Missing constraints for mode disabling dependencies and OperationInvokedEvents / BswOperationInvokedEvents

  Problem description:

  The SWS RTE excludes
  the combination of mode disabling dependencies and OperationInvokedEvents ([SWS_Rte_02706])

  Reading the rational, the same combination shall be excluded for BswOperationInvokedEvents

  Agreed solution:

  SWC-T
  [constr_xxxx] No mode disabling for OperationInvokedEvents d An OperationInvokedEvent shall not have a reference to a ModeDeclaration in the role disabledMode. c()

  Rationale
  The RTE does not support the disabling of server Runnables by modes. Instead, the server shall respond with an explicit error code if the execution of the server operation is not possible in specific side conditions.

  BSWMD-T
  [constr_yyyy] No mode disabling for BswOperationInvokedEvents d An BswOperationInvokedEvents shall not have a reference to a ModeDeclaration in the role disabledInMode. c()

SWS RTE:
adjust
[SWS_Rte_02706] The RTE shall reject the configurations violating constr_xxxx
add
[SWS_Rte_nnnn1] d The RTE shall reject the configurations violating constr_yyyy
–Last change on issue 76989 comment 3–

BW-C-Level:

Application	Specification	Bus
4	4	1

1.38 Specification Item SWS_Rte_03983

Trace References:

SRS_Rte_00252

Content:

The The Module Interlink Types Header File shall include the definitions of all constants of ImplementationDataTypes and ApplicationDataTypes for each ImplementationDataType-/ApplicationDataTypes used (referencedSee SWS_Rte_08803 for the meaning of the term "used") by this Basic Software module.

This includes constants for CompuMethods referenced by ImplementationDataTypeElements of ImplementationDataTypes directly referenced by the Basic Software module and constants for CompuMethods of ImplementationDataTypes which are referenced indirectly via ImplementationDataTypes / ImplementationDataTypeElements of category TYPE_REFERENCE.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #73034: Clarification required regarding applicability of SWS_Rte_03809 - should labels be generated for mapped ImplementationDataTypes?

  Problem description:

  SWS_Rte_03809 states that, for a given ApplicationSoftwareComponent, the constants associated with all ‘used’ AutosarDataTypes must be included in the Application Types Header. It is noted that AutosarDataTypes referenced from relevant DataPrototypes and AutosarDataTypes referenced from IncludedDataType-Sets are all ‘used’ in this sense and further that sub-element data types of ‘used’ complex data types are also ‘used’, as are data types that are aliased by ‘used’ type reference data types.

What is not entirely clear is whether an ImplementationDataType that is mapped by the component to a ‘used’ ApplicationDataType is to be considered ‘used’ if it is not directly referenced from a DataPrototype or IncludedDataTypeSet itself.

On the one hand, this case is not explicitly included, although it is not clear whether the list is intended to be exhaustive. Indeed it does not mention element types of ‘used’ complex ApplicationDataTypes although it would seem strange if they were not to be included analogously to the ImplementationDataType case that is explicitly included.

On the other hand, the example given in the definition of ‘AI’ within Table 5.39 in the SoftwareComponentTemplate (AR4.2.2) suggests that the labels from both types may be used. Indeed, when a DataPrototype references an ApplicationDataType then the corresponding APIs use the mapped ImplementationDataType so it is reasonable to expect the software component code to use its labels, as well as the aliases from the ApplicationDataType. It could also be argued that the ImplementationDataType /is/ referenced by the component, via the referenced DataTypeMappingSet.

Issue # 72232 has clarified that in the case of labels associated with both an ApplicationDataType and its mapped ImplementationDataType an IncludedDataTypeSet referencing only the ApplicationDataType gives its literalPrefix (by SWS_Rte_03810) to the definitions for the ApplicationDataType only but it is not entirely clear whether the mapped ImplementationDataType should be considered ‘used’ at all in this case. Since there is no API or other code artefact resulting from an IncludedDataTypeSet it seems reasonable that different behaviour may be required in this respect compared to the case where the ApplicationDataType is referenced by a DataPrototype.

Having the labels defined for both types ensures that whichever labels the component code uses will be available but the downside is that conflicts between labels of different ImplementationDataTypes may occur and require entries in IncludedDataTypeSets to resolve even if the labels are not actually being used in the code. A mismatch of interpretations in either direction can be worked around using IncludedDataTypeSets but a common understanding is required to reduce this workload on integrators.

Agreed solution:

— SWS RTE —
[SWS_Rte_03809] d The Application Types Header File shall include the definitions of all constants of ImplementationDataTypes and ApplicationDataTypes for each ImplementationDataType/ApplicationDataTypes used (See [SWS_Rte_0xxx1]

for the meaning of the term "used") by this software component.

[SWS_Rte_03983] d The The Module Interlink Types Header File shall include the definitions of all constants of ImplementationDataTypes and Application-DataTypes for each ImplementationDataType/ApplicationDataTypes used (See [SWS_Rte_0xxx2] for the meaning of the term "used") by this Basic Software module.

Add [SWS_Rte_0xxx1] in chapter 5.5.5

[SWS_Rte_0xxx1] The meaning of the term "used" with respect to Autosar-DataTypes [
An AutosarDataType is used if it meets any one of the following conditions:

- it is referenced by a DataPrototype in the InternalBehavior, or
- it is referenced by a VariationPointProxy in the InternalBehavior, or
- it is referenced by a DataPrototype in a PortInterface referenced by a PortPrototype, or
- it is referenced by an IncludedDataTypeSet in the InternalBehavior, or
- it is the ImplementationDataType mapped to an ApplicationDataType (i.e. via the DataTypeMappingSet) that is used in one of the above ways, or
- it is an ImplementationDataTypeElement of a complex ImplementationDataType that is used in one of the above ways, or
- it is referenced as the target type of an ImplementationDataType or ImplementationDataTypeElement of category TYPE_REFERENCE that is used in one of the above ways, or
- it is an ApplicationDataType referenced as the type of a sub-element of a complex ApplicationDataType that is used in one of the above ways.
]
Please note that in contrast to the TYPE_REFERENCE case, when an ImplementationDataType of category DATA_REFERENCE is "used" the target ImplementationDataType it references is not considered used, unless it is independently used in its own right.
-
Add [SWS_Rte_0xxx2] in chapter 6.4.3
[SWS_Rte_0xxx2] The meaning of the term "used" with respect to Autosar-DataTypes [
An AutosarDataType is used if it meets any one of the following conditions:

- it is referenced by a DataPrototype in the InternalBehavior, or
- it is referenced by a VariationPointProxy in the internalBehavior, or
- it is referenced by a DataPrototype referenced by a providedData or requiredData, or

- it is referenced by an IncludedDataTypeSet in the InternalBehavior, or
- it is the ImplementationDataType mapped to an ApplicationDataType (i.e. via the DataTypeMappingSet) that is used in one of the above ways, or
- it is an ImplementationDataTypeElement of a complex ImplementationDataType that is used in one of the above ways, or
- it is referenced as the target type of an ImplementationDataType or ImplementationDataTypeElement of category TYPE_REFERENCE that is used in one of the above ways, or
- it is an ApplicationDataType referenced as the type of a sub-element of a complex ApplicationDataType that is used in one of the above ways.

]
Please note that in contrast to the TYPE_REFERENCE case, when an ImplementationDataType of category DATA_REFERENCE is "used" the target ImplementationDataType it references is not considered used, unless it is independently used in its own right.
–Last change on issue 73034 comment 14–

BW-C-Level:

Application	Specification	Bus
3	3	1

1.39 Specification Item SWS_Rte_04526

Trace References:

SRS_Rte_00019, SRS_Rte_00028

Content:

For inter-ECU transmission of composite data type where

- a SenderReceiverToSignalGroupMapping to the VariableDataPrototype is defined
- and the respective ISignalGroup has no ISignalGroup.comBasedSignalGroupTransformation defined

the RTE shall invoke Com_SendSignal (if parameter RteBswGeneral.RteUseComShadowSignalApi is |FALSE|) or Com_UpdateShadowSignal (if parameter RteBswGeneral.RteUseComShadowSignalApi is |TRUE|) for each ISignal to which an element in the composite data type is mapped and Com_SendSignalGroup for each ISignalGroup to which the composite data element is mapped.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77468: [RTE] Remove not existing COM services

  Problem description:

  The following API's are removed from COM:
  Com_UpdateShadowSignal,
  Com_InvalidateShadowSignal,
  Com_ReceiveShadowSignal

  Thus, also these references to this COM APIs should be removed from the RTE specification.

  Agreed solution:

  === RTE ===
  1.) update [SWS_Rte_05173]
  [SWS_Rte_05173] d The RTE shall use the ComHandleId of the corresponding Com-
  GroupSignal when invoking the COM API for group signals. c
  ...
  The input information ComHandleId of group signals is used to establish the link between
  the ComGroupSignal of the COM modules configuration and the corresponding ISignal of the System Template.

  2.) update text after [SWS_Rte_04527] in Chapter "4.3.1.11.2 Atomicity"

  from "The RTE decomposes the composite data type into single signals as described
  above and passes them to the COM module by using the COM API
  call Com_SendSignal (if parameter RteUseComShadowSignalApi is FALSE) or Com_UpdateShadowSignal (if parameter RteUseComShadowSignalApi is TRUE).
  ...
  As would be expected, the receiver side is the exact reverse of the transmission side: the RTE must first call Com_ReceiveSignalGroup precisely once for the signal group and then call Com_ReceiveSignal (if parameter RteUseComShadowSignalApi is FALSE) or Com_ReceiveShadowSignal (if parameter RteUseComShadowSig-nalApi
  is TRUE) to extract the value of each signal within the signal group."
  to
  "The RTE decomposes the composite data type into single signals as described above and passes them to the COM module by using the COM API call Com_SendSignal."
  ...
  As would be expected, the receiver side is the exact reverse of the transmission

side: the RTE must first call Com_ReceiveSignalGroup precisely once for the signal group and then call Com_ReceiveSignal to extract the value of each signal within the signal group."

delete "shadow" in following sentence:
A signal group has the additional property that COM guarantees to inform the receiver
by invoking a call-back about its arrival only after all signals belonging to the signal group have been unpacked into a shadow buffer.

3.) update text after [SWS_Rte_06023]

[SWS_Rte_06023]
....
no Data Transformation: RTE invokes Com_SendSignal for each primitive element (ISignal) in the composite data type and each COM signal to which that primitive element is mapped, and Com_SendSignalGroup for each ISignalGroup
that does not require a Data Transformation to which the data element is mapped.
...
the RTE shall invoke Com_SendSignal for each ISignal to which an element in the composite data type is mapped and Com_SendSignalGroup for each ISignalGroup to which
the composite data element is mapped.

4.) Update "Chapter 4.3.1.16.1 COM"
Generally the communication of arrays in the case of inter-ECU communication must make use of the signal group mechanisms to send an array to COM.
This implies sending each array element to a buffer in COM (with Com_SendSignal() API, and in the end send the signal group (with Com_SendSignalGroup() API).

5.) update description of chapter "5.11.5.3.3 Signal Invalidation"
A trace event indicating a signal invalidation request of an
Inter-ECU signal (or of a signal in a signal group) by the RTE.
Invoked by the RTE just before Com_InvalidateSignal is invoked.

6.) update [SWS_Rte_04526]
...
the RTE shall invoke Com_SendSignal for each ISignal to which an element in the composite data type is mapped and Com_SendSignalGroup for each ISignalGroup to which the composite

data element is mapped.

=== ECUC ===

1.) Set [ECUC_Rte_09107] to obsolete
–Last change on issue 77468 comment 4–

BW-C-Level:

Application	Specification	Bus
3	4	1

1.40 Specification Item SWS_Rte_04552

Trace References:

SRS_Rte_00116

Content:

The basic software scheduler shall ignore incoming client server communication requests, before the basic software scheduler is initialized completely or after it is stopped.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76201: Clarification of the ignore value of SWS_Rte_02535

  Problem description:

  SWS_Rte_02535 states that the:

  RTE shall ignore incoming client server communication requests,
  before RTE is initialized completely and when it is stopped.

  It is not specified though what return value the client shall receive in case an incoming client server communication request is ignored.

  Please specify the required return value.

  This RfC was created based on comment: https://www.autosar.org/bugzilla/show_bug.cgi?id=6596
  c24

  Agreed solution:

  Add requirement to chapter 4.6.1.5 (Finalization of the Basic Software Scheduler):
  [SWS_Rte_XXXX5] The basic software scheduler shall ignore incoming client

server communication requests,
before the basic software scheduler is initialized completely or after it is stopped.

Add requirement to the chapter 5.6 (API Reference) for the Rte_Call return values:
[SWS_Rte_XXXX0] RTE_E_TIMEOUT - if the call is ignored according to the SWS_Rte_02535

Add requirement to the chapter 5.6 (API Reference) for the Rte_Result return values:
[SWS_Rte_XXXX1] RTE_E_TIMEOUT - if the call is ignored according to the SWS_Rte_02535

Add requirement to the chapter 6.7 (API Reference) for the SchM_Call return values:
[SWS_Rte_XXXX2] SCHM_E_TIMEOUT - if the call is ignored according to the SWS_Rte_XXXX5

Add requirement to the chapter 6.7 (API Reference) for the SchM_Result return values:
[SWS_Rte_XXXX3] SCHM_E_TIMEOUT - if the call is ignored according to the SWS_Rte_XXXX5
–Last change on issue 76201 comment 11–

BW-C-Level:

Application	Specification	Bus
4	4	1

1.41 Specification Item SWS_Rte_04553

Trace References:

none

Content:

|RTE_E_TIMEOUT| – if the call is ignored according to SWS_Rte_02535

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76201: Clarification of the ignore value of SWS_Rte_02535

  Problem description:

SWS_Rte_02535 states that the:

RTE shall ignore incoming client server communication requests,
before RTE is initialized completely and when it is stopped.

It is not specified though what return value the client shall receive in case an incoming client server communication request is ignored.

Please specify the required return value.

This RfC was created based on comment: https://www.autosar.org/bugzilla/show_bug.cgi?id=6596
c24

Agreed solution:

Add requirement to chapter 4.6.1.5 (Finalization of the Basic Software Scheduler):
[SWS_Rte_XXXX5] The basic software scheduler shall ignore incoming client server communication requests,
before the basic software scheduler is initialized completely or after it is stopped.

Add requirement to the chapter 5.6 (API Reference) for the Rte_Call return values:
[SWS_Rte_XXXX0] RTE_E_TIMEOUT - if the call is ignored according to the SWS_Rte_02535

Add requirement to the chapter 5.6 (API Reference) for the Rte_Result return values:
[SWS_Rte_XXXX1] RTE_E_TIMEOUT - if the call is ignored according to the SWS_Rte_02535

Add requirement to the chapter 6.7 (API Reference) for the SchM_Call return values:
[SWS_Rte_XXXX2] SCHM_E_TIMEOUT - if the call is ignored according to the SWS_Rte_XXXX5

Add requirement to the chapter 6.7 (API Reference) for the SchM_Result return values:
[SWS_Rte_XXXX3] SCHM_E_TIMEOUT - if the call is ignored according to the SWS_Rte_XXXX5
–Last change on issue 76201 comment 11–

BW-C-Level:

Application	Specification	Bus
4	4	1

1.42 Specification Item SWS_Rte_04554

Trace References:

none

Content:

|RTE_E_TIMEOUT| – if the call is ignored according to SWS_Rte_02535

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76201: Clarification of the ignore value of SWS_Rte_02535

  Problem description:

  SWS_Rte_02535 states that the:

  RTE shall ignore incoming client server communication requests,
  before RTE is initialized completely and when it is stopped.

  It is not specified though what return value the client shall receive in case an incoming client server communication request is ignored.

  Please specify the required return value.

  This RfC was created based on comment: https://www.autosar.org/bugzilla/show_bug.cgi?id=6596
  c24

  Agreed solution:

  Add requirement to chapter 4.6.1.5 (Finalization of the Basic Software Scheduler):
  [SWS_Rte_XXXX5] The basic software scheduler shall ignore incoming client server communication requests,
  before the basic software scheduler is initialized completely or after it is stopped.

  Add requirement to the chapter 5.6 (API Reference) for the Rte_Call return values:
  [SWS_Rte_XXXX0] RTE_E_TIMEOUT - if the call is ignored according to the SWS_Rte_02535

  Add requirement to the chapter 5.6 (API Reference) for the Rte_Result return values:
  [SWS_Rte_XXXX1] RTE_E_TIMEOUT - if the call is ignored according to the SWS_Rte_02535

Add requirement to the chapter 6.7 (API Reference) for the SchM_Call return values:
[SWS_Rte_XXXX2] SCHM_E_TIMEOUT - if the call is ignored according to the SWS_Rte_XXXX5

Add requirement to the chapter 6.7 (API Reference) for the SchM_Result return values:
[SWS_Rte_XXXX3] SCHM_E_TIMEOUT - if the call is ignored according to the SWS_Rte_XXXX5
–Last change on issue 76201 comment 11–

BW-C-Level:

Application	Specification	Bus
4	4	1

1.43 Specification Item SWS_Rte_04555

Trace References:

none

Content:

|SCHM_E_TIMEOUT| – if the call is ignored according to SWS_Rte_04552

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76201: Clarification of the ignore value of SWS_Rte_02535

  Problem description:

  SWS_Rte_02535 states that the:

  RTE shall ignore incoming client server communication requests,
  before RTE is initialized completely and when it is stopped.

  It is not specified though what return value the client shall receive in case an incoming client server communication request is ignored.

  Please specify the required return value.

  This RfC was created based on comment: https://www.autosar.org/bugzilla/show_bug.cgi?id=6596
  c24

Agreed solution:

Add requirement to chapter 4.6.1.5 (Finalization of the Basic Software Scheduler):
[SWS_Rte_XXXX5] The basic software scheduler shall ignore incoming client server communication requests,
before the basic software scheduler is initialized completely or after it is stopped.

Add requirement to the chapter 5.6 (API Reference) for the Rte_Call return values:
[SWS_Rte_XXXX0] RTE_E_TIMEOUT - if the call is ignored according to the SWS_Rte_02535

Add requirement to the chapter 5.6 (API Reference) for the Rte_Result return values:
[SWS_Rte_XXXX1] RTE_E_TIMEOUT - if the call is ignored according to the SWS_Rte_02535

Add requirement to the chapter 6.7 (API Reference) for the SchM_Call return values:
[SWS_Rte_XXXX2] SCHM_E_TIMEOUT - if the call is ignored according to the SWS_Rte_XXXX5

Add requirement to the chapter 6.7 (API Reference) for the SchM_Result return values:
[SWS_Rte_XXXX3] SCHM_E_TIMEOUT - if the call is ignored according to the SWS_Rte_XXXX5
–Last change on issue 76201 comment 11–

BW-C-Level:

Application	Specification	Bus
4	4	1

1.44 Specification Item SWS_Rte_04556

Trace References:

none

Content:

|SCHM_E_TIMEOUT| – if the call is ignored according to SWS_Rte_04552

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76201: Clarification of the ignore value of SWS_Rte_02535

  Problem description:

  SWS_Rte_02535 states that the:

  RTE shall ignore incoming client server communication requests,
  before RTE is initialized completely and when it is stopped.

  It is not specified though what return value the client shall receive in case an incoming client server communication request is ignored.

  Please specify the required return value.

  This RfC was created based on comment: https://www.autosar.org/bugzilla/show_bug.cgi?id=6596
  c24

  Agreed solution:

  Add requirement to chapter 4.6.1.5 (Finalization of the Basic Software Scheduler):
  [SWS_Rte_XXXX5] The basic software scheduler shall ignore incoming client server communication requests,
  before the basic software scheduler is initialized completely or after it is stopped.

  Add requirement to the chapter 5.6 (API Reference) for the Rte_Call return values:
  [SWS_Rte_XXXX0] RTE_E_TIMEOUT - if the call is ignored according to the SWS_Rte_02535

  Add requirement to the chapter 5.6 (API Reference) for the Rte_Result return values:
  [SWS_Rte_XXXX1] RTE_E_TIMEOUT - if the call is ignored according to the SWS_Rte_02535

  Add requirement to the chapter 6.7 (API Reference) for the SchM_Call return values:
  [SWS_Rte_XXXX2] SCHM_E_TIMEOUT - if the call is ignored according to the SWS_Rte_XXXX5

  Add requirement to the chapter 6.7 (API Reference) for the SchM_Result return values:
  [SWS_Rte_XXXX3] SCHM_E_TIMEOUT - if the call is ignored according to the SWS_Rte_XXXX5
  –Last change on issue 76201 comment 11–

BW-C-Level:

Application	Specification	Bus
4	4	1

1.45 Specification Item SWS_Rte_04557

Trace References:

SRS_Rte_00049, SRS_BSW_00351

Content:

The RTE Generator shall wrap each each definition of a task body with the Memory Mapping.

```
#define OS_START_SEC_<sadm>
#include "Os_MemMap.h"

<task body definition>

#define OS_STOP_SEC_<sadm>
#include "Os_MemMap.h"
```

where |<sadm>| is the Referrable.shortName of the SwAddrMethod, if configured in Os MemoryMappingCodeLocationRef of the according OsTask. If OsMemoryMappingCode LocationRef is not defined , |<sadm>| shall be |CODE_<Taskname>|.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #74978: Mismatch in memory allocation for Os Tasks

  Problem description:

  Currently following situation occurs:
  The OS so declared the Task Bodies (in order to mount in its own configuration structures)
  The RTE in turn defines some of the Task Bodies.
  But both do that with different memory sections since there is no common source of information which memory section is requested for a Task Bodies.

  In the meantime Compilers are on the market with a quite strict checking of the memory relates settings and in case of mismatch those compilers issue an error.

  Proposal: add a configuration parameter at OsTask referencing a SwAddrMethod.
  The OS shall use this SwAddrMethod in order to generate the correct MemMap

MemoryAllocationKeywords (like Rte for Runnables)
E.g.
1 # define OS_START_SEC_<sadm>
2 # include "Os_MemMap.h"
3

/.. Task Body or Task forward declaration ..//

8
9 # define OS_STOP_SEC_<sadm>
10 # include "Os_MemMap.h"

Agreed solution:

ECUC of SWS OS:
Add a reference attribute to the following Os config containers: OsTask (ECUC_Os_00073), OsISR (ECUC_Os_00041), OsApplicationHooks (ECUC_Os_00020) and OsHooks (ECUC_Os_00035)
Details:
Name: OsMemoryMappingCodeLocationRef
Description Reference to the memory mapping containing details about the section where the code is placed
Multiplicity: 0..1
Type: Foreign reference to SW-ADDR-METHOD
No default value
Not post build-able
Pre-compile time All Variants
Scope: ECU

—

Add a requirement to SWS_OS

[SWS_Os_xxxxx] d The OS code shall wrap each declaration of Task, ISR and hook functions with the Memory Mapping Allocation Keywords macros.
1 # define OS_START_SEC_<sadm>
2 # include "Os_MemMap.h"
3
4 <Task, ISR or hook functions declaration>
5
6 # define OS_STOP_SEC_<sadm>
7 # include "Os_MemMap.h"

where <sadm> is the shortName of the SwAddrMethod if configured in Os-

MemoryMappingCodeLocationRef. d (SRS_BSW_00351)

—

SWS RTE:

[SWS_Rte_yyyyy] d The RTE Generator shall wrap each definition of a task body
with the Memory Allokation Keywords.
1 # define OS_START_SEC_<sadm>
2 # include "Os_MemMap.h"
3
4 <Task definition>
5
6 # define OS_STOP_SEC_<sadm>
7 # include "Os_MemMap.h"

where <sadm> is the shortName of the SwAddrMethod if configured in OsMemoryMappingCodeLocationRef of the according OsTask.
If OsMemoryMappingCodeLocationRef is not defined , <sadm> shall be CODE_<Taskname>. d (SRS_BSW_00351)

Add a note below:
"Requirement SWS_Rte_yyyyy is an exception to SWS_Rte_05088."
–Last change on issue 74978 comment 12–

BW-C-Level:

Application	Specification	Bus
1	3	1

1.46 Specification Item SWS_Rte_05111

Trace References:

SRS_Rte_00018, SRS_Rte_00029, SRS_Rte_00082

Content:

The RTE Generator shall reject configurations where there is inter-ECU client-server communication from several clients on the same client-ECU and no client identifier is configured for at least one of these inter-ECU client-server communications.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #69594: [SWS_Rte_05111] is not detectable by the RTE generator

  Problem description:

  I believe that requirement [SWS_Rte_05111] is not detectable by the RTE generator.
  My interpretation of the requirement is that the RTE shall reject n:1 communication from one ECU to a server unless a client identifier is used.
  On client-ECU it is not possible to detect if the Clients are communicating with the same server.
  On server-ECU it is not possible to detect which Client-ECU is the origin of the client.

  Agreed solution:

  Remove SWS_Rte_05111.
  Remove SWS_Rte_02579.
  Add a reference to the constraint 3264 at the end of the preceding paragraph.
  "...if the client identifier is used to distinguish the different clients (see constr_3264)."
  –Last change on issue 69594 comment 16–

  BW-C-Level:

Application	Specification	Bus
1	1	1

1.47 Specification Item SWS_Rte_05173

Trace References:

SRS_Rte_00091

Content:

The RTE shall use the ComGroupSignal.ComHandleId of the corresponding ComGroup Signal when invoking the COM API for shadow group signals.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77468: [RTE] Remove not existing COM services

  Problem description:

  The following API's are removed from COM:
  Com_UpdateShadowSignal,
  Com_InvalidateShadowSignal,
  Com_ReceiveShadowSignal

Thus, also these references to this COM APIs should be removed from the RTE specification.

Agreed solution:

=== RTE ===
1.) update [SWS_Rte_05173]
[SWS_Rte_05173] d The RTE shall use the ComHandleId of the corresponding Com-
GroupSignal when invoking the COM API for group signals. c
...
The input information ComHandleId of group signals is used to establish the link between
the ComGroupSignal of the COM modules configuration and the corresponding ISignal of the System Template.

2.) update text after [SWS_Rte_04527] in Chapter "4.3.1.11.2 Atomicity"

from "The RTE decomposes the composite data type into single signals as described
above and passes them to the COM module by using the COM API
call Com_SendSignal (if parameter RteUseComShadowSignalApi is FALSE) or Com_UpdateShadowSignal (if parameter RteUseComShadowSignalApi is TRUE).
...
As would be expected, the receiver side is the exact reverse of the transmission side: the RTE must first call Com_ReceiveSignalGroup precisely once for the signal group and then call Com_ReceiveSignal (if parameter RteUseComShadowSignalApi is FALSE) or Com_ReceiveShadowSignal (if parameter RteUseComShadowSig-nalApi
is TRUE) to extract the value of each signal within the signal group."
to
"The RTE decomposes the composite data type into single signals as described above and passes them to the COM module by using the COM API call Com_SendSignal."
...
As would be expected, the receiver side is the exact reverse of the transmission side: the RTE must first call Com_ReceiveSignalGroup precisely once for the signal group and then call Com_ReceiveSignal to extract the value of each signal within the signal group."

delete "shadow" in following sentence:
A signal group has the additional property that COM guarantees to inform the receiver

by invoking a call-back about its arrival only after all signals belonging to the signal group have been unpacked into a shadow buffer.

3.) update text after [SWS_Rte_06023]

[SWS_Rte_06023]
....
no Data Transformation: RTE invokes Com_SendSignal for each primitive element (ISignal) in the composite data type and each COM signal to which that primitive element is mapped, and Com_SendSignalGroup for each ISignalGroup
that does not require a Data Transformation to which the data element is mapped.
...
the RTE shall invoke Com_SendSignal for each ISignal to which an element in the composite data type is mapped and Com_SendSignalGroup for each ISignalGroup to which
the composite data element is mapped.

4.) Update "Chapter 4.3.1.16.1 COM"
Generally the communication of arrays in the case of inter-ECU communication must make use of the signal group mechanisms to send an array to COM.
This implies sending each array element to a buffer in COM (with Com_SendSignal() API, and in the end send the signal group (with Com_SendSignalGroup() API).

5.) update description of chapter "5.11.5.3.3 Signal Invalidation"
A trace event indicating a signal invalidation request of an
Inter-ECU signal (or of a signal in a signal group) by the RTE.
Invoked by the RTE just before Com_InvalidateSignal is invoked.

6.) update [SWS_Rte_04526]
...
the RTE shall invoke Com_SendSignal for each ISignal to which an element in the composite data type is mapped and Com_SendSignalGroup for each ISignalGroup to which the composite
data element is mapped.

=== ECUC ===

1.) Set [ECUC_Rte_09107] to obsolete
–Last change on issue 77468 comment 4–

BW-C-Level:

Application	Specification	Bus
3	4	1

1.48 Specification Item SWS_Rte_06061

Trace References:

SRS_Rte_00244

Content:

For all RP service function defined by the input configuration the RTE generator shall invoke the RP service function in alphabetical order (ASCII / ISO 8859-1 code in ascending order).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76347: Explicit specification of alphabetical ordering

  Problem description:

  Requirements SWS_Rte_06061, SWS_Rte_03812, and SWS_Rte_07228 do not mention the exact criteria for alphabetical ordering.

  Agreed solution:

  Requirements SWS_Rte_06061, SWS_Rte_03812, and SWS_Rte_07228:
  Add the parenthesis phrase "(ASCII / ISO 8859-1 code in ascending order)" after "alphabetic ordering".
  –Last change on issue 76347 comment 1–

  BW-C-Level:

Application	Specification	Bus
1	4	1

1.49 Specification Item SWS_Rte_06113

Trace References:

SRS_Rte_00045, SRS_Rte_00244

Content:

```
void Rte_[<client>_]RptHook_<cts>_<var>_Transmit
     ([const Rte_CDS_<cts>*], <type>* <buffer> )
```

Where |<cts>| is the component type symbol of the AtomicSwComponentType, |<var>| the identifying name of the RP global buffer, e.g. port and data element names. |<buffer>| is a reference to the RP global buffer.

The instance handle is included if and only if the software component's SwcInternalBehavior.supportsMultipleInstantiation attribute is set to true. Note that |Rte_Instance| cannot be used directly, as there will be pointers to multiple components' structure types within the single VFB Tracing header file, and |Rte_Instance| would therefore be ambiguous.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76346: Component data structure type definition vs. structure declaration

  Problem description:

  In general the RTE SWS distinguishes between a structure declaration and a type declaration for a composite datatype that represents a structure.

  For a structure implementation data type SWS_Rte_07114 requires a structure declaration:

  struct _<name>_
  ...
  ;

  SWS_Rte_06812 requires the type declaration:

  typedef struct _<name>_ <name>;

  Regarding the component data structure currently there exist the following requirements only:

  [SWS_Rte_07132] The component data structure type shall be defined in the Application Header file.
  [SWS_Rte_03714] The type name of the component data structure shall be Rte_[Byps]_CDS_<cts> where <cts> is the component type symbol of the AtomicSwComponentType. ...

  Since SWS_Rte_03714 talks about the type name of the component data structure this requires a typedef statement for Rte_[Byps]_CDS_<cts> in the Application Header file:

typedef struct ... Rte_[Byps]_CDS_<cts>;

Currently there does not exist any requirement for a CDS structure declaration.

The code examples in the SWS RTE are somehow in contradiction to these requirements. They use "struct

Rte_CDS_<cts>" instead of the typedef name. See examples 5.24, 5.26, 5.28.

This inconsistency applies to the signature of some APIs, too:

[SWS_Rte_01238]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Start([const struct Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01239]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Return([const struct Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01248]
void Rte_[<client>_]Runnable_<cts>_<reName>_Start([const RTE_CDS_<cts>*])

[SWS_Rte_01249]
void Rte_[<client>_]Runnable_<cts>_<reName>_Return([const Rte_CDS_<cts>*])

[SWS_Rte_06113]
void Rte_[<client>_]RptHook_<cts>_<var>_Transmit([const RTE_CDS_<cts>*], <type>* <buffer>)

[SWS_Rte_06114]
void Rte_[<client>_]RptHook_<cts>_<var>_Reception([const RTE_CDS_<cts>*], <type>* <buffer>)

In addition some of these APIs use RTE_CDS_ instead of Rte_CDS_.

Agreed solution:

Delete SWS_Rte_07132 and SWS_Rte_03714.

Add new requirements:

[SWS_Rte_xxxx1] The Application Header File shall include a structure declaration for the component data structure as follows:

struct Rte_[Byps_]CDS_<cts> <component data sections> ;
where <cts> is the component type symbol of the AtomicSwComponentType. [Byps_] is an optional infix used when component wrapper method for bypass support is enabled for the related software componenttype (See chapter 4.9.2).

[SWS_Rte_xxxx2] The Application Header File shall include a type declaration for the component data structure type as follows:
typedef struct Rte_[Byps_]CDS_<cts> Rte_[Byps_]CDS_<cts>;
where <cts> is the component type symbol of the AtomicSwComponentType.[Byps_] is an optional infix used when component wrapper method for bypass support is enabled for the related software componenttype (See chapter 4.9.2).

Change the API signatures to:

[SWS_Rte_01238]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Start([const Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01239]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Return([const Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01248]
void Rte_[<client>_]Runnable_<cts>_<reName>_Start([const Rte_CDS_<cts>*])

[SWS_Rte_06113]
void Rte_[<client>_]RptHook_<cts>_<var>_Transmit([const Rte_CDS_<cts>*], <type>* <buffer>)

[SWS_Rte_06114]
void Rte_[<client>_]RptHook_<cts>_<var>_Reception([const Rte_CDS_<cts>*], <type>* <buffer>)

Change examples 5.24, 5.26, 5.28 so that they contain both structure declaration and typedef for component data structure type and use the type (without struct) at the interfaces.
–Last change on issue 76346 comment 1–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.50 Specification Item SWS_Rte_06114

Trace References:

SRS_Rte_00045, SRS_Rte_00244

Content:

```
void Rte_[<client>_]RptHook_<cts>_<var>_Reception
     ([const Rte_CDS_<cts>*], <type>* <buffer> )
```

Where |<cts>| is the component type symbol of the AtomicSwComponentType, |<var>| the identifying name of the RP global buffer, e.g. port and data element names. |<buffer>| is a reference to the RP global buffer.

The instance handle is included if and only if the software component's SwcInternalBehavior.supportsMultipleInstantiation attribute is set to true. Note that |Rte_Instance| cannot be used directly, as there will be pointers to multiple components' structure types within the single VFB Tracing header file, and |Rte_Instance| would therefore be ambiguous.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76346: Component data structure type definition vs. structure declaration

  Problem description:

  In general the RTE SWS distinguishes between a structure declaration and a type declaration for a composite datatype that represents a structure.

  For a structure implementation data type SWS_Rte_07114 requires a structure declaration:

  struct _<name>_
  ...
  ;

  SWS_Rte_06812 requires the type declaration:

  typedef struct _<name>_ <name>;

  Regarding the component data structure currently there exist the following requirements only:

  [SWS_Rte_07132] The component data structure type shall be defined in the Application Header file.
  [SWS_Rte_03714] The type name of the component data structure shall be

Rte_[Byps]_CDS_<cts> where <cts> is the component type symbol of the AtomicSwComponentType. ...

Since SWS_Rte_03714 talks about the type name of the component data structure this requires a typedef statement for Rte_[Byps]_CDS_<cts> in the Application Header file:

typedef struct ... Rte_[Byps]_CDS_<cts>;

Currently there does not exist any requirement for a CDS structure declaration.

The code examples in the SWS RTE are somehow in contradiction to these requirements. They use "struct

Rte_CDS_<cts>" instead of the typedef name. See examples 5.24, 5.26, 5.28.

This inconsistency applies to the signature of some APIs, too:

[SWS_Rte_01238]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Start([const struct Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01239]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Return([const struct Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01248]
void Rte_[<client>_]Runnable_<cts>_<reName>_Start([const RTE_CDS_<cts>*])

[SWS_Rte_01249]
void Rte_[<client>_]Runnable_<cts>_<reName>_Return([const Rte_CDS_<cts>*])

[SWS_Rte_06113]
void Rte_[<client>_]RptHook_<cts>_<var>_Transmit([const RTE_CDS_<cts>*], <type>* <buffer>)

[SWS_Rte_06114]
void Rte_[<client>_]RptHook_<cts>_<var>_Reception([const RTE_CDS_<cts>*], <type>* <buffer>)

In addition some of these APIs use RTE_CDS_ instead of Rte_CDS_.

Agreed solution:

Delete SWS_Rte_07132 and SWS_Rte_03714.

Add new requirements:

[SWS_Rte_xxxx1] The Application Header File shall include a structure declaration for the component data structure as follows:
struct Rte_[Byps_]CDS_<cts> <component data sections> ;
where <cts> is the component type symbol of the AtomicSwComponentType. [Byps_] is an optional infix used when component wrapper method for bypass support is enabled for the related software componenttype (See chapter 4.9.2).

[SWS_Rte_xxxx2] The Application Header File shall include a type declaration for the component data structure type as follows:
typedef struct Rte_[Byps_]CDS_<cts> Rte_[Byps_]CDS_<cts>;
where <cts> is the component type symbol of the AtomicSwComponentType.[Byps_] is an optional infix used when component wrapper method for bypass support is enabled for the related software componenttype (See chapter 4.9.2).

Change the API signatures to:

[SWS_Rte_01238]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Start([const Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01239]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Return([const Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01248]
void Rte_[<client>_]Runnable_<cts>_<reName>_Start([const Rte_CDS_<cts>*])

[SWS_Rte_06113]
void Rte_[<client>_]RptHook_<cts>_<var>_Transmit([const Rte_CDS_<cts>*], <type>* <buffer>)

[SWS_Rte_06114]
void Rte_[<client>_]RptHook_<cts>_<var>_Reception([const Rte_CDS_<cts>*], <type>* <buffer>)

Change examples 5.24, 5.26, 5.28 so that they contain both structure declaration and typedef for component data structure type and use the type (without

struct) at the interfaces.
–Last change on issue 76346 comment 1–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.51 Specification Item SWS_Rte_06611

Trace References:

SRS_Rte_00191

Content:

If the DET is enabled then the RTE shall generate validation code which at runtime (i.e. during initialization) validates the resolved post-build variants and check the integrity with regards to the active variants. If a violation is detected the RTE shall report a default development error to the DET. To execute this validation RTE initialization will get a pointer to the RtePostBuildVariantConfiguration instance to allow it to validate the selected variant.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #73570: No "default error" in AUTOSAR

  Problem description:

  The DET was renamed from development error tracer to default error tracer.

  This change was most of the time done automatically and unfortunately renamed "developement error" to "default error".

  "default error" should always be followed by "tracer", otherwise, "development error" is probably the right term.

  This could increase the impact (compared to my selection of impacted document, but formally, the configuration parameters *DevErrorDetect are not using the correct description:
  "Switches the Default Error Tracer (Det) detection and notification..."

  The parameter switches on/off the developement error detection. The DET does not need to be detected and can be present even when the parameter is set to false.

Agreed solution:

Rename "default error" to "development error" in all impacted documents, but not in an automated way (Do not change "default error tracer" to "developement error tracer"!)

Blueprint/Example:
- sub chapter is now called "7.x Default errors"
- "[SWS_xxx_yyyyy]
In case default error detection is enabled for the xxxx module: The xxxx module shall check API parameters for validity and report detected errors to the DET. ()"
- "[SWS_xxx_yyyyy]
If default error detection is enabled: the function shall check that the service xxx_Init was previously called. If the check fails, the function shall raise the default error XXX_E_NOT_INITIALIZED otherwise (if DET is disabled) return E_NOT_OK. ()"
- "In case default errors are enabled,..."
- "module raises the Default error XXX_E_TRANSITION"
- "The DET provides services to store default errors"
...

The correct text would be:
- sub chapter is called "7.x Development errors"
- "[SWS_xxx_yyyyy]
In case development error detection is enabled for the xxxx module: The xxxx module shall check API parameters for validity and report detected development errors to the DET. ()"
- "[SWS_xxx_yyyyy]
If development error detection is enabled: the function shall check that the service xxx_Init was previously called. If the check fails, the function shall raise the development error XXX_E_NOT_INITIALIZED otherwise (if DET is disabled) return E_NOT_OK. ()"
- "In case development errors are enabled,..."
- "module raises the development error XXX_E_TRANSITION"
- "The DET provides services to store development errors"

Solution for SWS_RTE:
– SWS_RTE —
- Change 4.8 Default errors to 4.8 Development errors
- Change "Errors which can occur at runtime in the RTE are classified as default errors" to "Errors which can occur at runtime in the RTE are classified as development errors"
- Remove [SWS_Rte_07676]

- Change [SWS_RTE_06611]"If a violation is detected the RTE shall report a default error to the DET." to "If a violation is detected the RTE shall report a development error to the DET."
- Change [SWS_Rte_06631]
[SWS_Rte_06631] d The RTE shall use the OS Application Identifier as the Instance Id to enable the developer to identify in which runtime section of the RTE the error occurs. This Instance ID is even unique across multi cores and so implicitly allows the development error to be traced to a specific core. c(SRS_BSW_00337)

SRS_Libraries:
- In chapter "3 Acronyms and abbreviations": Rename "Development Error Tracer" to "Default Error Tracer"

SRS_SPALGeneral:
- In chapter "6.1.1.3.1 [SRS_SPAL_00157] ...": Rename "Development Error Tracer" to "Default Error Tracer"
- In chapter "6.1.1.4.2 [SRS_SPAL_12448] ...": Rename "Development Error Tracer" to "Default Error Tracer"

SRS_FlashTest:
- In chapter "6.1 Functional Requirements": Rename "Development Error Tracer" to "Default Error Tracer"
- In chapter "7 References":
Rename "Development Error Tracer" to "Default Error Tracer"
Rename "AUTOSAR_SWS_DevelopmentErrorTracer" to "AUTOSAR_SWS_DefaultErrorTracer"

SWS_MFXLibrary:
- In chapter "2 Acronyms and abbreviations": Rename "Development Error Tracer" to "Default Error Tracer"

SWS_MemoryAbstractionInterface:
- In chapter "3.1 Input documents":
Rename "Development Error Tracer" to "Default Error Tracer"
Rename "AUTOSAR_SWS_DevelopmentErrorTracer" to "AUTOSAR_SWS_DefaultErrorTracer"

SWS_FlexRayNetworkManagement:
- In chapter "3.3 Related AUTOSAR documents":
Rename "Development Error Tracer" to "Default Error Tracer"
Rename "AUTOSAR_SWS_DevelopmentErrorTracer" to "AUTOSAR_SWS_DefaultErrorTracer"

SWS_CANStateManager:
- In chapter "3.1 Input documents": Rename "AUTOSAR_SWS_DevelopmentErrorTracer" to "AUTOSAR_SWS_DefaultErrorTracer"

SWS_PDURouter:
- In chapter "3.1 Input documents": Rename "AUTOSAR_SWS_DevelopmentErrorTracer" to "AUTOSAR_SWS_DefaultErrorTracer"

SWS_EEPROMDriver:
- In chapter "3.1 Input documents": Rename "AUTOSAR_SWS_DevelopmentErrorTracer" to "AUTOSAR_SWS_DefaultErrorTracer"
–Last change on issue 73570 comment 47–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.52 Specification Item SWS_Rte_06631

Trace References:

SRS_BSW_00337

Content:

The RTE shall use the OS Application Identifier as the Instance Id to enable the developer to identify in which runtime section of the RTE the error occurs. This Instance ID is even unique across multi cores and so implicitly allows the default development error to be traced to a specific core.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #73570: No "default error" in AUTOSAR

Problem description:

The DET was renamed from development error tracer to default error tracer.

This change was most of the time done automatically and unfortunately renamed "developement error" to "default error".

"default error" should always be followed by "tracer", otherwise, "development error" is probably the right term.

This could increase the impact (compared to my selection of impacted document, but formally, the configuration parameters *DevErrorDetect are not using the correct description:
"Switches the Default Error Tracer (Det) detection and notification..."

The parameter switches on/off the developement error detection. The DET does not need to be detected and can be present even when the parameter is set to false.

Agreed solution:

Rename "default error" to "development error" in all impacted documents, but not in an automated way (Do not change "default error tracer" to "developement error tracer"!)

Blueprint/Example:
- sub chapter is now called "7.x Default errors"
- "[SWS_xxx_yyyyy]
In case default error detection is enabled for the xxxx module: The xxxx module shall check API parameters for validity and report detected errors to the DET. ()"
- "[SWS_xxx_yyyyy]
If default error detection is enabled: the function shall check that the service xxx_Init was previously called. If the check fails, the function shall raise the default error XXX_E_NOT_INITIALIZED otherwise (if DET is disabled) return E_NOT_OK. ()"
- "In case default errors are enabled,..."
- "module raises the Default error XXX_E_TRANSITION"
- "The DET provides services to store default errors"
...

The correct text would be:
- sub chapter is called "7.x Development errors"
- "[SWS_xxx_yyyyy]
In case development error detection is enabled for the xxxx module: The xxxx

module shall check API parameters for validity and report detected development errors to the DET. ()"
- "[SWS_xxx_yyyyy]
If development error detection is enabled: the function shall check that the service xxx_Init was previously called. If the check fails, the function shall raise the development error XXX_E_NOT_INITIALIZED otherwise (if DET is disabled) return E_NOT_OK. ()"
- "In case development errors are enabled,..."
- "module raises the development error XXX_E_TRANSITION"
- "The DET provides services to store development errors"

Solution for SWS_RTE:
– SWS_RTE —
- Change 4.8 Default errors to 4.8 Development errors
- Change "Errors which can occur at runtime in the RTE are classified as default errors" to "Errors which can occur at runtime in the RTE are classified as development errors"
- Remove [SWS_Rte_07676]
- Change [SWS_RTE_06611]"If a violation is detected the RTE shall report a default error to the DET." to "If a violation is detected the RTE shall report a development error to the DET."
- Change [SWS_Rte_06631]
[SWS_Rte_06631] d The RTE shall use the OS Application Identifier as the Instance Id to enable the developer to identify in which runtime section of the RTE the error occurs. This Instance ID is even unique across multi cores and so implicitly allows the development error to be traced to a specific core. c(SRS_BSW_00337)

SRS_Libraries:
- In chapter "3 Acronyms and abbreviations": Rename "Development Error Tracer" to "Default Error Tracer"

SRS_SPALGeneral:
- In chapter "6.1.1.3.1 [SRS_SPAL_00157] ...": Rename "Development Error Tracer" to "Default Error Tracer"
- In chapter "6.1.1.4.2 [SRS_SPAL_12448] ...": Rename "Development Error Tracer" to "Default Error Tracer"

SRS_FlashTest:
- In chapter "6.1 Functional Requirements": Rename "Development Error Tracer" to "Default Error Tracer"

- In chapter "7 References":
Rename "Development Error Tracer" to "Default Error Tracer"
Rename "AUTOSAR_SWS_DevelopmentErrorTracer" to "AUTOSAR_SWS_DefaultErrorTracer"

SWS_MFXLibrary:
- In chapter "2 Acronyms and abbreviations": Rename "Development Error Tracer" to "Default Error Tracer"

SWS_MemoryAbstractionInterface:
- In chapter "3.1 Input documents":
Rename "Development Error Tracer" to "Default Error Tracer"
Rename "AUTOSAR_SWS_DevelopmentErrorTracer" to "AUTOSAR_SWS_DefaultErrorTracer"

SWS_FlexRayNetworkManagement:
- In chapter "3.3 Related AUTOSAR documents":
Rename "Development Error Tracer" to "Default Error Tracer"
Rename "AUTOSAR_SWS_DevelopmentErrorTracer" to "AUTOSAR_SWS_DefaultErrorTracer"

SWS_CANStateManager:
- In chapter "3.1 Input documents": Rename "AUTOSAR_SWS_DevelopmentErrorTracer" to "AUTOSAR_SWS_DefaultErrorTracer"

SWS_PDURouter:
- In chapter "3.1 Input documents": Rename "AUTOSAR_SWS_DevelopmentErrorTracer" to "AUTOSAR_SWS_DefaultErrorTracer"

SWS_EEPROMDriver:
- In chapter "3.1 Input documents": Rename "AUTOSAR_SWS_DevelopmentErrorTracer" to "AUTOSAR_SWS_DefaultErrorTracer"
–Last change on issue 73570 comment 47–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.53 Specification Item SWS_Rte_06706

Trace References:

SRS_Rte_00055, SRS_Rte_00164

Content:

For each Array Implementation Data Type which last ImplementationDataTypeElement is of category STRUCTURE, the RTE Types Header File shall include the corresponding type declaration as:

|typedef| |struct| | <elements> | |<type>| |<name>[<size 1>][<size 2>]...[<size n>];|
|[<size n>]|;

where |<elements>|<type>| is the record element specification and

Implementation Data Type Element shortName, |<name>| is the Implementation Data Type symbol of the Array Implementation Data Type. For each record element defined by one ImplementationDataTypeElement one record element specification |<elements>| is defined. The record element specifications are ordered according the order of the related ImplementationDataTypeElements in the input configuration. Sequent record elements are separated with a semicolon.

, |[<size x>]| is the ImplementationDataTypeElement.arraySize of the Array's ImplementationDataTypeElement. For each array dimension defined by one Array's Implementation DataTypeElement one array dimension definition |[<size x>]| is defined.

The array dimension definitions |[<size 1>], [<size 2>] ... [<size n>]| ordered from the root to the last ImplementationDataTypeElement belonging to the array definition.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77453: "[SWS_Rte_07072]" and "[SWS_Rte_06706]/[SWS_Rte_06707]" lead to uncompilable code

  Problem description:

  According to [SWS_Rte_06706] and [SWS_Rte_06707] a typedef shall be generated for each ArrayImplementationDataType which last ImplementationDataTypeElement is of category STRUCTURE,UNION.

  [SWS_Rte_06706] d For each Array Implementation Data Type which last ImplementationDataTypeElement is of category STRUCTURE, the RTE Types Header File shall include the corresponding type declaration as:
  typedef struct <elements> <name>[<size 1>][<size 2>]...
  [<size n>];
  where <elements> is the record element specification and

<name> is the Implementation Data Type symbol of the Array Implementation Data Type.
...
...

Assuming an ImplementationDataType TypeArrayOfStruct of category ARRAY with its ImplementationDataTypeElement ArrayOfStruct_ElementType of category STRUCTURE with an ArraySize of 2. Furthermore this ImplementationDataType-Element defines additional ImplementationDataTypeElements element1_boolean, element2_sint32 of category TYPE_REFERENCE.

From this definition following should be generated in Rte_Type.h, which is IMHO correct according to SWS_Rte_06706

typedef struct
boolean element1_boolean;
sint32 element2_sint32;
TypeArrayOfStruct[2];

Let's consider a DataPrototype "myArPimOfTypeArrayOfStruct" - defined e.g. in role of ArTypedPerInstanceMemory - associated to this ImplementationDataType "TypeArrayOfStruct".

This will lead to following generation of the RTE API in the Application header file.

define Rte_Pim_myArPimOfTypeArrayOfStruct() ((P2VAR(struct
boolean element1_boolean;
sint32 element2_sint32;
, AUTOMATIC, RTE_APPL_DATA)) & Rte_ArPim_rba_Swc1_myArPimOfTypeArrayOfStruct
)

According to "[SWS_Rte_07072]" which states "If the DataPrototype is associated to an Array Implementation
Data Type the RTE API shall return an array expression (that is a pointer to the array base type) pointing to variable holding the value of the DataPrototype for which the API provides access.
..." the generated code is correct, but leads to compiler errors.

IMHO following requirements need to be updated [SWS_Rte_06706]/[SWS_Rte_06707] to enable the generation of the typedef in case the ImplementationDataTypeElement is of category STRUCTURE/UNION.

Furthermore [SWS_Rte_07072] needs to be updated and it shall be clearly stated that the "array base type" shall be the ImplementationDataTypeElement.ShortName if it is of category STRUCTURE/UNION.

Agreed solution:

1.) add new requirement [SWS_Rte_0xxx0] before [SWS_Rte_06706]
[SWS_Rte_0xxx0] d For each Array Implementation Data Type which last ImplementationDataTypeElement is of
category STRUCTURE, the RTE Types Header File shall include the corresponding type declaration as:
typedef struct <elements> <name>; c(SRS_Rte_00055, SRS_Rte_00164)

where <elements> is the record element specification and
<name> is the Implementation Data Type Element shortName of the Array Implementation Data Type.

For each record element defined by one ImplementationDataTypeElement one record element specification <elements> is defined. The record element specifications
are ordered according the order of the related ImplementationDataTypeElements in the input configuration.
Sequent record elements are separated with a semicolon.
The definition of the record element specification is defined in section 5.3.4.6.

2.) update [SWS_Rte_06706]
[SWS_Rte_06706] d For each Array Implementation Data Type which last ImplementationDataTypeElement is of
category STRUCTURE, the RTE Types Header File shall include the corresponding type declaration as:

typedef <type> <name>[<size 1>][<size 2>]...[<size n>];

where <type> is the Implementation Data Type Element shortName,
<name> is the Implementation Data Type symbol of the Array Implementation Data Type,
[<size x>] is the arraySize of the Arrays ImplementationDataTypeElement.
For each array dimension defined by one Arrays ImplementationDataTypeElement one array dimension definition [<size x>] is defined.
The array dimension definitions [<size 1>], [<size 2>] ... [<size n>] ordered from the root to the last ImplementationDataTypeElement belonging to the array definition. c(SRS_Rte_00055, SRS_Rte_00164)

3.) add new requirement [SWS_Rte_0xxx1] before [SWS_Rte_06707]

[SWS_Rte_0xxx1] d For each Array Implementation Data Type which last ImplementationDataTypeElement is of
category UNION, the RTE Types Header File shall include the corresponding type declaration as:
typedef union <elements> <name>;

where <elements> is the union element specification and
<name> is the Implementation Data Type Element shortName of the Array Implementation Data Type.

For each union element defined by one ImplementationDataTypeElement one union element specification <elements> is defined. The union element specifications
are ordered according the order of the related ImplementationDataTypeElements in the input configuration.
Sequent union elements are separated with a semicolon.
The definition of the union element specification is defined in section 5.3.4.6.

4.) update [SWS_Rte_06707]
[SWS_Rte_06707] d For each Array Implementation Data Type which last ImplementationDataTypeElement is of
category UNION, the RTE Types Header File shall include the corresponding type declaration as:

typedef <type> <name>[<size 1>][<size 2>]...[<size n>];

where <type> is the Implementation Data Type Element shortName,
<name> is the Implementation Data Type symbol of the Array Implementation Data Type,
[<size x>] is the arraySize of the Arrays ImplementationDataTypeElement.
For each array dimension defined by one Arrays ImplementationDataTypeElement one array dimension definition [<size x>] is defined.
The array dimension definitions [<size 1>], [<size 2>] ... [<size n>] ordered from the root to the last ImplementationDataTypeElement belonging to the array definition. c(SRS_Rte_00055, SRS_Rte_00164)

5.) update [SWS_Rte_07072]
[SWS_Rte_07072] d If the DataPrototype is associated to an Array Implementation Data Type the RTE API shall return an array expression (that is a pointer
to the array base type) pointing to variable holding the value of the DataPrototype for which the API provides access. If the leaf ImplementationDataTypeElement is typed by a SwBaseType the array type name shall be equal to the nativeDeclaration

attribute of the SwBaseType. If the leaf ImplementationDataTypeElement is typed by an
ImplementationDataType the type name shall be equal to the shortName of this ImplementationDataType.
If the leaf ImplementationDataTypeElement is of category STRUCTURE or UNION the type name shall be equal to
the shortName of this ImplementationDataTypeElement. c(SRS_Rte_00059)
–Last change on issue 77453 comment 2–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.54 Specification Item SWS_Rte_06707

Trace References:

SRS_Rte_00055, SRS_Rte_00164

Content:

For each Array Implementation Data Type which last ImplementationDataTypeElement is of category UNION, the RTE Types Header File shall include the corresponding type declaration as:

|typedef| |union| | <elements> | |<type>| |<name>[<size 1>][<size 2>] ... | |[<size n>]|[<size n>]|;

where |<elements>|<type>| is the record element specification and

Implementation Data Type Element shortName, |<name>| is the Implementation Data Type symbol of the Array Implementation Data Type. For each record element defined by one ImplementationDataTypeElement one record element specification |<elements>| is defined. The record element specifications are ordered according the order of the related ImplementationDataTypeElements in the input configuration. Sequent record elements are separated with a semicolon.

, |[<size x>]| is the ImplementationDataTypeElement.arraySize of the Array's ImplementationDataTypeElement. For each array dimension defined by one Array's Implementation DataTypeElement one array dimension definition |[<size x>]| is defined.

The array dimension definitions |[<size 1>], [<size 2>] ... [<size n>]| ordered from the root to the last ImplementationDataTypeElement belonging to the array definition.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77453: "[SWS_Rte_07072]" and "[SWS_Rte_06706]/[SWS_Rte_06707]" lead to uncompilable code

  Problem description:

  According to [SWS_Rte_06706] and [SWS_Rte_06707] a typedef shall be generated for each ArrayImplementationDataType which last ImplementationDataTypeElement is of category STRUCTURE,UNION.

  [SWS_Rte_06706] d For each Array Implementation Data Type which last ImplementationDataTypeElement is of category STRUCTURE, the RTE Types Header File shall include the corresponding type declaration as:
  typedef struct <elements> <name>[<size 1>][<size 2>]...
  [<size n>];
  where <elements> is the record element specification and
  <name> is the Implementation Data Type symbol of the Array Implementation Data Type.
  ...
  ...

  Assuming an ImplementationDataType TypeArrayOfStruct of category ARRAY with its ImplementationDataTypeElement ArrayOfStruct_ElementType of category STRUCTURE with an ArraySize of 2. Furthermore this ImplementationDataTypeElement defines additional ImplementationDataTypeElements element1_boolean, element2_sint32 of category TYPE_REFERENCE.

  From this definition following should be generated in Rte_Type.h, which is IMHO correct according to SWS_Rte_06706

  typedef struct
  boolean element1_boolean;
  sint32 element2_sint32;
  TypeArrayOfStruct[2];

  Let's consider a DataPrototype "myArPimOfTypeArrayOfStruct" - defined e.g. in role of ArTypedPerInstanceMemory - associated to this ImplementationDataType "TypeArrayOfStruct".

  This will lead to following generation of the RTE API in the Application header file.

  # define Rte_Pim_myArPimOfTypeArrayOfStruct() ((P2VAR(struct
  boolean element1_boolean;
  sint32 element2_sint32;

, AUTOMATIC, RTE_APPL_DATA)) & Rte_ArPim_rba_Swc1_myArPimOfTypeArrayOfStruct)

According to "[SWS_Rte_07072]" which states "If the DataPrototype is associated to an Array Implementation
Data Type the RTE API shall return an array expression (that is a pointer to the array base type) pointing to variable holding the value of the DataPrototype for which the API provides access.
..." the generated code is correct, but leads to compiler errors.

IMHO following requirements need to be updated [SWS_Rte_06706]/[SWS_Rte_06707] to enable the generation of the typedef in case the ImplementationDataTypeElement is of category STRUCTURE/UNION. Furthermore [SWS_Rte_07072] needs to be updated and it shall be clearly stated that the "array base type" shall be the ImplementationDataTypeElement.ShortName if it is of category STRUCTURE/UNION.

Agreed solution:

1.) add new requirement [SWS_Rte_0xxx0] before [SWS_Rte_06706]
[SWS_Rte_0xxx0] d For each Array Implementation Data Type which last ImplementationDataTypeElement is of
category STRUCTURE, the RTE Types Header File shall include the corresponding type declaration as:
typedef struct <elements> <name>; c(SRS_Rte_00055, SRS_Rte_00164)

where <elements> is the record element specification and
<name> is the Implementation Data Type Element shortName of the Array Implementation Data Type.

For each record element defined by one ImplementationDataTypeElement one record element specification <elements> is defined. The record element specifications
are ordered according the order of the related ImplementationDataTypeElements in the input configuration.
Sequent record elements are separated with a semicolon.
The definition of the record element specification is defined in section 5.3.4.6.

2.) update [SWS_Rte_06706]
[SWS_Rte_06706] d For each Array Implementation Data Type which last ImplementationDataTypeElement is of
category STRUCTURE, the RTE Types Header File shall include the corresponding type declaration as:

typedef <type> <name>[<size 1>][<size 2>]...[<size n>];

where <type> is the Implementation Data Type Element shortName,
<name> is the Implementation Data Type symbol of the Array Implementation Data Type,
[<size x>] is the arraySize of the Arrays ImplementationDataTypeElement.
For each array dimension defined by one Arrays ImplementationDataTypeElement one array dimension definition [<size x>] is defined.
The array dimension definitions [<size 1>], [<size 2>] ... [<size n>] ordered from the root to the last ImplementationDataTypeElement belonging to the array definition. c(SRS_Rte_00055, SRS_Rte_00164)

3.) add new requirement [SWS_Rte_0xxx1] before [SWS_Rte_06707]
[SWS_Rte_0xxx1] d For each Array Implementation Data Type which last ImplementationDataTypeElement is of
category UNION, the RTE Types Header File shall include the corresponding type declaration as:
typedef union <elements> <name>;

where <elements> is the union element specification and
<name> is the Implementation Data Type Element shortName of the Array Implementation Data Type.

For each union element defined by one ImplementationDataTypeElement one union element specification <elements> is defined. The union element specifications
are ordered according the order of the related ImplementationDataTypeElements in the input configuration.
Sequent union elements are separated with a semicolon.
The definition of the union element specification is defined in section 5.3.4.6.

4.) update [SWS_Rte_06707]
[SWS_Rte_06707] d For each Array Implementation Data Type which last ImplementationDataTypeElement is of
category UNION, the RTE Types Header File shall include the corresponding type declaration as:

typedef <type> <name>[<size 1>][<size 2>]...[<size n>];

where <type> is the Implementation Data Type Element shortName,
<name> is the Implementation Data Type symbol of the Array Implementation Data Type,

[<size x>] is the arraySize of the Arrays ImplementationDataTypeElement.
For each array dimension defined by one Arrays ImplementationDataTypeElement one array dimension definition [<size x>] is defined.
The array dimension definitions [<size 1>], [<size 2>] ... [<size n>] ordered from the root to the last ImplementationDataTypeElement belonging to the array definition. c(SRS_Rte_00055, SRS_Rte_00164)

5.) update [SWS_Rte_07072]
[SWS_Rte_07072] d If the DataPrototype is associated to an Array Implementation Data Type the RTE API shall return an array expression (that is a pointer to the array base type) pointing to variable holding the value of the DataPrototype for which the API provides access. If the leaf ImplementationDataTypeElement is typed by a SwBaseType the array type name shall be equal to the nativeDeclaration attribute of the SwBaseType. If the leaf ImplementationDataTypeElement is typed by an ImplementationDataType the type name shall be equal to the shortName of this ImplementationDataType.
If the leaf ImplementationDataTypeElement is of category STRUCTURE or UNION the type name shall be equal to the shortName of this ImplementationDataTypeElement. c(SRS_Rte_00059)
–Last change on issue 77453 comment 2–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.55 Specification Item SWS_Rte_06711

Trace References:

none

Content:

The RTE generator shall reject configurations where the value of the attribute ImplementationDataType.typeEmitter is not defined or set to "|RTE|", and the ImplementationDataType references a SwBaseType without defined BaseTypeDirectDefinition.nativeDeclaration.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76816: SwBaseType without defined nativeDeclaration shall be rejected also for no typeEmitter

  Problem description:

  Name: Robert Sakretz
  Phone:
  Role: WP-M

  Description/Motivation:
  [SWS_Rte_06711] states: "The RTE generator shall reject configurations where the value of the attribute typeEmitter is set to "RTE" and the ImplementationDataType references a SwBaseType without defined nativeDeclaration."

  but according to [SWS_Rte_06709] it is equivalent to not define a typeEmitter. This case is not covered in [SWS_Rte_06711].

  Agreed solution:

  Update [SWS_Rte_06711] to
  The RTE generator shall reject configurations where the attribute typeEmitter is not defined or set to "RTE",
  and the ImplementationDataType references a SwBaseType without defined nativeDeclaration.
  –Last change on issue 76816 comment 2–

  BW-C-Level:

Application	Specification	Bus
1	4	1

1.56 Specification Item SWS_Rte_06828

Trace References:

SRS_Rte_00094

Content:

|RTE_E_COM_STOPPED| – The the RTE could not perform the operation because the COM communication service is currently not available (inter ECU communication only). RTE shall return |RTE_E_COM_STOPPED| whenthe corresponding COM :

- in case of COM the corresponding service returns |COM_SERVICE_NOT_AVAILABLE| .

- in case of LdCom the corresponding LdCom_Transmit returns |E_NOT_OK|

In case of stopped |I-PDUS| the last known value (or init value) is given back as data by the according Rte_IRead API.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76094: [Rte][LdCom] Reaction on E_NOT_OK when calling LdCom_Transmit

  Problem description:

  It is specified, that Rte_Write and Rte_Send shall return RTE_E_OK when
  "data passed to communication service _successfully_"
  But what shall happen if LdCom_Transmit returns E_NOT_OK ?

  Agreed solution:

  On behalf of WP-A1:

  Rephrase
  "[SWS_Rte_07822] RTE_E_COM_STOPPED the RTE could
  not perform the operation because the communication service is currently
  not available (inter ECU communication only). RTE shall return
  RTE_E_COM_STOPPED when:
  -in case of COM the corresponding COM service returns COM_SERVICE_NOT_AVAILABLE
  -in case of LdCom the corresponding LdCom_Transmit returns E_NOT_OK
  ."

  Do the same for SWS_Rte_07823, SWS_Rte_01106, SWS_Rte_06828 and SWS_Rte_01339

  SWS_Rte_01379: replace RTE_E_OK with E_OK.
  –Last change on issue 76094 comment 4–

  BW-C-Level:

Application	Specification	Bus
1	1	1

1.57 Specification Item SWS_Rte_07054

Trace References:

SRS_BSW_00327

Content:

Symbolic name: |SCHM_E_TIMEOUT|
Value: 129
Comments: The configured timeout exceeds before the intended result was ready. Note: The value has to be identical with SWS_Rte_01064

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77161: Tables in specification documents contain trace items

  Problem description:

  _____

  Name: Wolf-Hendrik Kaps
  Phone: 0711-811-23288
  Role: Jg-Tooling member
  _____

  Description/Motivation:

  Some specification documents contain tables which include trace items. E.g. SWS_Com, table in 7.12.1 Development Errors, SWS_Rte, Table 5.4: RTE Error and Status values.

  As discussed in tooling session we should extend ValidateARXML routine to elicit trace items inside tables.

  Further on we shall ensure that tracebles do not contain figures and tables (77206, 74860)
  –Last change on issue 77161 comment 3–

  Agreed solution:

  1. Affected Documents
  ==========

  1. Move the Traceable out of the f**ing tables (see attachment)
  2. move Tables and figures out of the Traceable

  2. Word2arxml and latex2arxml which is used by checkDocumentSource
  ==========

  let these scripts complain also requested by 77206, 74860 but summarized

here

* Tables in Traceable
* Traceable in Tables
* Figures in Traceable

2.1 GST: add these contraints
==========

3. CP_Tool_Scripts
==========

* let latexinstatiaator complain about Figures in Traceable
* note: tha table in traceable violates the schema and is flagged already
* let latexinstantiator complain about traceable in tables
–Last change on issue 77161 comment 7–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.58 Specification Item SWS_Rte_07072

Trace References:

SRS_Rte_00059

Content:

If the DataPrototype is associated to an Array Implementation Data Type the RTE API shall return an array expression (that is a pointer to the array base type) pointing to variable holding the value of the DataPrototype for which the API provides access. If the leaf ImplementationDataTypeElement is typed by a SwBaseType the array type name shall be equal to the BaseTypeDirectDefinition.nativeDeclaration attribute of the SwBaseType. If the leaf ImplementationDataTypeElement is typed by an ImplementationDataType the type name shall be equal to the Referrable.shortName of these this ImplementationDataType. If the leaf ImplementationDataTypeElement is of category |STRUCTURE| or |UNION| the type name shall be equal to the Referrable.shortName of this ImplementationDataType Element.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77453: "[SWS_Rte_07072]" and "[SWS_Rte_06706]/[SWS_Rte_06707]" lead to uncompilable code

  Problem description:

  According to [SWS_Rte_06706] and [SWS_Rte_06707] a typedef shall be generated for each ArrayImplementationDataType which last ImplementationDataTypeElement is of category STRUCTURE,UNION.

  [SWS_Rte_06706] d For each Array Implementation Data Type which last ImplementationDataTypeElement is of category STRUCTURE, the RTE Types Header File shall include the corresponding type declaration as:
  typedef struct <elements> <name>[<size 1>][<size 2>]...
  [<size n>];
  where <elements> is the record element specification and
  <name> is the Implementation Data Type symbol of the Array Implementation Data Type.
  ...
  ...

  Assuming an ImplementationDataType TypeArrayOfStruct of category ARRAY with its ImplementationDataTypeElement ArrayOfStruct_ElementType of category STRUCTURE with an ArraySize of 2. Furthermore this ImplementationDataTypeElement defines additional ImplementationDataTypeElements element1_boolean, element2_sint32 of category TYPE_REFERENCE.

  From this definition following should be generated in Rte_Type.h, which is IMHO correct according to SWS_Rte_06706

  typedef struct
  boolean element1_boolean;
  sint32 element2_sint32;
  TypeArrayOfStruct[2];

  Let's consider a DataPrototype "myArPimOfTypeArrayOfStruct" - defined e.g. in role of ArTypedPerInstanceMemory - associated to this ImplementationDataType "TypeArrayOfStruct".

  This will lead to following generation of the RTE API in the Application header file.

  # define Rte_Pim_myArPimOfTypeArrayOfStruct() ((P2VAR(struct
  boolean element1_boolean;
  sint32 element2_sint32;

, AUTOMATIC, RTE_APPL_DATA)) & Rte_ArPim_rba_Swc1_myArPimOfTypeArrayOfStruct)

According to "[SWS_Rte_07072]" which states "If the DataPrototype is associated to an Array Implementation
Data Type the RTE API shall return an array expression (that is a pointer to the array base type) pointing to variable holding the value of the DataPrototype for which the API provides access.
..." the generated code is correct, but leads to compiler errors.

IMHO following requirements need to be updated [SWS_Rte_06706]/[SWS_Rte_06707] to enable the generation of the typedef in case the ImplementationDataTypeElement is of category STRUCTURE/UNION. Furthermore [SWS_Rte_07072] needs to be updated and it shall be clearly stated that the "array base type" shall be the ImplementationDataTypeElement.ShortName if it is of category STRUCTURE/UNION.

Agreed solution:

1.) add new requirement [SWS_Rte_0xxx0] before [SWS_Rte_06706]
[SWS_Rte_0xxx0] d For each Array Implementation Data Type which last ImplementationDataTypeElement is of
category STRUCTURE, the RTE Types Header File shall include the corresponding type declaration as:
typedef struct <elements> <name>; c(SRS_Rte_00055, SRS_Rte_00164)

where <elements> is the record element specification and
<name> is the Implementation Data Type Element shortName of the Array Implementation Data Type.

For each record element defined by one ImplementationDataTypeElement one record element specification <elements> is defined. The record element specifications
are ordered according the order of the related ImplementationDataTypeElements in the input configuration.
Sequent record elements are separated with a semicolon.
The definition of the record element specification is defined in section 5.3.4.6.

2.) update [SWS_Rte_06706]
[SWS_Rte_06706] d For each Array Implementation Data Type which last ImplementationDataTypeElement is of
category STRUCTURE, the RTE Types Header File shall include the corresponding type declaration as:

typedef <type> <name>[<size 1>][<size 2>]...[<size n>];

where <type> is the Implementation Data Type Element shortName,
<name> is the Implementation Data Type symbol of the Array Implementation Data Type,
[<size x>] is the arraySize of the Arrays ImplementationDataTypeElement.
For each array dimension defined by one Arrays ImplementationDataTypeElement one array dimension definition [<size x>] is defined.
The array dimension definitions [<size 1>], [<size 2>] ... [<size n>] ordered from the root to the last ImplementationDataTypeElement belonging to the array definition. c(SRS_Rte_00055, SRS_Rte_00164)

3.) add new requirement [SWS_Rte_0xxx1] before [SWS_Rte_06707]
[SWS_Rte_0xxx1] d For each Array Implementation Data Type which last ImplementationDataTypeElement is of
category UNION, the RTE Types Header File shall include the corresponding type declaration as:
typedef union <elements> <name>;

where <elements> is the union element specification and
<name> is the Implementation Data Type Element shortName of the Array Implementation Data Type.

For each union element defined by one ImplementationDataTypeElement one union element specification <elements> is defined. The union element specifications
are ordered according the order of the related ImplementationDataTypeElements in the input configuration.
Sequent union elements are separated with a semicolon.
The definition of the union element specification is defined in section 5.3.4.6.

4.) update [SWS_Rte_06707]
[SWS_Rte_06707] d For each Array Implementation Data Type which last ImplementationDataTypeElement is of
category UNION, the RTE Types Header File shall include the corresponding type declaration as:

typedef <type> <name>[<size 1>][<size 2>]...[<size n>];

where <type> is the Implementation Data Type Element shortName,
<name> is the Implementation Data Type symbol of the Array Implementation Data Type,

[<size x>] is the arraySize of the Arrays ImplementationDataTypeElement.
For each array dimension defined by one Arrays ImplementationDataTypeElement one array dimension definition [<size x>] is defined.
The array dimension definitions [<size 1>], [<size 2>] ... [<size n>] ordered from the root to the last ImplementationDataTypeElement belonging to the array definition. c(SRS_Rte_00055, SRS_Rte_00164)

5.) update [SWS_Rte_07072]
[SWS_Rte_07072] d If the DataPrototype is associated to an Array Implementation Data Type the RTE API shall return an array expression (that is a pointer to the array base type) pointing to variable holding the value of the DataPrototype for which the API provides access. If the leaf ImplementationDataTypeElement is typed by a SwBaseType the array type name shall be equal to the nativeDeclaration attribute of the SwBaseType. If the leaf ImplementationDataTypeElement is typed by an ImplementationDataType the type name shall be equal to the shortName of this ImplementationDataType.
If the leaf ImplementationDataTypeElement is of category STRUCTURE or UNION the type name shall be equal to the shortName of this ImplementationDataTypeElement. c(SRS_Rte_00059)
–Last change on issue 77453 comment 2–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.59 Specification Item SWS_Rte_07087

Trace References:

SRS_Rte_00235

Content:

The RTE shall support the queuing of triggers for

- External Trigger Event Communication
- Inter Runnable Triggering
- Inter Basic Software Module Entity Triggering

if the RteInternalTriggerConfig.RteTriggerSourceQueueLength / RteBswInternalTrigger Config.RteBswTriggerSourceQueueLength is configured > 0. 0, regardless of the value of the attribute swImplPolicy of the trigger entity.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #74201: Clarification of trigger swImplPolicy vs. Rte[Bsw]TriggerSourceQueueLength (SWS_Rte_07087)

  Problem description:

  [constr_1169] clearly says if the Trigger.swImplPolicy is
  - STANDARD: "Trigger processing does not use a queue"
  - QUEUED: "processing of Triggers positively uses a queue"

  Clarification: how [SWS_Rte_07087] shall be interpreted with respect to [constr_1169]. Especially the explanatory text below [SWS_Rte_07087] ("it is possible to configure a trigger queue regardless on the value of the attribute swImplPolicy") sounds like the integrator can override the queued or non-queued behavior of the trigger.swImplPolicy by configuring the Rte[Bsw]TriggerSourceQueueLength to a value greater zero resp. to zero. If this is the intended use case then please clarify what shall happen if the integrator configured a zero queue length for a queued trigger resp. a non-zero queue length for a non-queued trigger.

  Other snippets to consider from the Rte spec:

  - Description of parameter RteTriggerSourceQueueLength: "A value greater or equal to 1 requests an queued behavior. Setting the value of RteTrigger-SourceQueueLength to 0 requests an none queued implementation of the trigger communication."

  - [SWS_Rte_07200]: "The signature for queuing support shall be generated by the RTE generator if the swImplPolicy of the associated Trigger is set to queued."
  –Last change on issue 74201 comment 11–

  Agreed solution:

  Rephrase the requirement[SWS_Rte_07087] as follows:

  The RTE shall implement the queuing of triggers for
  -External Trigger Event Communication
  -Inter Runnable Triggering
  -Inter Basic Software Module Entity Triggering

  if the RteTriggerSourceQueueLength / RteBswTriggerSourceQueueLength

is configured > 0, regardless of the value of the attribute swImplPolicy of the trigger entity. c(SRS_Rte_00235)

SWCT
====

Add note below constr_1169:

Please note that the value of Trigger.swImplPolicy is not final word on the implementation of a queue for the specific Trigger. The integrator still has the power to overrule the application software developer's verdict if applicable.
–Last change on issue 74201 comment 4–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.60 Specification Item SWS_Rte_07132

Trace References:

SRS_Rte_00011, SRS_Rte_00167

Content:

The component data structure type shall be defined in the Application Header file.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76346: Component data structure type definition vs. structure declaration

  Problem description:

  In general the RTE SWS distinguishes between a structure declaration and a type declaration for a composite datatype that represents a structure.

  For a structure implementation data type SWS_Rte_07114 requires a structure declaration:

  struct _<name>_
  ...
  ;

  SWS_Rte_06812 requires the type declaration:

typedef struct _<name>_ <name>;

Regarding the component data structure currently there exist the following requirements only:

[SWS_Rte_07132] The component data structure type shall be defined in the Application Header file.
[SWS_Rte_03714] The type name of the component data structure shall be Rte_[Byps]_CDS_<cts> where <cts> is the component type symbol of the AtomicSwComponentType. ...

Since SWS_Rte_03714 talks about the type name of the component data structure this requires a typedef statement for Rte_[Byps]_CDS_<cts> in the Application Header file:

typedef struct ... Rte_[Byps]_CDS_<cts>;

Currently there does not exist any requirement for a CDS structure declaration.

The code examples in the SWS RTE are somehow in contradiction to these requirements. They use "struct

Rte_CDS_<cts>" instead of the typedef name. See examples 5.24, 5.26, 5.28.

This inconsistency applies to the signature of some APIs, too:

[SWS_Rte_01238]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Start([const struct Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01239]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Return([const struct Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01248]
void Rte_[<client>_]Runnable_<cts>_<reName>_Start([const RTE_CDS_<cts>*])

[SWS_Rte_01249]
void Rte_[<client>_]Runnable_<cts>_<reName>_Return([const Rte_CDS_<cts>*])

[SWS_Rte_06113]

void Rte_[<client>_]RptHook_<cts>_<var>_Transmit([const RTE_CDS_<cts>*], <type>* <buffer>)

[SWS_Rte_06114]
void Rte_[<client>_]RptHook_<cts>_<var>_Reception([const RTE_CDS_<cts>*], <type>* <buffer>)

In addition some of these APIs use RTE_CDS_ instead of Rte_CDS_.

Agreed solution:

Delete SWS_Rte_07132 and SWS_Rte_03714.

Add new requirements:

[SWS_Rte_xxxx1] The Application Header File shall include a structure declaration for the component data structure as follows:
struct Rte_[Byps_]CDS_<cts> <component data sections> ;
where <cts> is the component type symbol of the AtomicSwComponentType. [Byps_] is an optional infix used when component wrapper method for bypass support is enabled for the related software componenttype (See chapter 4.9.2).

[SWS_Rte_xxxx2] The Application Header File shall include a type declaration for the component data structure type as follows:
typedef struct Rte_[Byps_]CDS_<cts> Rte_[Byps_]CDS_<cts>;
where <cts> is the component type symbol of the AtomicSwComponentType.[Byps_] is an optional infix used when component wrapper method for bypass support is enabled for the related software componenttype (See chapter 4.9.2).

Change the API signatures to:

[SWS_Rte_01238]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Start([const Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01239]
void Rte_[<client>_]<api>Hook_<cts>_<ap>_Return([const Rte_CDS_<cts>*,]<param>)

[SWS_Rte_01248]
void Rte_[<client>_]Runnable_<cts>_<reName>_Start([const Rte_CDS_<cts>*])

[SWS_Rte_06113]
void Rte_[<client>_]RptHook_<cts>_<var>_Transmit([const Rte_CDS_<cts>*],

<type>* <buffer>)

[SWS_Rte_06114]
void Rte_[<client>_]RptHook_<cts>_<var>_Reception([const Rte_CDS_<cts>*], <type>* <buffer>)

Change examples 5.24, 5.26, 5.28 so that they contain both structure declaration and typedef for component data structure type and use the type (without struct) at the interfaces.
–Last change on issue 76346 comment 1–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.61 Specification Item SWS_Rte_07175

Trace References:

SRS_Rte_00018

Content:

The RTE generator shall reject configurations violating the constr_1133. 1434.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76871: [constr_1434] introduced in R4.3.0 is too restrictive

  Problem description:

  After the reformulation of the constr_1434 due to RFC72773 the [constr_1434] is too restrictive since now the BITFILED_TEXTTABLE CompuMethods provided by AUOTSAR are violating this constraint.

  Those BITFILED_TEXTTABLE CompuMethods may defining multiple "boolean" kind values where for all of them FALSE is 0.
  The different TRUE values are covered by the shortLables which in turn are listed in the BSW Specification.

  Example:
  <COMPU-METHOD>
  <SHORT-NAME>Dem_MonitorStatusType</SHORT-NAME>
  <CATEGORY>BITFIELD_TEXTTABLE</CATEGORY>

```
<COMPU-INTERNAL-TO-PHYS>
<COMPU-SCALES>
<COMPU-SCALE>
<SHORT-LABEL>DEM_MONITOR_STATUS_TF</SHORT-LABEL>
<DESC>
<L-2 L="EN">Bit0: TestFailed</L-2>
</DESC>
<MASK>0x01</MASK>
<LOWER-LIMIT INTERVAL-TYPE="CLOSED">0</LOWER-LIMIT>
<UPPER-LIMIT INTERVAL-TYPE="CLOSED">0</UPPER-LIMIT>
<COMPU-CONST>
<VT>FALSE</VT>
</COMPU-CONST>
</COMPU-SCALE>
<COMPU-SCALE>
<SHORT-LABEL>DEM_MONITOR_STATUS_TF</SHORT-LABEL>
<SYMBOL>DEM_MONITOR_STATUS_TF</SYMBOL>
<DESC>
<L-2 L="EN">Bit0: TestFailed</L-2>
</DESC>
<MASK>0x01</MASK>
<LOWER-LIMIT INTERVAL-TYPE="CLOSED">1</LOWER-LIMIT>
<UPPER-LIMIT INTERVAL-TYPE="CLOSED">1</UPPER-LIMIT>
<COMPU-CONST>
<VT>TRUE</VT>
</COMPU-CONST>
</COMPU-SCALE>
<COMPU-SCALE>
<SHORT-LABEL>DEM_MONITOR_STATUS_TNCTOC</SHORT-LABEL>
<DESC>
<L-2 L="EN">Bit1: TestNotCompletedThisOperationCycle</L-2>
</DESC>
<MASK>0x02</MASK>
<LOWER-LIMIT INTERVAL-TYPE="CLOSED">0</LOWER-LIMIT>
<UPPER-LIMIT INTERVAL-TYPE="CLOSED">0</UPPER-LIMIT>
<COMPU-CONST>
<VT>FALSE</VT>
</COMPU-CONST>
</COMPU-SCALE>
<COMPU-SCALE>
<SHORT-LABEL>DEM_MONITOR_STATUS_TNCTOC</SHORT-LABEL>
<SYMBOL>DEM_MONITOR_STATUS_TNCTOC</SYMBOL>
<DESC>
```

<L-2 L="EN">Bit1: TestNotCompletedThisOperationCycle</L-2>
</DESC>
<MASK>0x02</MASK>
<LOWER-LIMIT INTERVAL-TYPE="CLOSED">2</LOWER-LIMIT>
<UPPER-LIMIT INTERVAL-TYPE="CLOSED">2</UPPER-LIMIT>
<COMPU-CONST>
<VT>TRUE</VT>
</COMPU-CONST>
</COMPU-SCALE>
</COMPU-SCALES>
</COMPU-INTERNAL-TO-PHYS>
</COMPU-METHOD>

Info - constr_1434 versus the removed constr_1133:

[constr_1434] CompuScales shall not have identical CompuScale Value Symbolic
Names d In a CompuMethod that is subject to [constr_1146], no two CompuScales shall have identical CompuScale Value Symbolic Names (according to [TPS_SWCT_01696]). c()

[constr_1133] Identical CompuScale Symbolic Names shall have the same
range d In a CompuMethod that is subject to [constr_1146], all CompuScales that yield identical CompuScale Symbolic Names shall have the same range defined by CompuScale.lowerLimit and CompuScale.upperLimit. c()

Additionally the SWS RTE might need a clean-up dependent from our conclusion since the SWS_Rte_07175 refers still to constr_1133.

**
[SWS_Rte_07175] d The RTE generator shall reject configurations violating the [constr_
1133]. c(SRS_Rte_00018)
This rejects configurations where an ImplementationDataType or
an ApplicationPrimitiveDataType references a CompuMethod
which is of category "TEXTTABLE", "SCALE_LINEAR_AND_TEXTTABLE",
"SCALE_RATIONAL_AND_TEXTTABLE", or BITFIELD_TEXTTABLE and has CompuScales with identical CompuScale symbolic names but different CompuScale.
lowerLimit or CompuScale.upperLimit.
Note that there might exist additional CompuScales with non-point ranges inside

a CompuMethod of category "TEXTTABLE", "SCALE_LINEAR_AND_TEXTTABLE", "SCALE_RATIONAL_AND_TEXTTABLE", or BITFIELD_TEXTTABLE , but for those no
enumeration literals are generated by the RTE generator.
**
–Last change on issue 76871 comment 3–

Agreed solution:

SWS_RTE
——-
Update [SWS_Rte_07175]:
[SWS_Rte_07175] d The RTE generator shall reject configurations violating the [constr_1434]. c(SRS_Rte_00018)

Rephase text below [SWS_Rte_07175]:
This rejects configurations where an ImplementationDataType or an ApplicationPrimitiveDataType references a CompuMethod which is of category "TEXTTABLE", "SCALE_LINEAR_AND_TEXTTABLE", "SCALE_RATIONAL_AND_TEXTTABLE", or BITFIELD_TEXTTABLE and has CompuScales with identical CompuScale Value Symbolic Names.

MOD_GeneralBlueprints
———————
add a symbol to the CompuScales with the FALSE value:
symbol is the symbol of the according TRUE case + "__FALSE"

MMT
—
Let MMT generate the new symbols for FALSE values in bitfield-CompuMethods in the bswtypes-generator as described above in the MOD_GeneralBlueprints section
–Last change on issue 76871 comment 12–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.62 Specification Item SWS_Rte_07228

Trace References:

SRS_Rte_00051, SRS_Rte_00163

Content:

Entries in the Inter Runnable Triggering handles section shall be sorted alphabetically (ASCII / ISO 8859-1 code in ascending order).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76347: Explicit specification of alphabetical ordering

  Problem description:

  Requirements SWS_Rte_06061, SWS_Rte_03812, and SWS_Rte_07228 do not mention the exact criteria for alphabetical ordering.

  Agreed solution:

  Requirements SWS_Rte_06061, SWS_Rte_03812, and SWS_Rte_07228:
  Add the parenthesis phrase "(ASCII / ISO 8859-1 code in ascending order)" after "alphabetic ordering".
  –Last change on issue 76347 comment 1–

  BW-C-Level:

Application	Specification	Bus
1	4	1

1.63 Specification Item SWS_Rte_07289

Trace References:

SRS_BSW_00327

Content:

Symbolic name: |SCHM_E_OK|
Value: 0
Comments: No error occurred.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77161: Tables in specification documents contain trace items

  Problem description:

  Name: Wolf-Hendrik Kaps

Phone: 0711-811-23288
Role: Jg-Tooling member

Description/Motivation:

Some specification documents contain tables which include trace items. E.g. SWS_Com, table in 7.12.1 Development Errors, SWS_Rte, Table 5.4: RTE Error and Status values.

As discussed in tooling session we should extend ValidateARXML routine to elicit trace items inside tables.

Further on we shall ensure that tracebles do not contain figures and tables (77206, 74860)
–Last change on issue 77161 comment 3–

Agreed solution:

1. Affected Documents
==========

1. Move the Traceable out of the f**ing tables (see attachment)
2. move Tables and figures out of the Traceable

2. Word2arxml and latex2arxml which is used by checkDocumentSource
==========

let these scripts complain also requested by 77206, 74860 but summarized here

* Tables in Traceable
* Traceable in Tables
* Figures in Traceable

2.1 GST: add these contraints
==========

3. CP_Tool_Scripts
==========

* let latexinstatiaator complain about Figures in Traceable
* note: tha table in traceable violates the schema and is flagged already

* let latexinstantiator complain about traceable in tables
–Last change on issue 77161 comment 7–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.64 Specification Item SWS_Rte_07290

Trace References:

SRS_BSW_00327

Content:

Symbolic name: |SCHM_E_LIMIT|
Value: 130
Comments: A internal Basic Software Scheduler limit has been exceeded. Request could not be handled. OUT buffers are not modified.

Note: The value has to be identical with SWS_Rte_01317

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77161: Tables in specification documents contain trace items

  Problem description:

  ———————————————
  Name: Wolf-Hendrik Kaps
  Phone: 0711-811-23288
  Role: Jg-Tooling member
  ———————————————
  Description/Motivation:

  Some specification documents contain tables which include trace items. E.g. SWS_Com, table in 7.12.1 Development Errors, SWS_Rte, Table 5.4: RTE Error and Status values.

  As discussed in tooling session we should extend ValidateARXML routine to elicit trace items inside tables.

  Further on we shall ensure that tracebles do not contain figures and tables

(77206, 74860)
–Last change on issue 77161 comment 3–

Agreed solution:

1. Affected Documents
==========

1. Move the Traceable out of the f**ing tables (see attachment)
2. move Tables and figures out of the Traceable

2. Word2arxml and latex2arxml which is used by checkDocumentSource
==========

let these scripts complain also requested by 77206, 74860 but summarized here

* Tables in Traceable
* Traceable in Tables
* Figures in Traceable

2.1 GST: add these contraints
==========

3. CP_Tool_Scripts
==========

* let latexinstatiaator complain about Figures in Traceable
* note: tha table in traceable violates the schema and is flagged already
* let latexinstantiator complain about traceable in tables
–Last change on issue 77161 comment 7–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.65 Specification Item SWS_Rte_07384

Trace References:

SRS_BSW_00327, SRS_Rte_00184

Content:

Symbolic name: |RTE_E_NEVER_RECEIVED|
Value: 133
Comments: No data received for the corresponding unqueued data element since system start or partition restart.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77161: Tables in specification documents contain trace items

  Problem description:

  _____

  Name: Wolf-Hendrik Kaps
  Phone: 0711-811-23288
  Role: Jg-Tooling member
  _____

  Description/Motivation:

  Some specification documents contain tables which include trace items. E.g. SWS_Com, table in 7.12.1 Development Errors, SWS_Rte, Table 5.4: RTE Error and Status values.

  As discussed in tooling session we should extend ValidateARXML routine to elicit trace items inside tables.

  Further on we shall ensure that tracebles do not contain figures and tables (77206, 74860)
  –Last change on issue 77161 comment 3–

  Agreed solution:

  1. Affected Documents
  ==========

  1. Move the Traceable out of the f**ing tables (see attachment)
  2. move Tables and figures out of the Traceable

  2. Word2arxml and latex2arxml which is used by checkDocumentSource
  ==========

  let these scripts complain also requested by 77206, 74860 but summarized

here

* Tables in Traceable
* Traceable in Tables
* Figures in Traceable

2.1 GST: add these contraints
==========

3. CP_Tool_Scripts
==========

* let latexinstatiaator complain about Figures in Traceable
* note: tha table in traceable violates the schema and is flagged already
* let latexinstantiator complain about traceable in tables
–Last change on issue 77161 comment 7–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.66 Specification Item SWS_Rte_07562

Trace References:

SRS_BSW_00327

Content:

Symbolic name: |SCHM_E_NO_DATA|
Value: 131
Comments: An explicit read API call returned no data. (This is no error.)

Note: The value has to be identical with SWS_Rte_01061

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77161: Tables in specification documents contain trace items

  Problem description:

  Name: Wolf-Hendrik Kaps
  Phone: 0711-811-23288

Role: Jg-Tooling member

Description/Motivation:

Some specification documents contain tables which include trace items. E.g. SWS_Com, table in 7.12.1 Development Errors, SWS_Rte, Table 5.4: RTE Error and Status values.

As discussed in tooling session we should extend ValidateARXML routine to elicit trace items inside tables.

Further on we shall ensure that tracebles do not contain figures and tables (77206, 74860)
–Last change on issue 77161 comment 3–

Agreed solution:

1. Affected Documents
==========

1. Move the Traceable out of the f**ing tables (see attachment)
2. move Tables and figures out of the Traceable

2. Word2arxml and latex2arxml which is used by checkDocumentSource
==========

let these scripts complain also requested by 77206, 74860 but summarized here

* Tables in Traceable
* Traceable in Tables
* Figures in Traceable

2.1 GST: add these contraints
==========

3. CP_Tool_Scripts
==========

* let latexinstatiaator complain about Figures in Traceable
* note: tha table in traceable violates the schema and is flagged already

* let latexinstantiator complain about traceable in tables
–Last change on issue 77161 comment 7–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.67 Specification Item SWS_Rte_07563

Trace References:

SRS_BSW_00327

Content:

Symbolic name: |SCHM_E_TRANSMIT_ACK|
Value: 132
Comments: Transmission acknowledgement received.

Note: The value has to be identically with SWS_Rte_01065

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77161: Tables in specification documents contain trace items

  Problem description:

  ———————————
  Name: Wolf-Hendrik Kaps
  Phone: 0711-811-23288
  Role: Jg-Tooling member
  ———————————
  Description/Motivation:

  Some specification documents contain tables which include trace items. E.g. SWS_Com, table in 7.12.1 Development Errors, SWS_Rte, Table 5.4: RTE Error and Status values.

  As discussed in tooling session we should extend ValidateARXML routine to elicit trace items inside tables.

  Further on we shall ensure that tracebles do not contain figures and tables (77206, 74860)
  –Last change on issue 77161 comment 3–

Agreed solution:

1. Affected Documents
==========

1. Move the Traceable out of the f**ing tables (see attachment)
2. move Tables and figures out of the Traceable

2. Word2arxml and latex2arxml which is used by checkDocumentSource
==========

let these scripts complain also requested by 77206, 74860 but summarized here

* Tables in Traceable
* Traceable in Tables
* Figures in Traceable

2.1 GST: add these contraints
==========

3. CP_Tool_Scripts
==========

* let latexinstatiaator complain about Figures in Traceable
* note: tha table in traceable violates the schema and is flagged already
* let latexinstantiator complain about traceable in tables
–Last change on issue 77161 comment 7–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.68 Specification Item SWS_Rte_07655

Trace References:

SRS_BSW_00327, SRS_Rte_00139, SRS_Rte_00200

Content:

Symbolic name: |RTE_E_UNCONNECTED|
Value: 134
Comments: The port used for communication is not connected.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77161: Tables in specification documents contain trace items

  Problem description:

  _____
  Name: Wolf-Hendrik Kaps
  Phone: 0711-811-23288
  Role: Jg-Tooling member
  _____
  Description/Motivation:

  Some specification documents contain tables which include trace items. E.g. SWS_Com, table in 7.12.1 Development Errors, SWS_Rte, Table 5.4: RTE Error and Status values.

  As discussed in tooling session we should extend ValidateARXML routine to elicit trace items inside tables.

  Further on we shall ensure that tracebles do not contain figures and tables (77206, 74860)
  –Last change on issue 77161 comment 3–

  Agreed solution:

  1. Affected Documents
  ==========

  1. Move the Traceable out of the f**ing tables (see attachment)
  2. move Tables and figures out of the Traceable

  2. Word2arxml and latex2arxml which is used by checkDocumentSource
  ==========

  let these scripts complain also requested by 77206, 74860 but summarized here

  * Tables in Traceable

* Traceable in Tables
* Figures in Traceable

2.1 GST: add these contraints
==========

3. CP_Tool_Scripts
==========

* let latexinstatiaator complain about Figures in Traceable
* note: tha table in traceable violates the schema and is flagged already
* let latexinstantiator complain about traceable in tables
–Last change on issue 77161 comment 7–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.69 Specification Item SWS_Rte_07676

Trace References:

SRS_BSW_00337

Content:

Default errors shall be reported to the DET if and only if RteGeneration.RteDevErrorDetect is enabled.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #73570: No "default error" in AUTOSAR

  Problem description:

  The DET was renamed from development error tracer to default error tracer.

  This change was most of the time done automatically and unfortunately renamed "developement error" to "default error".

  "default error" should always be followed by "tracer", otherwise, "development error" is probably the right term.

  This could increase the impact (compared to my selection of impacted docu-

ment, but formally, the configuration parameters *DevErrorDetect are not using the correct description:
"Switches the Default Error Tracer (Det) detection and notification..."

The parameter switches on/off the developement error detection. The DET does not need to be detected and can be present even when the parameter is set to false.

Agreed solution:

Rename "default error" to "development error" in all impacted documents, but not in an automated way (Do not change "default error tracer" to "developement error tracer"!)

Blueprint/Example:
- sub chapter is now called "7.x Default errors"
- "[SWS_xxx_yyyyy]
In case default error detection is enabled for the xxxx module: The xxxx module shall check API parameters for validity and report detected errors to the DET. ()"
- "[SWS_xxx_yyyyy]
If default error detection is enabled: the function shall check that the service xxx_Init was previously called. If the check fails, the function shall raise the default error XXX_E_NOT_INITIALIZED otherwise (if DET is disabled) return E_NOT_OK. ()"
- "In case default errors are enabled,..."
- "module raises the Default error XXX_E_TRANSITION"
- "The DET provides services to store default errors"
...

The correct text would be:
- sub chapter is called "7.x Development errors"
- "[SWS_xxx_yyyyy]
In case development error detection is enabled for the xxxx module: The xxxx module shall check API parameters for validity and report detected development errors to the DET. ()"
- "[SWS_xxx_yyyyy]
If development error detection is enabled: the function shall check that the service xxx_Init was previously called. If the check fails, the function shall raise the development error XXX_E_NOT_INITIALIZED otherwise (if DET is disabled) return E_NOT_OK. ()"
- "In case development errors are enabled,..."
- "module raises the development error XXX_E_TRANSITION"
- "The DET provides services to store development errors"

Solution for SWS_RTE:
– SWS_RTE —
- Change 4.8 Default errors to 4.8 Development errors
- Change "Errors which can occur at runtime in the RTE are classified as default errors" to "Errors which can occur at runtime in the RTE are classified as development errors"
- Remove [SWS_Rte_07676]
- Change [SWS_RTE_06611]"If a violation is detected the RTE shall report a default error to the DET." to "If a violation is detected the RTE shall report a development error to the DET."
- Change [SWS_Rte_06631]
[SWS_Rte_06631] d The RTE shall use the OS Application Identifier as the Instance Id to enable the developer to identify in which runtime section of the RTE the error occurs. This Instance ID is even unique across multi cores and so implicitly allows the development error to be traced to a specific core. c(SRS_BSW_00337)

SRS_Libraries:
- In chapter "3 Acronyms and abbreviations": Rename "Development Error Tracer" to "Default Error Tracer"

SRS_SPALGeneral:
- In chapter "6.1.1.3.1 [SRS_SPAL_00157] ...": Rename "Development Error Tracer" to "Default Error Tracer"
- In chapter "6.1.1.4.2 [SRS_SPAL_12448] ...": Rename "Development Error Tracer" to "Default Error Tracer"

SRS_FlashTest:
- In chapter "6.1 Functional Requirements": Rename "Development Error Tracer" to "Default Error Tracer"
- In chapter "7 References":
Rename "Development Error Tracer" to "Default Error Tracer"
Rename "AUTOSAR_SWS_DevelopmentErrorTracer" to "AUTOSAR_SWS_DefaultErrorTracer"

SWS_MFXLibrary:
- In chapter "2 Acronyms and abbreviations": Rename "Development Error Tracer" to "Default Error Tracer"

SWS_MemoryAbstractionInterface:
- In chapter "3.1 Input documents":
Rename "Development Error Tracer" to "Default Error Tracer"
Rename "AUTOSAR_SWS_DevelopmentErrorTracer" to "AUTOSAR_SWS_DefaultErrorTracer"

SWS_FlexRayNetworkManagement:
- In chapter "3.3 Related AUTOSAR documents":
Rename "Development Error Tracer" to "Default Error Tracer"
Rename "AUTOSAR_SWS_DevelopmentErrorTracer" to "AUTOSAR_SWS_DefaultErrorTracer"

SWS_CANStateManager:
- In chapter "3.1 Input documents": Rename "AUTOSAR_SWS_DevelopmentErrorTracer" to "AUTOSAR_SWS_DefaultErrorTracer"

SWS_PDURouter:
- In chapter "3.1 Input documents": Rename "AUTOSAR_SWS_DevelopmentErrorTracer" to "AUTOSAR_SWS_DefaultErrorTracer"

SWS_EEPROMDriver:
- In chapter "3.1 Input documents": Rename "AUTOSAR_SWS_DevelopmentErrorTracer" to "AUTOSAR_SWS_DefaultErrorTracer"
–Last change on issue 73570 comment 47–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.70 Specification Item SWS_Rte_07822

Trace References:

SRS_Rte_00094

Content:

|RTE_E_COM_STOPPED| – the RTE could not perform the operation because the COM communication service is currently not available (inter ECU communication only). RTE shall return |RTE_E_COM_STOPPED| whenthe corresponding COM :

- in case of COM the corresponding service returns |COM_SERVICE_NOT_AVAILABLE| .
- in case of LdCom the corresponding LdCom_Transmit returns |E_NOT_OK|

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76094: [Rte][LdCom] Reaction on E_NOT_OK when calling LdCom_Transmit

  Problem description:

  It is specified, that Rte_Write and Rte_Send shall return RTE_E_OK when
  "data passed to communication service _successfully_"
  But what shall happen if LdCom_Transmit returns E_NOT_OK ?

  Agreed solution:

  On behalf of WP-A1:

  Rephrase
  "[SWS_Rte_07822] RTE_E_COM_STOPPED the RTE could
  not perform the operation because the communication service is currently
  not available (inter ECU communication only). RTE shall return
  RTE_E_COM_STOPPED when:
  -in case of COM the corresponding COM service returns COM_SERVICE_NOT_AVAILABLE
  -in case of LdCom the corresponding LdCom_Transmit returns E_NOT_OK
  ."

  Do the same for SWS_Rte_07823, SWS_Rte_01106, SWS_Rte_06828 and SWS_Rte_01339

  SWS_Rte_01379: replace RTE_E_OK with E_OK.
  –Last change on issue 76094 comment 4–

  BW-C-Level:

Application	Specification	Bus
1	1	1

1.71 Specification Item SWS_Rte_07823

Trace References:

SRS_Rte_00094

Content:

|RTE_E_COM_STOPPED| – the RTE could not perform the operation because the COM communication service is currently not available (inter ECU communication only). RTE shall return |RTE_E_COM_STOPPED| whenthe corresponding COM :

- in case of COM the corresponding service returns |COM_SERVICE_NOT_AVAILABLE| .
- in case of LdCom the corresponding LdCom_Transmit returns |E_NOT_OK|

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76094: [Rte][LdCom] Reaction on E_NOT_OK when calling LdCom_Transmit

  Problem description:

  It is specified, that Rte_Write and Rte_Send shall return RTE_E_OK when
  "data passed to communication service _successfully_"
  But what shall happen if LdCom_Transmit returns E_NOT_OK ?

  Agreed solution:

  On behalf of WP-A1:

  Rephrase
  "[SWS_Rte_07822] RTE_E_COM_STOPPED the RTE could
  not perform the operation because the communication service is currently
  not available (inter ECU communication only). RTE shall return
  RTE_E_COM_STOPPED when:
  -in case of COM the corresponding COM service returns COM_SERVICE_NOT_AVAILABLE
  -in case of LdCom the corresponding LdCom_Transmit returns E_NOT_OK
  ."

  Do the same for SWS_Rte_07823, SWS_Rte_01106, SWS_Rte_06828 and SWS_Rte_01339

  SWS_Rte_01379: replace RTE_E_OK with E_OK.
  –Last change on issue 76094 comment 4–

  BW-C-Level:

Application	Specification	Bus
1	1	1

1.72 Specification Item SWS_Rte_08001

Trace References:

SRS_Rte_00019, SRS_Rte_00033

Content:

If two multiple operations are mapped to the same RunnableEntity, and SWS_Rte_04522 requires a call serialization, then the operation invoked events shall be mapped to same task and they shall have the same position in task. Otherwise the RTE Generator shall reject configuration.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77673: Handling of multiple ServerComSpec queueLength if SWS_Rte_08002 applies

  Problem description:

  Please clarify how to handle multiple ServerComSpec.queueLength in case [SWS_Rte_08002] applies, i.e. several operations are mapped to the same server runnable resulting in a single server invocation queue.

  What is the resulting queue size?

  In my opinion it would make sense to take the sum of all ServerComSpec.queueLengths, but SWCT [constr_1128] could be interpreted as to take only one of the queueLengths (which all have the same value anyway). Otherwise, I don't get the intention behind [constr_1128].

  Btw: Should [SWS_Rte_08001] and [SWS_Rte_08002] better say "If two or more operations..." instead of "If two operations..."?

  Agreed solution:

  – TPS SWC-T –

  Rephrase [TPS_SWCT_01225]:

  [TPS_SWCT_01225] RunnableEntity implements the functionality of more than one ClientServerOperations

A single RunnableEntity can implement the functionality of more than one ClientServerOperations. For this purpose, one OperationInvokedEvent for each affected ClientServerOperation shall reference the respective RunnableEntity. The attribute ServerComSpec.queueLength shall be taken for the determination of the resulting queue length, constr_1128 applies.

– SWS RTE –

[SWS_Rte_08001] d If multiple operations are mapped to the same RunnableEntity, and [SWS_Rte_04522] requires a call serialization, then the operation invoked events
shall be mapped to same task and they shall have the same position in task. Otherwise
the RTE Generator shall reject configuration. c(SRS_Rte_00019, SRS_Rte_00033)

[SWS_Rte_08002] d If multiple operations are mapped to the same RunnableEntity, and [SWS_Rte_04522] requires a call serialization, then a single queue is implemented
for invocations coming from any of the operations. c(SRS_Rte_00019, SRS_Rte_00033)
–Last change on issue 77673 comment 6–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.73 Specification Item SWS_Rte_08002

Trace References:

SRS_Rte_00019, SRS_Rte_00033

Content:

If two multiple operations are mapped to the same RunnableEntity, and SWS_Rte_04522 requires a call serialization, then a single queue is implemented for invocations coming from any of the operations.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77673: Handling of multiple ServerComSpec queueLength if SWS_Rte_08002 applies

Problem description:

Please clarify how to handle multiple ServerComSpec.queueLength in case [SWS_Rte_08002] applies, i.e. several operations are mapped to the same server runnable resulting in a single server invocation queue.

What is the resulting queue size?

In my opinion it would make sense to take the sum of all ServerComSpec.queueLengths, but SWCT [constr_1128] could be interpreted as to take only one of the queueLengths (which all have the same value anyway). Otherwise, I don't get the intention behind [constr_1128].

Btw: Should [SWS_Rte_08001] and [SWS_Rte_08002] better say "If two or more operations..." instead of "If two operations..."?

Agreed solution:

– TPS SWC-T –

Rephrase [TPS_SWCT_01225]:

[TPS_SWCT_01225] RunnableEntity implements the functionality of more than one ClientServerOperations
A single RunnableEntity can implement the functionality of more than one ClientServerOperations. For this purpose, one OperationInvokedEvent for each affected ClientServerOperation shall reference the respective RunnableEntity. The attribute ServerComSpec.queueLength shall be taken for the determination of the resulting queue length, constr_1128 applies.

– SWS RTE –

[SWS_Rte_08001] d If multiple operations are mapped to the same RunnableEntity, and [SWS_Rte_04522] requires a call serialization, then the operation invoked events
shall be mapped to same task and they shall have the same position in task. Otherwise
the RTE Generator shall reject configuration. c(SRS_Rte_00019, SRS_Rte_00033)

[SWS_Rte_08002] d If multiple operations are mapped to the same RunnableEntity, and [SWS_Rte_04522] requires a call serialization, then a single queue is implemented
for invocations coming from any of the operations. c(SRS_Rte_00019,

SRS_Rte_00033)
–Last change on issue 77673 comment 6–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.74 Specification Item SWS_Rte_08065

Trace References:

SRS_BSW_00327, SRS_Rte_00180

Content:

Symbolic name: |RTE_E_OUT_OF_RANGE|
Value: 137
Comments: The received data is out of range.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77161: Tables in specification documents contain trace items

  Problem description:

  ———————————————
  Name: Wolf-Hendrik Kaps
  Phone: 0711-811-23288
  Role: Jg-Tooling member
  ———————————————
  Description/Motivation:

  Some specification documents contain tables which include trace items. E.g. SWS_Com, table in 7.12.1 Development Errors, SWS_Rte, Table 5.4: RTE Error and Status values.

  As discussed in tooling session we should extend ValidateARXML routine to elicit trace items inside tables.

  Further on we shall ensure that tracebles do not contain figures and tables (77206, 74860)
  –Last change on issue 77161 comment 3–

  Agreed solution:

1. Affected Documents
==========

1. Move the Traceable out of the f**ing tables (see attachment)
2. move Tables and figures out of the Traceable

2. Word2arxml and latex2arxml which is used by checkDocumentSource
==========

let these scripts complain also requested by 77206, 74860 but summarized here

* Tables in Traceable
* Traceable in Tables
* Figures in Traceable

2.1 GST: add these contraints
==========

3. CP_Tool_Scripts
==========

* let latexinstatiaator complain about Figures in Traceable
* note: tha table in traceable violates the schema and is flagged already
* let latexinstantiator complain about traceable in tables
–Last change on issue 77161 comment 7–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.75 Specification Item SWS_Rte_08082

Trace References:

SRS_Rte_00177, SRS_Rte_00245

Content:

If an AtomicSwComponentType using a PortPrototype with an NvDataInterface invokes the explicit API Rte_Write and the attributes NvBlockDescriptor.NvBlockDescriptor.dirty

FlagSupport and NvBlockNeeds.NvBlockNeeds.storeCyclic are set to true, the RTE shall write the associated RAM Block(s) to NV memory by calling the NvM_WriteBlock PRAM-Block function of the NvM module in the next cycle of a periodic activity after the data accessed by the Rte_Write function is written back to the RAM Block(s). The periodic activity shall be implemented in the context of an NvBlockDescriptor's RunnableEntity (see requirements SWS_Rte_08086, SWS_Rte_08087, SWS_Rte_08088, SWS_Rte_08089, SWS_Rte_08090) according to the cycle period defined in the attribute NvBlockDescriptor.NvBlockDescriptor.timingEvent.TimingEvent.period.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #74354: In case of NvBlockSwComponentType, instead of NvM_WriteBlock, use NvM_WritePRAMBlock

  Problem description:

  In case of NvBlockSwComponentType, while calling NvM module function for initiating write operation, NvM_WriteBlock is used.
  Here as explicit synchronization shall be used, input argument pointer to Ram data block shall be passed as NULL_PTR while calling NvM_WriteBlock. And actual data shall be copied in the mirror callback.

  Instead, NvM_WritePRAMBlock shall be used with only BlockId as input argument. In this case as well, actual data shall be copied in the mirror callback.

  Following requirements shall be impacted - SWS_Rte_08082, SWS_Rte_08083, SWS_Rte_08084, SWS_Rte_08085.
  Change NvM_WriteBlock with NvM_WritePRAMBlock in these requirements.

  Agreed solution:

  [SWS_Rte_08082] d If an AtomicSwComponentType using a PortPrototype with an NvDataInterface invokes the explicit API Rte_Write and the attributes NvBlockDescriptor.dirtyFlagSupport and NvBlockNeeds.storeCyclic are set to true, the RTE shall write the associated RAM Block(s) to NV memory by calling the NvM_WritePRAMBlock function of the NvM module in the next cycle of a periodic activity after the data accessed by the Rte_Write function is written back to the RAM Block(s). The periodic activity shall be implemented in the context of an NvBlockDescriptors RunnableEntity (see requirements[SWS_Rte_08086], [SWS_Rte_08087], [SWS_Rte_08088], [SWS_Rte_08089],
  [SWS_Rte_08090]) according to the cycle period defined in the attribute NvBlockDescriptor.
  timingEvent.period. c(SRS_Rte_00177, SRS_Rte_00245)

  [SWS_Rte_08083] d If an AtomicSwComponentType using a PortPrototype

with an NvDataInterface invokes the implicit APIs Rte_IWrite / Rte_IWriteRef and the attributes NvBlockDescriptor.dirtyFlagSupport and NvBlockNeeds.storeCyclic are set to true, the RTE shall write the associated RAM Block(s) to NV memory by calling the NvM_WritePRAMBlock function of the NvM module in the cycle of a periodic activity after the data accessed by the Rte_IWrite / Rte_IWriteRef functions is written back from the preemption area buffer to the RAM Block(s) (for further details see chapter 4.3.1.5.1). The periodic activity shall be implemented in the context of an NvBlockDescriptors RunnableEntity (see requirements [SWS_Rte_08086], [SWS_Rte_08087], [SWS_Rte_08088], [SWS_Rte_08089], [SWS_Rte_08090]) according to the cycle period defined in the attribute NvBlockDescriptor. timingEvent.period. c(SRS_Rte_00177, SRS_Rte_00245)

[SWS_Rte_08084] d If an AtomicSwComponentType using a PortPrototype with an NvDataInterface invokes the explicit API Rte_Write and the attributes NvBlockDescriptor.dirtyFlagSupport and NvBlockNeeds.storeImmediate are set to true, the RTE shall write the associated RAM Block(s) to NV memory by calling the NvM_WritePRAMBlock function of the NvM module. The NvM_WritePRAMBlock function shall be called in the context of an NvBlockDescriptors RunnableEntity (see requirements [SWS_Rte_08086], [SWS_Rte_08087], [SWS_Rte_08088], [SWS_Rte_08089], [SWS_Rte_08090]) after the data accessed by the Rte_Write function is written back to the RAM Block(s). c(SRS_Rte_00177,SRS_Rte_00245)

[SWS_Rte_08085] d If an AtomicSwComponentType using a PortPrototype with an NvDataInterface invokes the implicit APIs Rte_IWrite / Rte_IWriteRef and the attributes NvBlockDescriptor.dirtyFlagSupport and NvBlockNeeds.storeImmediate are set to true, the RTE shall write the associated RAM Block(s) to NV memory by calling the NvM_WritePRAMBlock function of the NvM module. The function NvM_WritePRAMBlock shall be called in the context of an NvBlockDescriptors RunnableEntity (see requirements [SWS_Rte_08086], [SWS_Rte_08087], [SWS_Rte_08088], [SWS_Rte_08089], [SWS_Rte_08090]) after the data accessed by the Rte_IWrite / Rte_IWriteRef functions is written back from the preemption area buffer to the RAM Block(s) (for further details see chapter 4.3.1.5.1). c(SRS_Rte_00177, SRS_Rte_00245)

Add NvM_WritePRAMBlock to figure 4.34.
–Last change on issue 74354 comment 6–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.76 Specification Item SWS_Rte_08083

Trace References:

SRS_Rte_00177, SRS_Rte_00245

Content:

If an AtomicSwComponentType using a PortPrototype with an NvDataInterface invokes the implicit APIs Rte_IWrite / Rte_IWriteRef and the attributes NvBlockDescriptor.NvBlockDescriptor.dirtyFlagSupport and NvBlockNeeds.NvBlockNeeds.storeCyclic are set to true, the RTE shall write the associated RAM Block(s) to NV memory by calling the NvM_WriteBlock PRAMBlock function of the NvM module in the cycle of a periodic activity after the data accessed by the Rte_IWrite / Rte_IWriteRef functions is written back from the preemption area buffer to the RAM Block(s) (for further details see chapter [REF sec_3a_TheConceptOfImplicitDataAccess_2f_transmission]). The periodic activity shall be implemented in the context of an NvBlockDescriptor's RunnableEntity (see requirements SWS_Rte_08086, SWS_Rte_08087, SWS_Rte_08088, SWS_Rte_08089, SWS_Rte_08090) according to the cycle period defined in the attribute NvBlockDescriptor.NvBlockDescriptor.timingEvent.TimingEvent.period.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #74354: In case of NvBlockSwComponentType, instead of NvM_WriteBlock, use NvM_WritePRAMBlock

  Problem description:

  In case of NvBlockSwComponentType, while calling NvM module function for initiating write operation, NvM_WriteBlock is used.
  Here as explicit synchronization shall be used, input argument pointer to Ram data block shall be passed as NULL_PTR while calling NvM_WriteBlock. And actual data shall be copied in the mirror callback.

  Instead, NvM_WritePRAMBlock shall be used with only BlockId as input argument. In this case as well, actual data shall be copied in the mirror callback.

  Following requirements shall be impacted - SWS_Rte_08082, SWS_Rte_08083, SWS_Rte_08084, SWS_Rte_08085.
  Change NvM_WriteBlock with NvM_WritePRAMBlock in these requirements.

Agreed solution:

[SWS_Rte_08082] d If an AtomicSwComponentType using a PortPrototype with an NvDataInterface invokes the explicit API Rte_Write and the attributes NvBlockDescriptor.dirtyFlagSupport and NvBlockNeeds.storeCyclic are set to true, the RTE shall write the associated RAM Block(s) to NV memory by calling the NvM_WritePRAMBlock function of the NvM module in the next cycle of a periodic activity after the data accessed by the Rte_Write function is written back to the RAM Block(s). The periodic activity shall be implemented in the context of an NvBlockDescriptors RunnableEntity (see requirements[SWS_Rte_08086], [SWS_Rte_08087], [SWS_Rte_08088], [SWS_Rte_08089],
[SWS_Rte_08090]) according to the cycle period defined in the attribute NvBlockDescriptor.
timingEvent.period. c(SRS_Rte_00177, SRS_Rte_00245)

[SWS_Rte_08083] d If an AtomicSwComponentType using a PortPrototype with an NvDataInterface invokes the implicit APIs Rte_IWrite / Rte_IWriteRef
and the attributes NvBlockDescriptor.dirtyFlagSupport and NvBlockNeeds.storeCyclic are set to true, the RTE shall write the associated RAM Block(s) to NV memory by calling the NvM_WritePRAMBlock function of the NvM module in the cycle of a periodic activity after the data accessed by the Rte_IWrite / Rte_IWriteRef functions is written back from the preemption area buffer to the RAM Block(s) (for further details see chapter 4.3.1.5.1). The periodic activity shall be implemented in the context of an NvBlockDescriptors RunnableEntity (see requirements [SWS_Rte_08086], [SWS_Rte_08087], [SWS_Rte_08088], [SWS_Rte_08089], [SWS_Rte_08090]) according to the cycle period defined in the attribute NvBlockDescriptor. timingEvent.period. c(SRS_Rte_00177, SRS_Rte_00245)

[SWS_Rte_08084] d If an AtomicSwComponentType using a PortPrototype with an NvDataInterface invokes the explicit API Rte_Write and the attributes NvBlockDescriptor.dirtyFlagSupport and NvBlockNeeds.storeImmediate are set to true, the RTE shall write the associated RAM Block(s) to
NV memory by calling the NvM_WritePRAMBlock function of the NvM module. The NvM_WritePRAMBlock function shall be called in the context of an NvBlockDescriptors
RunnableEntity (see requirements [SWS_Rte_08086], [SWS_Rte_08087], [SWS_Rte_08088], [SWS_Rte_08089], [SWS_Rte_08090]) after the data accessed
by the Rte_Write function is written back to the RAM Block(s). c(SRS_Rte_00177,SRS_Rte_00245)

[SWS_Rte_08085] d If an AtomicSwComponentType using a PortPrototype with an NvDataInterface invokes the implicit APIs Rte_IWrite / Rte_IWriteRef and the attributes NvBlockDescriptor.dirtyFlagSupport and NvBlockNeeds.storeImmediate are set to true, the RTE shall write the associated RAM Block(s) to NV memory by calling the NvM_WritePRAMBlock function of the NvM module. The function NvM_WritePRAMBlock shall be called in the context of an NvBlockDescriptors RunnableEntity (see requirements [SWS_Rte_08086], [SWS_Rte_08087], [SWS_Rte_08088], [SWS_Rte_08089], [SWS_Rte_08090]) after the data accessed by the Rte_IWrite / Rte_IWriteRef functions is written back from the preemption area buffer to the RAM Block(s) (for further details see chapter 4.3.1.5.1). c(SRS_Rte_00177, SRS_Rte_00245)

Add NvM_WritePRAMBlock to figure 4.34.
–Last change on issue 74354 comment 6–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.77 Specification Item SWS_Rte_08084

Trace References:

SRS_Rte_00177, SRS_Rte_00245

Content:

If an AtomicSwComponentType using a PortPrototype with an NvDataInterface invokes the explicit API Rte_Write and the attributes NvBlockDescriptor.NvBlockDescriptor.dirty FlagSupport and NvBlockNeeds.NvBlockNeeds.storeImmediate are set to true, the RTE shall write the associated RAM Block(s) to NV memory by calling the NvM_Write Block PRAMBlock function of the NvM module. The NvM_WriteBlock PRAMBlock function shall be called in the context of an NvBlockDescriptor's RunnableEntity (see requirements SWS_Rte_08086, SWS_Rte_08087, SWS_Rte_08088, SWS_Rte_08089, SWS_Rte_08090) after the data accessed by the Rte_Write function is written back to the RAM Block(s).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #74354: In case of NvBlockSwComponentType, instead of NvM_WriteBlock, use NvM_WritePRAMBlock

  Problem description:

In case of NvBlockSwComponentType, while calling NvM module function for initiating write operation, NvM_WriteBlock is used.
Here as explicit synchronization shall be used, input argument pointer to Ram data block shall be passed as NULL_PTR while calling NvM_WriteBlock. And actual data shall be copied in the mirror callback.

Instead, NvM_WritePRAMBlock shall be used with only BlockId as input argument. In this case as well, actual data shall be copied in the mirror callback.

Following requirements shall be impacted - SWS_Rte_08082, SWS_Rte_08083, SWS_Rte_08084, SWS_Rte_08085.
Change NvM_WriteBlock with NvM_WritePRAMBlock in these requirements.

Agreed solution:

[SWS_Rte_08082] d If an AtomicSwComponentType using a PortPrototype with an NvDataInterface invokes the explicit API Rte_Write and the attributes NvBlockDescriptor.dirtyFlagSupport and NvBlockNeeds.storeCyclic are set to true, the RTE shall write the associated RAM Block(s) to NV memory by calling the NvM_WritePRAMBlock function of the NvM module in the next cycle of a periodic activity after the data accessed by the Rte_Write function is written back to the RAM Block(s). The periodic activity shall be implemented in the context of an NvBlockDescriptors RunnableEntity (see requirements[SWS_Rte_08086], [SWS_Rte_08087], [SWS_Rte_08088], [SWS_Rte_08089],
[SWS_Rte_08090]) according to the cycle period defined in the attribute NvBlockDescriptor.
timingEvent.period. c(SRS_Rte_00177, SRS_Rte_00245)

[SWS_Rte_08083] d If an AtomicSwComponentType using a PortPrototype with an NvDataInterface invokes the implicit APIs Rte_IWrite / Rte_IWriteRef
and the attributes NvBlockDescriptor.dirtyFlagSupport and NvBlockNeeds.storeCyclic are set to true, the RTE shall write the associated RAM Block(s) to NV memory by calling the NvM_WritePRAMBlock function of the NvM module in the cycle of a periodic activity after the data accessed by the Rte_IWrite / Rte_IWriteRef functions is written back from the preemption area buffer to the RAM Block(s) (for further details see chapter 4.3.1.5.1). The periodic activity shall be implemented in the context of an NvBlockDescriptors RunnableEntity (see requirements [SWS_Rte_08086], [SWS_Rte_08087], [SWS_Rte_08088], [SWS_Rte_08089], [SWS_Rte_08090]) according to the cycle period defined in the attribute NvBlockDescriptor. timingEvent.period. c(SRS_Rte_00177, SRS_Rte_00245)

[SWS_Rte_08084] d If an AtomicSwComponentType using a PortPrototype

with an NvDataInterface invokes the explicit API Rte_Write and the attributes NvBlockDescriptor.dirtyFlagSupport and NvBlockNeeds.storeImmediate are set to true, the RTE shall write the associated RAM Block(s) to
NV memory by calling the NvM_WritePRAMBlock function of the NvM module. The NvM_WritePRAMBlock function shall be called in the context of an NvBlockDescriptors
RunnableEntity (see requirements [SWS_Rte_08086], [SWS_Rte_08087], [SWS_Rte_08088], [SWS_Rte_08089], [SWS_Rte_08090]) after the data accessed
by the Rte_Write function is written back to the RAM Block(s). c(SRS_Rte_00177,SRS_Rte_00245)

[SWS_Rte_08085] d If an AtomicSwComponentType using a PortPrototype with an NvDataInterface invokes the implicit APIs Rte_IWrite / Rte_IWriteRef
and the attributes NvBlockDescriptor.dirtyFlagSupport and NvBlockNeeds.storeImmediate are set to true, the RTE shall write the associated RAM Block(s) to NV memory by calling the NvM_WritePRAMBlock function of the NvM module. The function NvM_WritePRAMBlock shall be called in the context of an NvBlockDescriptors RunnableEntity (see requirements [SWS_Rte_08086], [SWS_Rte_08087], [SWS_Rte_08088], [SWS_Rte_08089], [SWS_Rte_08090]) after the data accessed by the Rte_IWrite / Rte_IWriteRef functions is written back from the preemption area buffer to the RAM Block(s) (for further details see chapter 4.3.1.5.1). c(SRS_Rte_00177, SRS_Rte_00245)

Add NvM_WritePRAMBlock to figure 4.34.
–Last change on issue 74354 comment 6–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.78 Specification Item SWS_Rte_08085

Trace References:

SRS_Rte_00177, SRS_Rte_00245

Content:

If an AtomicSwComponentType using a PortPrototype with an NvDataInterface invokes the implicit APIs Rte_IWrite / Rte_IWriteRef and the attributes NvBlockDescriptor.NvBlock Descriptor.dirtyFlagSupport and NvBlockNeeds.NvBlockNeeds.storeImmediate are set to true, the RTE shall write the associated RAM Block(s) to NV memory by calling the Nv

M_WriteBlock PRAMBlock function of the NvM module. The function NvM_WriteBlock PRAMBlock shall be called in the context of an NvBlockDescriptor's RunnableEntity (see requirements SWS_Rte_08086, SWS_Rte_08087, SWS_Rte_08088, SWS_Rte_08089, SWS_Rte_08090) after the data accessed by the Rte_IWrite / Rte_IWriteRef functions is written back from the preemption area buffer to the RAM Block(s) (for further details see chapter [REF sec_3a_TheConceptOfImplicitDataAccess_2f_transmission]).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #74354: In case of NvBlockSwComponentType, instead of NvM_WriteBlock, use NvM_WritePRAMBlock

  Problem description:

  In case of NvBlockSwComponentType, while calling NvM module function for initiating write operation, NvM_WriteBlock is used.
  Here as explicit synchronization shall be used, input argument pointer to Ram data block shall be passed as NULL_PTR while calling NvM_WriteBlock. And actual data shall be copied in the mirror callback.

  Instead, NvM_WritePRAMBlock shall be used with only BlockId as input argument. In this case as well, actual data shall be copied in the mirror callback.

  Following requirements shall be impacted - SWS_Rte_08082, SWS_Rte_08083, SWS_Rte_08084, SWS_Rte_08085.
  Change NvM_WriteBlock with NvM_WritePRAMBlock in these requirements.

  Agreed solution:

  [SWS_Rte_08082] d If an AtomicSwComponentType using a PortPrototype with an NvDataInterface invokes the explicit API Rte_Write and the attributes NvBlockDescriptor.dirtyFlagSupport and NvBlockNeeds.storeCyclic are set to true, the RTE shall write the associated RAM Block(s) to NV memory by calling the NvM_WritePRAMBlock function of the NvM module in the next cycle of a periodic activity after the data accessed by the Rte_Write function is written back to the RAM Block(s). The periodic activity shall be implemented in the context of an NvBlockDescriptors RunnableEntity (see requirements[SWS_Rte_08086], [SWS_Rte_08087], [SWS_Rte_08088], [SWS_Rte_08089],
  [SWS_Rte_08090]) according to the cycle period defined in the attribute NvBlockDescriptor.
  timingEvent.period. c(SRS_Rte_00177, SRS_Rte_00245)

  [SWS_Rte_08083] d If an AtomicSwComponentType using a PortPrototype with an NvDataInterface invokes the implicit APIs Rte_IWrite / Rte_IWriteRef
  and the attributes NvBlockDescriptor.dirtyFlagSupport and NvBlock-

Needs.storeCyclic are set to true, the RTE shall write the associated RAM Block(s) to NV memory by calling the NvM_WritePRAMBlock function of the NvM module in the cycle of a periodic activity after the data accessed by the Rte_IWrite / Rte_IWriteRef functions is written back from the preemption area buffer to the RAM Block(s) (for further details see chapter 4.3.1.5.1). The periodic activity shall be implemented in the context of an NvBlockDescriptors RunnableEntity (see requirements [SWS_Rte_08086], [SWS_Rte_08087], [SWS_Rte_08088], [SWS_Rte_08089], [SWS_Rte_08090]) according to the cycle period defined in the attribute NvBlockDescriptor. timingEvent.period. c(SRS_Rte_00177, SRS_Rte_00245)

[SWS_Rte_08084] d If an AtomicSwComponentType using a PortPrototype with an NvDataInterface invokes the explicit API Rte_Write and the attributes NvBlockDescriptor.dirtyFlagSupport and NvBlockNeeds.storeImmediate are set to true, the RTE shall write the associated RAM Block(s) to NV memory by calling the NvM_WritePRAMBlock function of the NvM module. The NvM_WritePRAMBlock function shall be called in the context of an NvBlockDescriptors RunnableEntity (see requirements [SWS_Rte_08086], [SWS_Rte_08087], [SWS_Rte_08088], [SWS_Rte_08089], [SWS_Rte_08090]) after the data accessed by the Rte_Write function is written back to the RAM Block(s). c(SRS_Rte_00177,SRS_Rte_00245)

[SWS_Rte_08085] d If an AtomicSwComponentType using a PortPrototype with an NvDataInterface invokes the implicit APIs Rte_IWrite / Rte_IWriteRef and the attributes NvBlockDescriptor.dirtyFlagSupport and NvBlockNeeds.storeImmediate are set to true, the RTE shall write the associated RAM Block(s) to NV memory by calling the NvM_WritePRAMBlock function of the NvM module. The function NvM_WritePRAMBlock shall be called in the context of an NvBlockDescriptors RunnableEntity (see requirements [SWS_Rte_08086], [SWS_Rte_08087], [SWS_Rte_08088], [SWS_Rte_08089], [SWS_Rte_08090]) after the data accessed by the Rte_IWrite / Rte_IWriteRef functions is written back from the preemption area buffer to the RAM Block(s) (for further details see chapter 4.3.1.5.1). c(SRS_Rte_00177, SRS_Rte_00245)

Add NvM_WritePRAMBlock to figure 4.34.
–Last change on issue 74354 comment 6–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.79 Specification Item SWS_Rte_08400

Trace References:

SRS_Rte_00210

Content:

The RTE Generator in Generation Phase shall generate internal ImplementationData Types types used for IOC configuration, if the IOC module is used.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #68369: Allow optimized inter-core communication mechanism

  Problem description:

  An internal evaluation of the IOC within Continental shows following drawbacks, among others:
  - IOC does not support multiple receivers (The 1:N communication is done with the help of the RTE by subsequently calling IOC)
  - No buffer reuse in IOC (i.e. increased resource overhead)
  - No shared remote and local usage of the IOC buffer (i.e. resource overhead with risk of coherency glitches)
  - No variable grouping of IOC read/write operations to build blocks (i.e. increased runtime overhead and risk of coherency glitches)
  - IOC uses copy operations even for read operations of primitive data (i.e. resource overhead)
  - For memory partitioning the IOC is usually not needed. Instead the MPU is configured with read all policy.

  The IOC seems from specification point not very well optimized with respect to resources and flexibility. It turns out the IOC is less efficient than individual vendor specific solutions.

  Therefore we propose to allow the implementation of an optimized vendor specific implementation for inter-core communication. A vendor specific solution could support the individual domain specific needs regarding optimization of buffering and data protection. Also the system dynamics can be taken into account. Additionally such an implementation can be customized for a dedicated microcontroller.

  The IOC module shall remain mandatory in the OS but it shall be allowed to use a different mechanism to cross core or memory partition boundaries. In case the IOC is not used, the inter-core communication could be done by a vendor specific RTE extension.

Referring the RTE and OS specification this seems not to be explicitly forbidden so far (to be proven by document owner). But both the specifications give the impression that the usage of the IOC is mandatory for inter-partition communication.

E.g.:
- AUTOSAR_SWS_OS (Rev. 4.2.1), chapter 7.10.2 IOC General purpose: To keep the RTE as hardware independent as possible, all inter OS-Application and inter core communication mechanisms and implementation variants are encapsulated in the IOC.
- AUTOSAR_SWS_RTE (Rev. 4.2.1), chapter 4.3 Communication Paradigms: For inter-Partition communication (within the same ECU) the RTE uses functionalities provided by the IOC module.

Therefore it should be explicitly mentioned that the inter-partition communication could be also realized by some optimized alternative mechanism (not using the IOC). Either directly realized by the RTE or by invoking some vendor specific module.

Agreed solution:

See attachment# 3657
–Last change on issue 68369 comment 13–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.80 Specification Item SWS_Rte_08531

Trace References:

SRS_Rte_00248

Content:

The If the attribute BufferProperties.inPlace in the BufferProperties of a Transformation Technology is set to FALSE, the RTE shall provide the a separate buffer to the transformers which contains their input datain which they can write their output.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #74076: Redundant requirement w.r.t to buffer handling

Problem description:

Below requirements exist in SWS_RTE.

[SWS_Rte_08531] d The RTE shall provide the buffer to the transformers which contains their input data. c(SRS_Rte_00248)

[SWS_Rte_08532] d If the attribute inPlace in the BufferProperties of a TransformationTechnology is not set, the RTE shall provide a separate buffer to the transformers in which they can write their output. c(SRS_Rte_00248)
If the attribute inPlace in the BufferProperties of a TransformationTechnology is set, the RTE doesnt need to provide a separate output buffer to the transformer because with inplace buffer handling the transformer will read the input data
from a buffer and writes its output into the same buffer. For this, the RTE hands over to the transformer a pointer and a length which represents the buffer both for input an output.

[SWS_Rte_08533] d The RTE shall provide a buffer to the transformer for the transformers
output. c(SRS_Rte_00248)

According to my understanding [SWS_Rte_08531] and [SWS_Rte_08533] are redundant as there are already covered by [SWS_Rte_08532]

Agreed solution:

Remove [SWS_Rte_08533] and update [SWS_Rte_08531] resp. [SWS_Rte_08532] accordingly

[SWS_Rte_08531] If the attribute inPlace in the BufferProperties of a TransformationTechnology is set to FALSE, the RTE shall provide a separate buffer to the transformers in which they can write their output. c(SRS_Rte_00248)

[SWS_Rte_08532]If the attribute inPlace in the BufferProperties of a TransformationTechnology is set to TRUE, the RTE shall provide one buffer to the transformer.
Rationale: With inplace buffer handling the transformer will read the input data from a buffer and writes its output into the same buffer. For this, the RTE hands over to the transformer a pointer and a length which represents the buffer both for input an output.
–Last change on issue 74076 comment 7–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.81 Specification Item SWS_Rte_08532

Trace References:

SRS_Rte_00248

Content:

If the attribute BufferProperties.inPlace in the BufferProperties of a TransformationTechnology is not set set to TRUE, the RTE shall provide a separate one buffer to the transformers in which they can write their outputtransformer.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #74076: Redundant requirement w.r.t to buffer handling

  Problem description:

  Below requirements exist in SWS_RTE.

  [SWS_Rte_08531] d The RTE shall provide the buffer to the transformers which contains their input data. c(SRS_Rte_00248)

  [SWS_Rte_08532] d If the attribute inPlace in the BufferProperties of a TransformationTechnology is not set, the RTE shall provide a separate buffer to the transformers in which they can write their output. c(SRS_Rte_00248)
  If the attribute inPlace in the BufferProperties of a TransformationTechnology is set, the RTE doesnt need to provide a separate output buffer to the transformer because with inplace buffer handling the transformer will read the input data
  from a buffer and writes its output into the same buffer. For this, the RTE hands over to the transformer a pointer and a length which represents the buffer both for input an output.

  [SWS_Rte_08533] d The RTE shall provide a buffer to the transformer for the transformers
  output. c(SRS_Rte_00248)

  According to my understanding [SWS_Rte_08531] and [SWS_Rte_08533] are redundant as there are already covered by [SWS_Rte_08532]

  Agreed solution:

Remove [SWS_Rte_08533] and update [SWS_Rte_08531] resp. [SWS_Rte_08532] accordingly

[SWS_Rte_08531] If the attribute inPlace in the BufferProperties of a TransformationTechnology is set to FALSE, the RTE shall provide a separate buffer to the transformers in which they can write their output. c(SRS_Rte_00248)

[SWS_Rte_08532]If the attribute inPlace in the BufferProperties of a TransformationTechnology is set to TRUE, the RTE shall provide one buffer to the transformer.
Rationale: With inplace buffer handling the transformer will read the input data from a buffer and writes its output into the same buffer. For this, the RTE hands over to the transformer a pointer and a length which represents the buffer both for input an output.
–Last change on issue 74076 comment 7–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.82 Specification Item SWS_Rte_08533

Trace References:

SRS_Rte_00248

Content:

The RTE shall provide a buffer to the transformer for the transformer's output.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #74076: Redundant requirement w.r.t to buffer handling

  Problem description:

  Below requirements exist in SWS_RTE.

  [SWS_Rte_08531] d The RTE shall provide the buffer to the transformers which contains their input data. c(SRS_Rte_00248)

  [SWS_Rte_08532] d If the attribute inPlace in the BufferProperties of a TransformationTechnology is not set, the RTE shall provide a separate buffer to the transformers in which they can write their output. c(SRS_Rte_00248)

If the attribute inPlace in the BufferProperties of a TransformationTechnology is set, the RTE doesnt need to provide a separate output buffer to the transformer because with inplace buffer handling the transformer will read the input data
from a buffer and writes its output into the same buffer. For this, the RTE hands over to the transformer a pointer and a length which represents the buffer both for input an output.

[SWS_Rte_08533] d The RTE shall provide a buffer to the transformer for the transformers
output. c(SRS_Rte_00248)

According to my understanding [SWS_Rte_08531] and [SWS_Rte_08533] are redundant as there are already covered by [SWS_Rte_08532]

Agreed solution:

Remove [SWS_Rte_08533] and update [SWS_Rte_08531] resp. [SWS_Rte_08532] accordingly

[SWS_Rte_08531] If the attribute inPlace in the BufferProperties of a TransformationTechnology is set to FALSE, the RTE shall provide a separate buffer to the transformers in which they can write their output. c(SRS_Rte_00248)

[SWS_Rte_08532]If the attribute inPlace in the BufferProperties of a TransformationTechnology is set to TRUE, the RTE shall provide one buffer to the transformer.
Rationale: With inplace buffer handling the transformer will read the input data from a buffer and writes its output into the same buffer. For this, the RTE hands over to the transformer a pointer and a length which represents the buffer both for input an output.
–Last change on issue 74076 comment 7–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.83 Specification Item SWS_Rte_08551

Trace References:

SRS_Rte_00091, SRS_BSW_00327

Content:

Symbolic name: |RTE_E_SOFT_TRANSFORMER_ERROR|
|ERROR| Value: 140
Comments: An error during transformation occured which shall be notified to the SWC but still produces valid data as output (comparable to a warning).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77161: Tables in specification documents contain trace items

  Problem description:

  _____
  Name: Wolf-Hendrik Kaps
  Phone: 0711-811-23288
  Role: Jg-Tooling member
  _____
  Description/Motivation:

  Some specification documents contain tables which include trace items. E.g. SWS_Com, table in 7.12.1 Development Errors, SWS_Rte, Table 5.4: RTE Error and Status values.

  As discussed in tooling session we should extend ValidateARXML routine to elicit trace items inside tables.

  Further on we shall ensure that tracebles do not contain figures and tables (77206, 74860)
  –Last change on issue 77161 comment 3–

  Agreed solution:

  1. Affected Documents
  ==========

  1. Move the Traceable out of the f**ing tables (see attachment)
  2. move Tables and figures out of the Traceable

  2. Word2arxml and latex2arxml which is used by checkDocumentSource
  ==========

  let these scripts complain also requested by 77206, 74860 but summarized here

* Tables in Traceable
* Traceable in Tables
* Figures in Traceable

2.1 GST: add these contraints
==========

3. CP_Tool_Scripts
==========

* let latexinstatiaator complain about Figures in Traceable
* note: tha table in traceable violates the schema and is flagged already
* let latexinstantiator complain about traceable in tables
–Last change on issue 77161 comment 7–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.84 Specification Item SWS_Rte_08725

Trace References:

SRS_Rte_00091, SRS_BSW_00327

Content:

Symbolic name: |RTE_E_SERIALIZATION_| |ERROR|, |RTE_E_HARD_TRANSFORMER_ERROR|
|ERROR| Value: 138
Comments: An error during transformation occured.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77161: Tables in specification documents contain trace items

  Problem description:

  Name: Wolf-Hendrik Kaps
  Phone: 0711-811-23288
  Role: Jg-Tooling member

  Description/Motivation:

Some specification documents contain tables which include trace items. E.g. SWS_Com, table in 7.12.1 Development Errors, SWS_Rte, Table 5.4: RTE Error and Status values.

As discussed in tooling session we should extend ValidateARXML routine to elicit trace items inside tables.

Further on we shall ensure that tracebles do not contain figures and tables (77206, 74860)
–Last change on issue 77161 comment 3–

Agreed solution:

1. Affected Documents
==========

1. Move the Traceable out of the f**ing tables (see attachment)
2. move Tables and figures out of the Traceable

2. Word2arxml and latex2arxml which is used by checkDocumentSource
==========

let these scripts complain also requested by 77206, 74860 but summarized here

* Tables in Traceable
* Traceable in Tables
* Figures in Traceable

2.1 GST: add these contraints
==========

3. CP_Tool_Scripts
==========

* let latexinstatiaator complain about Figures in Traceable
* note: tha table in traceable violates the schema and is flagged already
* let latexinstantiator complain about traceable in tables
–Last change on issue 77161 comment 7–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.85 Specification Item SWS_Rte_08726

Trace References:

SRS_Rte_00091, SRS_BSW_00327

Content:

Symbolic name: |RTE_E_SERIALIZATION_| |LIMIT|, |RTE_E_TRANSFORMER_LIMIT| |LIMIT| Value: 139
Comments: Buffer for transformation operation could not be created.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77161: Tables in specification documents contain trace items

  Problem description:

  ———————————
  Name: Wolf-Hendrik Kaps
  Phone: 0711-811-23288
  Role: Jg-Tooling member
  ———————————
  Description/Motivation:

  Some specification documents contain tables which include trace items. E.g. SWS_Com, table in 7.12.1 Development Errors, SWS_Rte, Table 5.4: RTE Error and Status values.

  As discussed in tooling session we should extend ValidateARXML routine to elicit trace items inside tables.

  Further on we shall ensure that tracebles do not contain figures and tables (77206, 74860)
  –Last change on issue 77161 comment 3–

  Agreed solution:

  1. Affected Documents
  ==========

  1. Move the Traceable out of the f**ing tables (see attachment)

2. move Tables and figures out of the Traceable

2. Word2arxml and latex2arxml which is used by checkDocumentSource
==========

let these scripts complain also requested by 77206, 74860 but summarized here

* Tables in Traceable
* Traceable in Tables
* Figures in Traceable

2.1 GST: add these contraints
==========

3. CP_Tool_Scripts
==========

* let latexinstatiaator complain about Figures in Traceable
* note: tha table in traceable violates the schema and is flagged already
* let latexinstantiator complain about traceable in tables
–Last change on issue 77161 comment 7–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.86 Specification Item SWS_Rte_08802

Trace References:

none

Content:

An AutosarDataType is used if it meets any one of the following conditions:

- it is referenced by a DataPrototype in the InternalBehavior, or
- it is referenced by a VariationPointProxy in the InternalBehavior, or

- it is referenced by a DataPrototype in a PortInterface referenced by a PortPrototype, or
- it is referenced by an IncludedDataTypeSet in the InternalBehavior, or
- it is the ImplementationDataType mapped to an ApplicationDataType (i.e. via the DataTypeMappingSet) that is used in one of the above ways, or
- it is an ImplementationDataTypeElement of a complex ImplementationDataType that is used in one of the above ways, or
- it is referenced as the target type of an ImplementationDataType or ImplementationDataTypeElement of category TYPE_REFERENCE that is used in one of the above ways, or
- it is an ApplicationDataType referenced as the type of a sub-element of a complex ApplicationDataType that is used in one of the above ways.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #73034: Clarification required regarding applicability of SWS_Rte_03809 - should labels be generated for mapped ImplementationDataTypes?

  Problem description:

  SWS_Rte_03809 states that, for a given ApplicationSoftwareComponent, the constants associated with all 'used' AutosarDataTypes must be included in the Application Types Header. It is noted that AutosarDataTypes referenced from relevant DataPrototypes and AutosarDataTypes referenced from IncludedDataTypeSets are all 'used' in this sense and further that sub-element data types of 'used' complex data types are also 'used', as are data types that are aliased by 'used' type reference data types.

  What is not entirely clear is whether an ImplementationDataType that is mapped by the component to a 'used' ApplicationDataType is to be considered 'used' if it is not directly referenced from a DataPrototype or IncludedDataTypeSet itself.

  On the one hand, this case is not explicitly included, although it is not clear whether the list is intended to be exhaustive. Indeed it does not mention element types of 'used' complex ApplicationDataTypes although it would seem strange if they were not to be included analogously to the ImplementationDataType case that is explicitly included.

  On the other hand, the example given in the definition of 'AI' within Table 5.39 in the SoftwareComponentTemplate (AR4.2.2) suggests that the labels from both types may be used. Indeed, when a DataPrototype references an Application-

DataType then the corresponding APIs use the mapped ImplementationDataType so it is reasonable to expect the software component code to use its labels, as well as the aliases from the ApplicationDataType. It could also be argued that the ImplementationDataType /is/ referenced by the component, via the referenced DataTypeMappingSet.

Issue # 72232 has clarified that in the case of labels associated with both an ApplicationDataType and its mapped ImplementationDataType an IncludedDataTypeSet referencing only the ApplicationDataType gives its literalPrefix (by SWS_Rte_03810) to the definitions for the ApplicationDataType only but it is not entirely clear whether the mapped ImplementationDataType should be considered 'used' at all in this case. Since there is no API or other code artefact resulting from an IncludedDataTypeSet it seems reasonable that different behaviour may be required in this respect compared to the case where the ApplicationDataType is referenced by a DataPrototype.

Having the labels defined for both types ensures that whichever labels the component code uses will be available but the downside is that conflicts between labels of different ImplementationDataTypes may occur and require entries in IncludedDataTypeSets to resolve even if the labels are not actually being used in the code. A mismatch of interpretations in either direction can be worked around using IncludedDataTypeSets but a common understanding is required to reduce this workload on integrators.

Agreed solution:

— SWS RTE —
[SWS_Rte_03809] d The Application Types Header File shall include the definitions of all constants of ImplementationDataTypes and ApplicationDataTypes for each ImplementationDataType/ApplicationDataTypes used (See [SWS_Rte_0xxx1] for the meaning of the term "used") by this software component.

[SWS_Rte_03983] d The The Module Interlink Types Header File shall include the definitions of all constants of ImplementationDataTypes and Application-DataTypes for each ImplementationDataType/ApplicationDataTypes used (See [SWS_Rte_0xxx2] for the meaning of the term "used") by this Basic Software module.

Add [SWS_Rte_0xxx1] in chapter 5.5.5

[SWS_Rte_0xxx1] The meaning of the term "used" with respect to Autosar-DataTypes [
An AutosarDataType is used if it meets any one of the following conditions:

- it is referenced by a DataPrototype in the InternalBehavior, or
- it is referenced by a VariationPointProxy in the InternalBehavior, or
- it is referenced by a DataPrototype in a PortInterface referenced by a PortPrototype, or
- it is referenced by an IncludedDataTypeSet in the InternalBehavior, or
- it is the ImplementationDataType mapped to an ApplicationDataType (i.e. via the DataTypeMappingSet) that is used in one of the above ways, or
- it is an ImplementationDataTypeElement of a complex ImplementationDataType that is used in one of the above ways, or
- it is referenced as the target type of an ImplementationDataType or ImplementationDataTypeElement of category TYPE_REFERENCE that is used in one of the above ways, or
- it is an ApplicationDataType referenced as the type of a sub-element of a complex ApplicationDataType that is used in one of the above ways.

]
Please note that in contrast to the TYPE_REFERENCE case, when an ImplementationDataType of category DATA_REFERENCE is "used" the target ImplementationDataType it references is not considered used, unless it is independently used in its own right.
-
Add [SWS_Rte_0xxx2] in chapter 6.4.3
[SWS_Rte_0xxx2] The meaning of the term "used" with respect to AutosarDataTypes [
An AutosarDataType is used if it meets any one of the following conditions:

- it is referenced by a DataPrototype in the InternalBehavior, or
- it is referenced by a VariationPointProxy in the internalBehavior, or
- it is referenced by a DataPrototype referenced by a providedData or requiredData, or
- it is referenced by an IncludedDataTypeSet in the InternalBehavior, or
- it is the ImplementationDataType mapped to an ApplicationDataType (i.e. via the DataTypeMappingSet) that is used in one of the above ways, or
- it is an ImplementationDataTypeElement of a complex ImplementationDataType that is used in one of the above ways, or
- it is referenced as the target type of an ImplementationDataType or ImplementationDataTypeElement of category TYPE_REFERENCE that is used in one of the above ways, or
- it is an ApplicationDataType referenced as the type of a sub-element of a complex ApplicationDataType that is used in one of the above ways.

]
Please note that in contrast to the TYPE_REFERENCE case, when an ImplementationDataType of category DATA_REFERENCE is "used" the target ImplementationDataType it references is not considered used, unless it is indepen-

dently used in its own right.
–Last change on issue 73034 comment 14–

BW-C-Level:

Application	Specification	Bus
3	3	1

1.87 Specification Item SWS_Rte_08803

Trace References:

none

Content:

An AutosarDataType is used if it meets any one of the following conditions:

- it is referenced by a DataPrototype in the InternalBehavior, or
- it is referenced by a VariationPointProxy in the InternalBehavior, or
- it is referenced by a DataPrototype referenced by a BswModuleDescription.provided Data or BswModuleDescription.requiredData, or
- it is referenced by an IncludedDataTypeSet in the InternalBehavior, or
- it is the ImplementationDataType mapped to an ApplicationDataType (i.e. via the DataTypeMappingSet) that is used in one of the above ways, or
- it is an ImplementationDataTypeElement of a complex ImplementationDataType that is used in one of the above ways, or
- it is referenced as the target type of an ImplementationDataType or Implementation DataTypeElement of category TYPE_REFERENCE that is used in one of the above ways, or
- it is an ApplicationDataType referenced as the type of a sub-element of a complex ApplicationDataType that is used in one of the above ways.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #73034: Clarification required regarding applicability of SWS_Rte_03809 - should labels be generated for mapped ImplementationDataTypes?

  Problem description:

  SWS_Rte_03809 states that, for a given ApplicationSoftwareComponent, the constants associated with all ‘used’ AutosarDataTypes must be included in the

Application Types Header. It is noted that AutosarDataTypes referenced from relevant DataPrototypes and AutosarDataTypes referenced from IncludedDataTypeSets are all ‘used’ in this sense and further that sub-element data types of ‘used’ complex data types are also ‘used’, as are data types that are aliased by ‘used’ type reference data types.

What is not entirely clear is whether an ImplementationDataType that is mapped by the component to a ‘used’ ApplicationDataType is to be considered ‘used’ if it is not directly referenced from a DataPrototype or IncludedDataTypeSet itself.

On the one hand, this case is not explicitly included, although it is not clear whether the list is intended to be exhaustive. Indeed it does not mention element types of ‘used’ complex ApplicationDataTypes although it would seem strange if they were not to be included analogously to the ImplementationDataType case that is explicitly included.

On the other hand, the example given in the definition of ‘AI’ within Table 5.39 in the SoftwareComponentTemplate (AR4.2.2) suggests that the labels from both types may be used. Indeed, when a DataPrototype references an ApplicationDataType then the corresponding APIs use the mapped ImplementationDataType so it is reasonable to expect the software component code to use its labels, as well as the aliases from the ApplicationDataType. It could also be argued that the ImplementationDataType /is/ referenced by the component, via the referenced DataTypeMappingSet.

Issue # 72232 has clarified that in the case of labels associated with both an ApplicationDataType and its mapped ImplementationDataType an IncludedDataTypeSet referencing only the ApplicationDataType gives its literalPrefix (by SWS_Rte_03810) to the definitions for the ApplicationDataType only but it is not entirely clear whether the mapped ImplementationDataType should be considered ‘used’ at all in this case. Since there is no API or other code artefact resulting from an IncludedDataTypeSet it seems reasonable that different behaviour may be required in this respect compared to the case where the ApplicationDataType is referenced by a DataPrototype.

Having the labels defined for both types ensures that whichever labels the component code uses will be available but the downside is that conflicts between labels of different ImplementationDataTypes may occur and require entries in IncludedDataTypeSets to resolve even if the labels are not actually being used in the code. A mismatch of interpretations in either direction can be worked around using IncludedDataTypeSets but a common understanding is required to reduce this workload on integrators.

Agreed solution:

— SWS RTE —
[SWS_Rte_03809] d The Application Types Header File shall include the definitions of all constants of ImplementationDataTypes and ApplicationDataTypes for each ImplementationDataType/ApplicationDataTypes used (See [SWS_Rte_0xxx1] for the meaning of the term "used") by this software component.

[SWS_Rte_03983] d The The Module Interlink Types Header File shall include the definitions of all constants of ImplementationDataTypes and Application-DataTypes for each ImplementationDataType/ApplicationDataTypes used (See [SWS_Rte_0xxx2] for the meaning of the term "used") by this Basic Software module.

Add [SWS_Rte_0xxx1] in chapter 5.5.5

[SWS_Rte_0xxx1] The meaning of the term "used" with respect to Autosar-DataTypes [
An AutosarDataType is used if it meets any one of the following conditions:

- it is referenced by a DataPrototype in the InternalBehavior, or
- it is referenced by a VariationPointProxy in the InternalBehavior, or
- it is referenced by a DataPrototype in a PortInterface referenced by a PortPrototype, or
- it is referenced by an IncludedDataTypeSet in the InternalBehavior, or
- it is the ImplementationDataType mapped to an ApplicationDataType (i.e. via the DataTypeMappingSet) that is used in one of the above ways, or
- it is an ImplementationDataTypeElement of a complex ImplementationDataType that is used in one of the above ways, or
- it is referenced as the target type of an ImplementationDataType or ImplementationDataTypeElement of category TYPE_REFERENCE that is used in one of the above ways, or
- it is an ApplicationDataType referenced as the type of a sub-element of a complex ApplicationDataType that is used in one of the above ways.
]
Please note that in contrast to the TYPE_REFERENCE case, when an ImplementationDataType of category DATA_REFERENCE is "used" the target ImplementationDataType it references is not considered used, unless it is independently used in its own right.
-
Add [SWS_Rte_0xxx2] in chapter 6.4.3
[SWS_Rte_0xxx2] The meaning of the term "used" with respect to Autosar-DataTypes [

An AutosarDataType is used if it meets any one of the following conditions:

- it is referenced by a DataPrototype in the InternalBehavior, or
- it is referenced by a VariationPointProxy in the internalBehavior, or
- it is referenced by a DataPrototype referenced by a providedData or requiredData, or
- it is referenced by an IncludedDataTypeSet in the InternalBehavior, or
- it is the ImplementationDataType mapped to an ApplicationDataType (i.e. via the DataTypeMappingSet) that is used in one of the above ways, or
- it is an ImplementationDataTypeElement of a complex ImplementationDataType that is used in one of the above ways, or
- it is referenced as the target type of an ImplementationDataType or ImplementationDataTypeElement of category TYPE_REFERENCE that is used in one of the above ways, or
- it is an ApplicationDataType referenced as the type of a sub-element of a complex ApplicationDataType that is used in one of the above ways.

]
Please note that in contrast to the TYPE_REFERENCE case, when an ImplementationDataType of category DATA_REFERENCE is "used" the target ImplementationDataType it references is not considered used, unless it is independently used in its own right.
–Last change on issue 73034 comment 14–

BW-C-Level:

Application	Specification	Bus
3	3	1

1.88 Specification Item SWS_Rte_CONSTR_03870

Trace References:

none

Content:

In case that RteGeneration.RteDevErrorDetectUninit is configured to true, RteGeneration.RteDevErrorDetect shall be configured to true.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #75045: [Cleanup] Cleanup Constraint Handling in SWS items: Rollout replacement of dependencies by constraints

Problem description:

AUTOSAR specified in the field "Dependency" in the chapter 10 description in an informal way dependencies between different configuration parameters. On the other hand some constraints are created in single BSW modules with the same namespace as the metamodel constraints. This should be cleaned up (see presentation in the attachment).

Agreed solution:

Rollout to replace in each SWS the dependencies by constraints (could be done step by step, not mandatory in one release)

When all SWS specifications are cleaned up the dependency field can be removed from the database

FIM
—-
Remove dependency and in the following configuration parameters and remove the following text from the description field "At least one FiMInhSumRef or FiMInhEventRef or
FiMInhComponentRef needs to be configured." :

FiMInhEventRef [ECUC_FiM_00100]
FiMInhSumRef [ECUC_FiM_00102]
FiMInhComponentRef [ECUC_FiM_00605]

Create chapter 7.4 Configuration Constraints
Add new constraints :
SWS_FIM_CONSTR_XXX1 : For each configured FiMInhibitionConfiguration, at least one of FiMInhSumRef or FiMInhEventRef or FiMInhComponentRef shall be configured.

DEM
—-
see file under svn.:
https://svn.autosar.org/repos/work/09_WorkPackages/WP-A4/70_InternalDocuments/048_Dependency_Removal/DCM_Dem_Dependency_Removal.xlsx

DCM
—-
See file under svn :
https://svn.autosar.org/repos/work/09_WorkPackages/WP-

A4/70_InternalDocuments/048_Dependency_Removal/DCM_Dem_Dependency_Removal.xlsx

Rte
—-
RteDevErrorDetectUninit: SWS_RTE_CONSTR_XXX1: In case that RteDevErrorDetectUninit is configured to true, RteDevErrorDetect shall be configured to true.
–Last change on issue 75045 comment 21–

BW-C-Level:

Application	Specification	Bus
1	1	1