Document Title	ASWS_TransformerGeneral: 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 ASWS_TransformerGeneral

1.1 Specification Item ECUC_Xfrm_00012

Trace References:

none

Content:

Container Name	XfrmGeneralXfrmGeneral
Description	Contains the general configuration parameters of the module.
Configuration Parameters	

Included parameters:

Included Parameters	
Parameter Name	SWS Item ID
XfrmDevErrorDetect	ECUC_Xfrm_00013
XfrmInstanceId	ECUC_Xfrm_00020
XfrmVersionInfoApi	ECUC_Xfrm_00019

Included containers:

No Included Containers

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77845: [diverse] Configuration of instance ID for instantiated modules

  Problem description:

  Some modules that can exist multiple times in an AUTOSAR BSW stack have configurable instance IDs that are used to e.g. call DET. Examples are the bus drivers. Others, like the CDD, Crypto driver, or DIO driver, lack such a configuration parameter.
  –Last change on issue 77845 comment 2–

  Agreed solution:

  TPS EcuConfigurationSpecification (CDD):

  Add container CddGeneral with one parameter CddInstanceId to Cdd EcucModuleDef

Description: Specifies the InstanceId of this module instance. If only one instance is present it shall have the Id 0.
Multiplicity: 1
Type: EcucIntegerParamDef
Range: 0 .. 255
Default value: -
Post-Build Variant Value: false
Value Configuration Class: Pre-compile time - All Variants
==========
Crypto:

Add new pre-compile integer parameter "CryptoInstanceId" (range 0..255) to the container CryptoGeneral,
Description: "Instance ID of the crypto driver. This ID is used to discern several crypto drivers in case more than one driver is used in the same ECU."
==========
Eep:

Change ECUC_Eep_00189 Description from : "Represents the Index of the driver, used by EA" to
"Specifies the InstanceId of this module instance. If only one instance is present it shall have the Id 0."
==========
Fls:

Add REQ:
SWS_Fls_xxx: If more than one instance of the flash driver is used in an ECU, the individual instances have to be given a unique instance ID. This instance ID shall be configured as the parameter FlsDriverIndex. If only one instance of the flash driver is used in an ECU, this instance shall have the parameter FlsDriverIndex configured as 0.
==========
Wdg:

Change ECUC_Wdg_00117 Description from : "Represents the watchdog driver's ID so that it can be referenced by the watchdog interface." to
"Specifies the InstanceId of this module instance. If only one instance is present it shall have the Id 0."
==========
Xfrm:

Add into the container XfrmGeneral a new parameter XfrmInstanceId to Xfrm EcucModuleDef

Description: Specifies the InstanceId of this module instance. If only one instance is present it shall have the Id 0.
Multiplicity: 1
Type: EcucIntegerParamDef
Range: 0 .. 255
Default value: -
Post-Build Variant Value: false
Value Configuration Class: Pre-compile time - All Variants
–Last change on issue 77845 comment 32–

BW-C-Level:

Application	Specification	Bus
1	3	1

1.2 Specification Item ECUC_Xfrm_00020

Trace References:

none

Content:

Name	XfrmInstanceIdXfrmGeneral.XfrmInstanceId		
Parent Container	XfrmGeneral		
Description	Specifies the InstanceId of this module instance. If only one instance is present it shall have the Id 0.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
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 #77845: [diverse] Configuration of instance ID for instantiated modules

Problem description:

Some modules that can exist multiple times in an AUTOSAR BSW stack have configurable instance IDs that are used to e.g. call DET. Examples are the bus drivers. Others, like the CDD, Crypto driver, or DIO driver, lack such a configuration parameter.
–Last change on issue 77845 comment 2–

Agreed solution:

TPS EcuConfigurationSpecification (CDD):

Add container CddGeneral with one parameter CddInstanceId to Cdd EcucModuleDef
Description: Specifies the InstanceId of this module instance. If only one instance is present it shall have the Id 0.
Multiplicity: 1
Type: EcucIntegerParamDef
Range: 0 .. 255
Default value: -
Post-Build Variant Value: false
Value Configuration Class: Pre-compile time - All Variants
==========
Crypto:

Add new pre-compile integer parameter "CryptoInstanceId" (range 0..255) to the container CryptoGeneral,
Description: "Instance ID of the crypto driver. This ID is used to discern several crypto drivers in case more than one driver is used in the same ECU."
==========
Eep:

Change ECUC_Eep_00189 Description from : "Represents the Index of the driver, used by EA" to
"Specifies the InstanceId of this module instance. If only one instance is present it shall have the Id 0."
==========
Fls:

Add REQ:
SWS_Fls_xxx: If more than one instance of the flash driver is used in an ECU, the individual instances have to be given a unique instance ID. This instance ID shall be configured as the parameter FlsDriverIndex. If only one instance of the flash driver is used in an ECU, this instance shall have the parameter FlsDriverIndex configured

as 0.
==========
Wdg:

Change ECUC_Wdg_00117 Description from : "Represents the watchdog driver's ID so that it can be referenced by the watchdog interface." to
"Specifies the InstanceId of this module instance. If only one instance is present it shall have the Id 0."
==========
Xfrm:

Add into the container XfrmGeneral a new parameter XfrmInstanceId to Xfrm EcucModuleDef
Description: Specifies the InstanceId of this module instance. If only one instance is present it shall have the Id 0.
Multiplicity: 1
Type: EcucIntegerParamDef
Range: 0 .. 255
Default value: -
Post-Build Variant Value: false
Value Configuration Class: Pre-compile time - All Variants
–Last change on issue 77845 comment 32–

BW-C-Level:

Application	Specification	Bus
1	3	1

1.3 Specification Item SWS_Xfrm_00036

Trace References:

SRS_Xfrm_00002

Content:

Service name:	<Mip>_<transformerId>Xfrm_transformerId1
Syntax:	uint8 <Mip>_<transformerId>(uint8* buffer, uint32* bufferLength, const <typeparamtype> * dataElement)
Service ID[hex]:	0x03
Sync/Async:	Synchronous
Reentrancy:	Reentrant

Parameters (in):	dataElementXfrm_transformerId1.dataElement	Data element which shall be transformed
Parameters (inout):	None	
Parameters (out):	bufferXfrm_transformerId1.buffer	Buffer allocated by the RTE, where the transformed data has to be stored by the transformer
	bufferLengthXfrm_transformerId1.bufferLength	Used length of the buffer
Return value:	uint8	0x00 (E_OK): Transformation successful 0x01 - 0xff: Specific errors
Description:	This function is the interface of the first transformer in a transformer chain of Sender/Receiver communication. The length of the transformed data shall be calculated by the transformer during runtime and returned in the OUT parameter bufferLength. It may be smaller than the maximum buffer size used by the RTE for buffer allocation.	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #68035: [diverse] Introduce rules defining which input parameters shall be passed per value and which ones per const reference

  Problem description:

  SWS_BSW_00186 especially states that input pointer parameters shall use the const qualifier (i.e., shall be P2CONST).

  In addition to that there shall be a SWS item that states that input parameters of integral and enum type shall be passed by value whereas input parameters of structure type shall be passed by reference.

  The various transformer SWS documents shall be adapted accordingly.
  –Last change on issue 68035 comment 4–

  Agreed solution:

  BSW UML model
  ————-
  The attachment "Changed Proposal in WP-A meeting" contains a list of changes to the APIs in the model (see column H). Afterwards all related documents (included in impact list) shall update their
  generated artifacts.

  General Requirements on Basic Software Modules
  ~~~~~~
  ~~~~~~

Introduce the following requirements prior to SRS_BSW_00371:

SRS_BSW_xxxxx: Input parameters of scalar and enum types shall be passed as a value.
Type: valid
Description: All input parameters of scalar or enum type shall be passed as a value.
Rationale:

Use case: For example a function named <Mip>_SomeFunction with a return type of Std_ReturnType and a single parameter named SomeParameter of type uint8 is defined with the following signature:

Std_ReturnType <Mip>_SomeFunction(uint8 SomeParameter);
Dependencies: –
Supporting Material: —

SRS_BSW_yyyyy: Input parameters of structure type shall be passed as a reference to a constant structure
Type: valid
Description: All input parameters of structure type shall be passed as a reference constant structure
Rationale: Passing input parameters of structure type by value would result in additional run-time overhead due to efforts for copying the whole structure.
Use case: For example a function named <Mip>_SomeFunction with a return type of Std_ReturnType and a single parameter named SomeParameter of type SomeStructure (where SomeStructure is a struct) is defined with the following signature:

Std_ReturnType <Mip>_SomeFunction(P2CONST(SomeStructure, AUTOMATIC, <MIP>_APPL_DATA) SomeParameter);
Dependencies: –
Supporting Material: —

SRS_BSW_zzzzz: Input parameters of array type shall be passed as a reference to the constant array base type
Type: valid
Description: All input parameters of array type shall be passed as a reference to the constant array base type
Rationale: This effectively matches the behavior specified in the ISO-C:90 namely that a "declaration of a parameter as 'array of type' shall be adjusted to 'qualified pointer to

type'".
Use case: For example a function named <Mip>_SomeFunction with a return type of Std_ReturnType and a single parameter named SomeParameter of type array of uint8 is defined with the following signature:

Std_ReturnType <Mip>_SomeFunction(P2CONST(uint8, AUTOMATIC, <MIP>_APPL_DATA) SomeParameter);
Dependencies: –
Supporting Material: —

General Specification of Transformers
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

In SWS_Xfrm_00036 change

const <type>* dataElement

to

<paramtype> dataElement

and add the following to the where clause after the API table after the bullet "type is data type of the data element
"

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).

In SWS_Xfrm_00038 change

[<type> data_1,] ...
[<type> data_n]

to

[<paramtype> data_1,] ...
[<paramtype> data_n]

and add the following to the where clause after the API table after the bullet
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

"type is data type of the data element
"

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).

The following paragraph shall then be removed:

For the arguments of ClientServerOperation which are handed over to the transformer as data_1, ..., data_n the requirements to API parameters stated in chapter API Parameters of [5, SWS RTE] are valid (especially [SWS_Rte_01017], [SWS_Rte_01018] and [SWS_Rte_05107]).

In SWS_Xfrm_00040 change

[<originalData1>, ...
<originalDataN>]

to

[<paramtype> originalData1,] ...
[<paramtype> originalDataN]

and add the following to the where clause after the API table after the bullet "type is data type of the data element
"

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).

In SWS_Xfrm_00044 change

<type> *data_1, ...
<type> *data_n

to

[<paramtype> data_1,] ...
[<paramtype> data_n]

and add the following to the where clause after the API table after the bullet "type is data type of the data element
"

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).

The following paragraph shall then be removed:

For the arguments of ClientServerOperation which are handed over to the transformer as data_1, ..., data_n the requirements to API parameters stated in chapter API Parameters of [5, SWS RTE] are valid (especially [SWS_Rte_01017], [SWS_Rte_01018] and [SWS_Rte_05107]).

Speci?cation of SOME/IP Transformer
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

In SWS_SomeIpXf_00138 change

const <type>* dataElement

to

<paramtype> dataElement

and add the following to the where clause after the API table after the bullet "type is data type of the data element
"

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

In SWS_SomeIpXf_00141 change

[<type> data_1,] ...
[<type> data_n]

to

[<paramtype> data_1,] ...
[<paramtype> data_n]

and add the following to the where clause after the API table after the bullet "type is data type of the data element
"

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).

The following paragraph shall then be removed:

For the arguments of ClientServerOperation which are handed over to the transformer as data_1, ..., data_n the requirements to API parameters stated in chapter API Parameters of [5, SWS RTE] are valid (especially [SWS_Rte_01017], [SWS_Rte_01018] and [SWS_Rte_05107]).

In SWS_SomeIpXf_00145 change

<type> *data_1, ...
<type> *data_n

to

[<paramtype> data_1,] ...
[<paramtype> data_n]

and add the following to the where clause after the API table after the bullet "type is data type of the data element
"

<paramtype> is derived from <type> according to the parameter passing rules

rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).

The following paragraph shall then be removed:

For the arguments of ClientServerOperation which are handed over to the transformer as data_1, ..., data_n the requirements to API parameters stated in chapter API Parameters of [5, SWS RTE] are valid (especially [SWS_Rte_01017], [SWS_Rte_01018] and [SWS_Rte_05107]).

Specification of COM Based Transformer
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

In SWS_ComXf_00007 change

const <type>* dataElement

to

<paramtype> dataElement

and add the following to the where clause after the API table after the bullet "type is data type of the data element
"

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).

Specification of Time Sync over Ethernet
~~~~~~

In SWS_EthTSyn_00040 make the parameter DataPtr of EthTSyn_RxIndication const.

Specification of SWS FlexRay Interface
~~~~~~

Change SWS_FrIf_05073 from
FrIf_NumOfStartupFramesPtr (IN)
to
FrIf_NumOfStartupFramesPtr (OUT)

Specification of ADC
~~~~~~
~[SWS_Adc_00419] Adc_SetupResultBuffer: change Adc_ValueGroupType* to const Adc_ValueGroupType*
~[SWS_Adc_00369] Adc_ReadGroup: move Adc_ValueGroupType * from Parameters (in) to Parameters (out)

There is no need to change parameter from IN to INOUT in Adc_SetupResultBuffer

Specification of Com
~~~~~~
Change type of parameter MetaData of Com_TriggerIPDUSendWithMetaData from uint8* to const uint8*

Specification of ComM
~~~~~~
no change required

Specification of Dem
~~~~~~
no change required

Specification of DLT
~~~~~~
no change required

Specification of DoIP
~~~~~~
From:
~~~~~~

Std_ReturnType <User>_DoIPRoutingActivationConfirmation(boolean* Confirmed, uint8* ConfirmationReqData, uint8* ConfirmationResData)
Std_ReturnType <User>_DoIPRoutingActivationAuthentication(boolean* Authentified, uint8* AuthenticationReqData, uint8* AuthenticationResData)

To:
Std_ReturnType <User>_DoIPRoutingActivationConfirmation(boolean* Confirmed, const uint8* ConfirmationReqData, uint8* ConfirmationResData)
Std_ReturnType <User>_DoIPRoutingActivationAuthentication(boolean* Authentified, const uint8* AuthenticationReqData, uint8* AuthenticationResData)

Specification of E2ELibrary
~~~~~~
no change required

Specification of Eth
~~~~~~
no change required

Specification of EthIf
~~~~~~
no change required

Specification of EthSwitchDriver
~~~~~~
no change required

Specification of ICUDriver
~~~~~~
SWS_Icu_00201: Icu_StartTimestamp
Parameter (IN): Icu_ValueType* BufferPtr shall be changed to Parameters (out) type

Specification of LdCom
~~~~~~
[SWS_LDCOM_00027]: LdCom_CopyTxData
BufReq_ReturnType LdCom_CopyTxData(PduIdType id, const PduInfoType* info, RetryInfoType* retry, PduLengthType* availableDataPtr) shall be changed to

BufReq_ReturnType LdCom_CopyTxData(PduIdType id, const PduInfoType* info, const RetryInfoType* retry, PduLengthType* availableDataPtr)

[SWS_LDCOM_00036]: Rte_LdComCbkCopyTxData_<sn>
BufReq_ReturnType Rte_LdComCbkCopyTxData_<sn>(const PduInfoType* info, RetryInfoType* retry, PduLengthType* availableDataPtr) shall be changed to
BufReq_ReturnType Rte_LdComCbkCopyTxData_<sn>(const PduInfoType* info, const RetryInfoType* retry, PduLengthType* availableDataPtr)

Specification of Lin
~~~~~~
PduInfoPtr needs to be const in Std_ReturnType Lin_SendFrame(uint8 Channel, const Lin_PduType* PduInfoPtr)

Specification of PduR
~~~~~~
* PduR_<User:LoTp>CopyTxData
add const to "RetryInfoType* retry"

Specification of J1939Nm
~~~~~~
Change parameter 'name' of User_AddressClaimedIndication to type 'const uint8*'

Specification of SoAd
~~~~~~
=> everything already fixed with RfC 65633

Specification of SPIHandlerDriver
~~~~~~
==> nothing to change for SWS SPI

Specification of SynchronizedTimeBaseManager
~~~~~~
"StbM not affected. All issues listed in the WP-A attachment have been already implemented by IT 69124 in context of RfC 65633"

Specification of TcpIp
~~~~~~
~[SWS_TCPIP_00040] TcpIp_DhcpReadOption: change DataPtr from (IN) to (OUT)
~[SWS_TCPIP_00189] TcpIp_DhcpV6ReadOption: change DataPtr from (IN) to (OUT)
=> everything else already fixed with RfC 65633

Specification of TimeSyncOverFlexRay
~~~~~~
"Change SWS_FrTSyn_00064: parameter versioninfo of type Std_VersionInfoType* is marked wrongly as IN. Change to OUT"

Specification of EFX
~~~~~~~~~~~~~~~~~~~~~~
~ [SWS_Efx_00355] Efx_Debounce_u8_u8: Include constant for pointer Input-parameter as like below.
uint8 Efx_Debounce_u8_u8(boolean X, Efx_DebounceState_Type * State, const Efx_DebounceParam_Type * Param, sint32 dT)

~ [SWS_Efx_00376] Efx_MedianSort: The parameter <InType>* Array should be InOut instead of In parameter as like below.
Parameters (in): N Size of an array
Parameters (inout): Array Pointer to an array

~ [SWS_Efx_00309] Efx_RampCheckActivity: Include constant for pointer Input-parameter as like below.
boolean Efx_RampCheckActivity(const Efx_StateRamp_Type* State_cpst)

~ [SWS_Efx_00307] Efx_RampGetSwitchPos: Include constant for pointer Input-parameter as like below.
boolean Efx_RampGetSwitchPos(const Efx_StateRamp_Type* State_cpst)

~ [SWS_Efx_00193] Efx_Array_Average: Include constant for pointer Input-parameter as like below.
<OutType> Efx_Array_Average_<InTypeMn>_<OutTypeMn>(const <InType>* Array, uint16 Count)

Specification of MFL
~~~~~~~~~~~~~~~~~~~~~~

~ [SWS_Mfl_00192] Mfl_Debounce_u8_u8: Include constant for pointer Input-parameter as like below.
boolean Mfl_Debounce_u8_u8(boolean X, Mfl_DebounceState_Type* State, const Mfl_DebounceParam_Type* Param, float32 dT)

~ [SWS_Mfl_00266] Mfl_DebounceInit: The parameter Mfl_DebounceState_Type* State should be Out instead of In parameter as like below.
Parameters (in): X Initial value for the input state
Parameters (out): State Pointer to structure for debouncing state variables

~ [SWS_Mfl_00246] Mfl_HystDeltaRight_f32_u8: Include constant for pointer Input-parameter as like below.
boolean Mfl_HystDeltaRight_f32_u8(float32 X, float32 Delta, float32 Rsp, const uint8* State)

~ [SWS_Mfl_00285] Mfl_MedianSort_f32_f32: The parameter Array should be InOut instead of In parameter as like below.
Parameters (in): N Size of an array
Parameters (inout): Array Pointer to an array

~ [SWS_Mfl_00037] Mfl_PT1SetState: The parameter State_cpst should be Out instead of In parameter as like below.
Parameters (in): X1_f32 Initial value for input state
Y1_f32 Initial value for output state
Parameters (out): State_cpst Pointer to internal state structure

~ [SWS_Mfl_00225] Mfl_RampCheckActivity: Include constant for pointer Input-parameter as like below.
boolean Mfl_RampCheckActivity(const Mfl_StateRamp_Type* State_cpst)

~ [SWS_Mfl_00223] Mfl_RampGetSwitchPos: Include constant for pointer Input-parameter as like below.
boolean Mfl_RampGetSwitchPos(const Mfl_StateRamp_Type* State_cpst)
–Last change on issue 68035 comment 135–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.4 Specification Item SWS_Xfrm_00038

Trace References:

SRS_Xfrm_00002

Content:

Service name:	<Mip>_<transformerId>Xfrm_transformerId2	
Syntax:	uint8 <Mip>_<transformerId>(const Rte_Cs_TransactionHandleType* TransactionHandle, uint8* buffer, uint32* bufferLength, [Std_ReturnType returnValue,] [<typeparamtype> data_1,] ... [<typeparamtype> data_n])	
Service ID[hex]:	0x03	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	TransactionHandleXfrm_transformerId2.TransactionHandle	Transaction handle according to [SWS_Rte_08732] (clientId and sequenceCounter) needed to differentiate between multiple requests.
	returnValueXfrm_transformerId2.returnValue	Return value of the server runnable which needs to be transformed on server side for transmission to the calling client. This argument is only available for serializers of the response of a Client/Server communication and if the ClientServerOperation has at least one PossibleError defined.
	data_1Xfrm_transformerId2.data_1	Client/Server operation argument which shall be transformed (in the same order as in the corresponding interface)
...Xfrm_transformerId2....	...	
data_nXfrm_transformerId2.data_n	Client/Server operation argument which shall be transformed (in the same order as in the corresponding interface)	
Parameters (inout):	None	
Parameters (out):	bufferXfrm_transformerId2.buffer	Buffer allocated by the RTE, where the transformed data has to be stored by the transformer
	bufferLengthXfrm_transformerId2.bufferLength	Used length of the buffer
Return value:	uint8	0x00 (E_OK): Transformation successful 0x01 - 0xff: Specific errors
Description:	This function is the interface of the first transformer in a transformer chain of Client/Server communication. It takes the operation arguments and optionally the return value as input and outputs an a uint8 array containing the transformed data. The length of the transformed data shall be calculated by the transformer during runtime and returned in the OUT parameter bufferLength. It may be smaller than the maximum buffer size used by the RTE for buffer allocation.	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #68035: [diverse] Introduce rules defining which input parameters shall be passed per value and which ones per const reference

  Problem description:

  SWS_BSW_00186 especially states that input pointer parameters shall use the const qualifier (i.e., shall be P2CONST).

  In addition to that there shall be a SWS item that states that input parameters of integral and enum type shall be passed by value whereas input parameters of structure type shall be passed by reference.

  The various transformer SWS documents shall be adapted accordingly.
  –Last change on issue 68035 comment 4–

  Agreed solution:

  BSW UML model
  ————-
  The attachment "Changed Proposal in WP-A meeting" contains a list of changes to the APIs in the model (see column H). Afterwards all related documents (included in impact list) shall update their
  generated artifacts.

  General Requirements on Basic Software Modules
  ~~~~~~

  Introduce the following requirements prior to SRS_BSW_00371:

  SRS_BSW_xxxxx: Input parameters of scalar and enum types shall be passed as a value.
  Type: valid
  Description: All input parameters of scalar or enum type shall be passed as a value.
  Rationale:

  Use case: For example a function named <Mip>_SomeFunction with a return type of Std_ReturnType and a single parameter named SomeParameter of type uint8 is defined with the following signature:

  Std_ReturnType <Mip>_SomeFunction(uint8 SomeParameter);
  Dependencies: –
  ~~~~~~

Supporting Material: —

SRS_BSW_yyyyy: Input parameters of structure type shall be passed as a reference to a constant structure
Type: valid
Description: All input parameters of structure type shall be passed as a reference constant structure
Rationale: Passing input parameters of structure type by value would result in additional run-time overhead due to efforts for copying the whole structure.
Use case: For example a function named <Mip>_SomeFunction with a return type of Std_ReturnType and a single parameter named SomeParameter of type SomeStructure (where SomeStructure is a struct) is defined with the following signature:

Std_ReturnType <Mip>_SomeFunction(P2CONST(SomeStructure, AUTOMATIC, <MIP>_APPL_DATA) SomeParameter);
Dependencies: –
Supporting Material: —

SRS_BSW_zzzzz: Input parameters of array type shall be passed as a reference to the constant array base type
Type: valid
Description: All input parameters of array type shall be passed as a reference to the constant array base type
Rationale: This effectively matches the behavior specified in the ISO-C:90 namely that a "declaration of a parameter as 'array of type' shall be adjusted to 'qualified pointer to
type'".
Use case: For example a function named <Mip>_SomeFunction with a return type of Std_ReturnType and a single parameter named SomeParameter of type array of uint8 is defined with the following signature:

Std_ReturnType <Mip>_SomeFunction(P2CONST(uint8, AUTOMATIC, <MIP>_APPL_DATA) SomeParameter);
Dependencies: –
Supporting Material: —

General Specification of Transformers
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

In SWS_Xfrm_00036 change

const <type>* dataElement

to

<paramtype> dataElement

and add the following to the where clause after the API table after the bullet "type is data type of the data element
"

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).

In SWS_Xfrm_00038 change

[<type> data_1,] ...
[<type> data_n]

to

[<paramtype> data_1,] ...
[<paramtype> data_n]

and add the following to the where clause after the API table after the bullet "type is data type of the data element
"

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).

The following paragraph shall then be removed:

For the arguments of ClientServerOperation which are handed over to the transformer as data_1, ..., data_n the requirements to API parameters stated in chapter API Parameters of [5, SWS RTE] are valid (especially [SWS_Rte_01017],

[SWS_Rte_01018] and [SWS_Rte_05107]).

In SWS_Xfrm_00040 change

[<originalData1>, ...
<originalDataN>]

to

[<paramtype> originalData1,] ...
[<paramtype> originalDataN]

and add the following to the where clause after the API table after the bullet "type is data type of the data element
"

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).

In SWS_Xfrm_00044 change

<type> *data_1, ...
<type> *data_n

to

[<paramtype> data_1,] ...
[<paramtype> data_n]

and add the following to the where clause after the API table after the bullet "type is data type of the data element
"

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).

The following paragraph shall then be removed:

For the arguments of ClientServerOperation which are handed over to the transformer as data_1, ..., data_n the requirements to API parameters stated in chapter API Parameters of [5, SWS RTE] are valid (especially [SWS_Rte_01017], [SWS_Rte_01018] and [SWS_Rte_05107]).

Speci?cation of SOME/IP Transformer
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

In SWS_SomeIpXf_00138 change

const <type>* dataElement

to

<paramtype> dataElement

and add the following to the where clause after the API table after the bullet "type is data type of the data element
"

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).

In SWS_SomeIpXf_00141 change

[<type> data_1,] ...
[<type> data_n]

to

[<paramtype> data_1,] ...
[<paramtype> data_n]

and add the following to the where clause after the API table after the bullet "type is data type of the data element
"
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).

The following paragraph shall then be removed:

For the arguments of ClientServerOperation which are handed over to the transformer as data_1, ..., data_n the requirements to API parameters stated in chapter API Parameters of [5, SWS RTE] are valid (especially [SWS_Rte_01017], [SWS_Rte_01018] and [SWS_Rte_05107]).

In SWS_SomeIpXf_00145 change

<type> *data_1, ...
<type> *data_n

to

[<paramtype> data_1,] ...
[<paramtype> data_n]

and add the following to the where clause after the API table after the bullet "type is data type of the data element
"

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).

The following paragraph shall then be removed:

For the arguments of ClientServerOperation which are handed over to the transformer as data_1, ..., data_n the requirements to API parameters stated in chapter API Parameters of [5, SWS RTE] are valid (especially [SWS_Rte_01017], [SWS_Rte_01018] and [SWS_Rte_05107]).

Specification of COM Based Transformer

~~~~~~

In SWS_ComXf_00007 change

const <type>* dataElement

to

<paramtype> dataElement

and add the following to the where clause after the API table after the bullet "type is data type of the data element
"

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).

Specification of Time Sync over Ethernet
~~~~~~

In SWS_EthTSyn_00040 make the parameter DataPtr of EthTSyn_RxIndication const.

Specification of SWS FlexRay Interface
~~~~~~

Change SWS_FrIf_05073 from
FrIf_NumOfStartupFramesPtr (IN)
to
FrIf_NumOfStartupFramesPtr (OUT)

Specification of ADC
~~~~~~

~[SWS_Adc_00419] Adc_SetupResultBuffer: change Adc_ValueGroupType* to const Adc_ValueGroupType*
~[SWS_Adc_00369] Adc_ReadGroup: move Adc_ValueGroupType * from Parame-

ters (in) to Parameters (out)

There is no need to change parameter from IN to INOUT in Adc_SetupResultBuffer

Specification of Com
~~~~~~
Change type of parameter MetaData of Com_TriggerIPDUSendWithMetaData from uint8* to const uint8*

Specification of ComM
~~~~~~
no change required

Specification of Dem
~~~~~~
no change required

Specification of DLT
~~~~~~
no change required

Specification of DoIP
~~~~~~
From:
Std_ReturnType <User>_DoIPRoutingActivationConfirmation(boolean* Confirmed, uint8* ConfirmationReqData, uint8* ConfirmationResData)
Std_ReturnType <User>_DoIPRoutingActivationAuthentication(boolean* Authentified, uint8* AuthenticationReqData, uint8* AuthenticationResData)

To:
Std_ReturnType <User>_DoIPRoutingActivationConfirmation(boolean* Confirmed, const uint8* ConfirmationReqData, uint8* ConfirmationResData)
Std_ReturnType <User>_DoIPRoutingActivationAuthentication(boolean* Authentified, const uint8* AuthenticationReqData, uint8* AuthenticationResData)

Specification of E2ELibrary
~~~~~~

no change required

Specification of Eth
~~~~~~
no change required

Specification of EthIf
~~~~~~
no change required

Specification of EthSwitchDriver
~~~~~~
no change required

Specification of ICUDriver
~~~~~~
SWS_Icu_00201: Icu_StartTimestamp
Parameter (IN): Icu_ValueType* BufferPtr shall be changed to Parameters (out) type

Specification of LdCom
~~~~~~
[SWS_LDCOM_00027]: LdCom_CopyTxData
BufReq_ReturnType LdCom_CopyTxData(PduIdType id, const PduInfoType* info, RetryInfoType* retry, PduLengthType* availableDataPtr) shall be changed to
BufReq_ReturnType LdCom_CopyTxData(PduIdType id, const PduInfoType* info, const RetryInfoType* retry, PduLengthType* availableDataPtr)

[SWS_LDCOM_00036]: Rte_LdComCbkCopyTxData_<sn>
BufReq_ReturnType Rte_LdComCbkCopyTxData_<sn>(const PduInfoType* info, RetryInfoType* retry, PduLengthType* availableDataPtr) shall be changed to
BufReq_ReturnType Rte_LdComCbkCopyTxData_<sn>(const PduInfoType* info, const RetryInfoType* retry, PduLengthType* availableDataPtr)

Specification of Lin
~~~~~~
PduInfoPtr needs to be const in Std_ReturnType Lin_SendFrame(uint8 Channel, const Lin_PduType* PduInfoPtr)

Specification of PduR
~~~~~~
* PduR_<User:LoTp>CopyTxData
add const to "RetryInfoType* retry"

Specification of J1939Nm
~~~~~~
Change parameter 'name' of User_AddressClaimedIndication to type 'const uint8*'

Specification of SoAd
~~~~~~
=> everything already fixed with RfC 65633

Specification of SPIHandlerDriver
~~~~~~
==> nothing to change for SWS SPI

Specification of SynchronizedTimeBaseManager
~~~~~~
"StbM not affected. All issues listed in the WP-A attachment have been already implemented by IT 69124 in context of RfC 65633"

Specification of TcpIp
~~~~~~
~[SWS_TCPIP_00040] TcpIp_DhcpReadOption: change DataPtr from (IN) to (OUT)
~[SWS_TCPIP_00189] TcpIp_DhcpV6ReadOption: change DataPtr from (IN) to (OUT)
=> everything else already fixed with RfC 65633

Specification of TimeSyncOverFlexRay
~~~~~~
"Change SWS_FrTSyn_00064: parameter versioninfo of type Std_VersionInfoType* is marked wrongly as IN. Change to OUT"
~~~~~~

Specification of EFX
~~~~~~~~~~~~~~~~~~~~~

~ [SWS_Efx_00355] Efx_Debounce_u8_u8: Include constant for pointer Input-parameter as like below.
uint8 Efx_Debounce_u8_u8(boolean X, Efx_DebounceState_Type * State, const Efx_DebounceParam_Type * Param, sint32 dT)

~ [SWS_Efx_00376] Efx_MedianSort: The parameter <InType>* Array should be InOut instead of In parameter as like below.
Parameters (in): N Size of an array
Parameters (inout): Array Pointer to an array

~ [SWS_Efx_00309] Efx_RampCheckActivity: Include constant for pointer Input-parameter as like below.
boolean Efx_RampCheckActivity(const Efx_StateRamp_Type* State_cpst)

~ [SWS_Efx_00307] Efx_RampGetSwitchPos: Include constant for pointer Input-parameter as like below.
boolean Efx_RampGetSwitchPos(const Efx_StateRamp_Type* State_cpst)

~ [SWS_Efx_00193] Efx_Array_Average: Include constant for pointer Input-parameter as like below.
<OutType> Efx_Array_Average_<InTypeMn>_<OutTypeMn>(const <InType>* Array, uint16 Count)

Specification of MFL
~~~~~~~~~~~~~~~~~~~~~

~ [SWS_Mfl_00192] Mfl_Debounce_u8_u8: Include constant for pointer Input-parameter as like below.
boolean Mfl_Debounce_u8_u8(boolean X, Mfl_DebounceState_Type* State, const Mfl_DebounceParam_Type* Param, float32 dT)

~ [SWS_Mfl_00266] Mfl_DebounceInit: The parameter Mfl_DebounceState_Type* State should be Out instead of In parameter as like below.
Parameters (in): X Initial value for the input state
Parameters (out): State Pointer to structure for debouncing state variables

~ [SWS_Mfl_00246] Mfl_HystDeltaRight_f32_u8: Include constant for pointer Input-parameter as like below.
boolean Mfl_HystDeltaRight_f32_u8(float32 X, float32 Delta, float32 Rsp, const uint8* State)

~ [SWS_Mfl_00285] Mfl_MedianSort_f32_f32: The parameter Array should be InOut instead of In parameter as like below.
Parameters (in): N Size of an array
Parameters (inout): Array Pointer to an array

~ [SWS_Mfl_00037] Mfl_PT1SetState: The parameter State_cpst should be Out instead of In parameter as like below.
Parameters (in): X1_f32 Initial value for input state
Y1_f32 Initial value for output state
Parameters (out): State_cpst Pointer to internal state structure

~ [SWS_Mfl_00225] Mfl_RampCheckActivity: Include constant for pointer Input-parameter as like below.
boolean Mfl_RampCheckActivity(const Mfl_StateRamp_Type* State_cpst)

~ [SWS_Mfl_00223] Mfl_RampGetSwitchPos: Include constant for pointer Input-parameter as like below.
boolean Mfl_RampGetSwitchPos(const Mfl_StateRamp_Type* State_cpst)
–Last change on issue 68035 comment 135–

BW-C-Level:

Application	Specification	Bus
1	4	1

- RfC #74087: Change "an uint" to "a uint"

  Problem description:

  Remainder from # 73404:
  The affected documents contain text generated artefacts which contain the text "an uint".

  Correct is "a uint".

  The changes of the artefacts need changes in Metamodel and BSW UML Model.

  Agreed solution:

  Change "an uint" to "a uint" in metamodel artifacts.
  –Last change on issue 74087 comment 2–

  BW-C-Level:

Application	Specification	Bus
1	1	1

1.5 Specification Item SWS_Xfrm_00040

Trace References:

SRS_Xfrm_00002

Content:

Service name:	<Mip>_<transformerId>Xfrm_transformerId3	
Syntax:	uint8 <Mip>_<transformerId>(uint8* buffer, uint32* bufferLength, [const uint8* inputBuffer,] uint32 inputBufferLength, [<typeparamtype> <originalData>_1,] ... [<typeparamtype> <originalData>_n])	
Service ID[hex]:	0x03	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	inputBufferXfrm_transformerId3.input Buffer	This argument only exists for transformers configured for out-of-place transformation. It holds the input data for the transformer.
	inputBufferLengthXfrm_transformer Id3.inputBufferLength	This argument holds the length of the transformer's input data (in the input Buffer argument).
	<originalData>_1Xfrm_transformer Id3.<originalData>_1	These arguments only exists for transformers on the sending side that are configured for access to the original data. - This denotes the data element represented by the VariableData Prototype if a Sender/Receiver communication is transformed. - This denotes all arguments of the Client ServerOperation if a Client/Server communication is transformed.
...Xfrm_transformerId3....	...	
<originalData>_nXfrm_transformer Id3.<originalData>_n	These arguments only exists for transformers on the sending side that are configured for access to the original data. - This denotes the data element represented by the VariableData Prototype if a Sender/Receiver communication is transformed. - This denotes all arguments of the Client ServerOperation if a Client/Server communication is transformed.	

Parameters (inout):	bufferXfrm_transformerId3.buffer	This argument is only an INOUT argument for transformers which are not configured for out-of-place transformation. It is the buffer where the input data are placed by the RTE and which is filled by the transformer with its output. This parameter points to the buffer with the output of the previous transformer. If the current transformer has a headerLength different from 0, the output data of the previous transformer begin at position headerLength.
Parameters (out):	bufferXfrm_transformerId3.buffer	This argument is only an OUT argument for transformers configured for out-of-place transformation. It is the buffer allocated by the RTE, where the transformed data has to be stored by the transformer.
	bufferLengthXfrm_transformerId3.bufferLength	Used length of the buffer
Return value:	uint8	0x00 (E_OK): Transformation successful 0x01 - 0xff: Specific errors
Description:	This function is the interface of the first transformer in a transformer chain of Sender/Receiver communication. The length of the transformed data shall be calculated by the transformer during runtime and returned in the OUT parameter bufferLength. It may be smaller than the maximum buffer size used by the RTE for buffer allocation.	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #68035: [diverse] Introduce rules defining which input parameters shall be passed per value and which ones per const reference

  Problem description:

  SWS_BSW_00186 especially states that input pointer parameters shall use the const qualifier (i.e., shall be P2CONST).

  In addition to that there shall be a SWS item that states that input parameters of integral and enum type shall be passed by value whereas input parameters of structure type shall be passed by reference.

  The various transformer SWS documents shall be adapted accordingly.
  –Last change on issue 68035 comment 4–

  Agreed solution:

  BSW UML model

  ———-

  The attachment "Changed Proposal in WP-A meeting" contains a list of changes to

the APIs in the model (see column H). Afterwards all related documents (included in impact list) shall update their
generated artifacts.

General Requirements on Basic Software Modules
~~~~~~

Introduce the following requirements prior to SRS_BSW_00371:

SRS_BSW_xxxxx: Input parameters of scalar and enum types shall be passed as a value.
Type: valid
Description: All input parameters of scalar or enum type shall be passed as a value.
Rationale:

Use case: For example a function named <Mip>_SomeFunction with a return type of Std_ReturnType and a single parameter named SomeParameter of type uint8 is defined with the following signature:

Std_ReturnType <Mip>_SomeFunction(uint8 SomeParameter);
Dependencies: –
Supporting Material: —

SRS_BSW_yyyyy: Input parameters of structure type shall be passed as a reference to a constant structure
Type: valid
Description: All input parameters of structure type shall be passed as a reference constant structure
Rationale: Passing input parameters of structure type by value would result in additional run-time overhead due to efforts for copying the whole structure.
Use case: For example a function named <Mip>_SomeFunction with a return type of Std_ReturnType and a single parameter named SomeParameter of type SomeStructure (where SomeStructure is a struct) is defined with the following signature:

Std_ReturnType <Mip>_SomeFunction(P2CONST(SomeStructure, AUTOMATIC, <MIP>_APPL_DATA) SomeParameter);
Dependencies: –
Supporting Material: —
~~~~~~

SRS_BSW_zzzzz: Input parameters of array type shall be passed as a reference to the constant array base type
Type: valid
Description: All input parameters of array type shall be passed as a reference to the constant array base type
Rationale: This effectively matches the behavior specified in the ISO-C:90 namely that a "declaration of a parameter as 'array of type' shall be adjusted to 'qualified pointer to
type'".
Use case: For example a function named <Mip>_SomeFunction with a return type of Std_ReturnType and a single parameter named SomeParameter of type array of uint8 is defined with the following signature:

Std_ReturnType <Mip>_SomeFunction(P2CONST(uint8, AUTOMATIC, <MIP>_APPL_DATA) SomeParameter);
Dependencies: –
Supporting Material: —

General Specification of Transformers
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

In SWS_Xfrm_00036 change

const <type>* dataElement

to

<paramtype> dataElement

and add the following to the where clause after the API table after the bullet "type is data type of the data element
"

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).

In SWS_Xfrm_00038 change

[<type> data_1,] ...
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

[<type> data_n]

to

[<paramtype> data_1,] ...
[<paramtype> data_n]

and add the following to the where clause after the API table after the bullet "type is data type of the data element
"

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).

The following paragraph shall then be removed:

For the arguments of ClientServerOperation which are handed over to the transformer as data_1, ..., data_n the requirements to API parameters stated in chapter API Parameters of [5, SWS RTE] are valid (especially [SWS_Rte_01017], [SWS_Rte_01018] and [SWS_Rte_05107]).

In SWS_Xfrm_00040 change

[<originalData1>, ...
<originalDataN>]

to

[<paramtype> originalData1,] ...
[<paramtype> originalDataN]

and add the following to the where clause after the API table after the bullet "type is data type of the data element
"

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).

In SWS_Xfrm_00044 change

<type> *data_1, ...
<type> *data_n

to

[<paramtype> data_1,] ...
[<paramtype> data_n]

and add the following to the where clause after the API table after the bullet "type is data type of the data element
"

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).

The following paragraph shall then be removed:

For the arguments of ClientServerOperation which are handed over to the transformer as data_1, ..., data_n the requirements to API parameters stated in chapter API Parameters of [5, SWS RTE] are valid (especially [SWS_Rte_01017], [SWS_Rte_01018] and [SWS_Rte_05107]).

Speci?cation of SOME/IP Transformer
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

In SWS_SomeIpXf_00138 change

const <type>* dataElement

to

<paramtype> dataElement

and add the following to the where clause after the API table after the bullet "type is data type of the data element
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

"

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).

In SWS_SomeIpXf_00141 change

[<type> data_1,] ...
[<type> data_n]

to

[<paramtype> data_1,] ...
[<paramtype> data_n]

and add the following to the where clause after the API table after the bullet "type is data type of the data element
"

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).

The following paragraph shall then be removed:

For the arguments of ClientServerOperation which are handed over to the transformer as data_1, ..., data_n the requirements to API parameters stated in chapter API Parameters of [5, SWS RTE] are valid (especially [SWS_Rte_01017], [SWS_Rte_01018] and [SWS_Rte_05107]).

In SWS_SomeIpXf_00145 change

<type> *data_1, ...
<type> *data_n

to

[<paramtype> data_1,] ...
[<paramtype> data_n]

and add the following to the where clause after the API table after the bullet "type is data type of the data element
"

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).

The following paragraph shall then be removed:

For the arguments of ClientServerOperation which are handed over to the transformer as data_1, ..., data_n the requirements to API parameters stated in chapter API Parameters of [5, SWS RTE] are valid (especially [SWS_Rte_01017], [SWS_Rte_01018] and [SWS_Rte_05107]).

Specification of COM Based Transformer
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

In SWS_ComXf_00007 change

const <type>* dataElement

to

<paramtype> dataElement

and add the following to the where clause after the API table after the bullet "type is data type of the data element
"

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).

Specification of Time Sync over Ethernet
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

~~~~~~

In SWS_EthTSyn_00040 make the parameter DataPtr of EthTSyn_RxIndication const.

Specification of SWS FlexRay Interface
~~~~~~

Change SWS_FrIf_05073 from
FrIf_NumOfStartupFramesPtr (IN)
to
FrIf_NumOfStartupFramesPtr (OUT)

Specification of ADC
~~~~~~
~[SWS_Adc_00419] Adc_SetupResultBuffer: change Adc_ValueGroupType* to const Adc_ValueGroupType*
~[SWS_Adc_00369] Adc_ReadGroup: move Adc_ValueGroupType * from Parameters (in) to Parameters (out)

There is no need to change parameter from IN to INOUT in Adc_SetupResultBuffer

Specification of Com
~~~~~~
Change type of parameter MetaData of Com_TriggerIPDUSendWithMetaData from uint8* to const uint8*

Specification of ComM
~~~~~~
no change required

Specification of Dem
~~~~~~
no change required

Specification of DLT
~~~~~~
no change required

Specification of DoIP
~~~~~~
From:
Std_ReturnType <User>_DoIPRoutingActivationConfirmation(boolean* Confirmed, uint8* ConfirmationReqData, uint8* ConfirmationResData)
Std_ReturnType <User>_DoIPRoutingActivationAuthentication(boolean* Authentified, uint8* AuthenticationReqData, uint8* AuthenticationResData)

To:
Std_ReturnType <User>_DoIPRoutingActivationConfirmation(boolean* Confirmed, const uint8* ConfirmationReqData, uint8* ConfirmationResData)
Std_ReturnType <User>_DoIPRoutingActivationAuthentication(boolean* Authentified, const uint8* AuthenticationReqData, uint8* AuthenticationResData)

Specification of E2ELibrary
~~~~~~
no change required

Specification of Eth
~~~~~~
no change required

Specification of EthIf
~~~~~~
no change required

Specification of EthSwitchDriver
~~~~~~
no change required

Specification of ICUDriver
~~~~~~
SWS_Icu_00201: Icu_StartTimestamp
~~~~~~

Parameter (IN): Icu_ValueType* BufferPtr shall be changed to Parameters (out) type

Specification of LdCom
~~~~~~
[SWS_LDCOM_00027]: LdCom_CopyTxData
BufReq_ReturnType LdCom_CopyTxData(PduIdType id, const PduInfoType* info, RetryInfoType* retry, PduLengthType* availableDataPtr) shall be changed to
BufReq_ReturnType LdCom_CopyTxData(PduIdType id, const PduInfoType* info, const RetryInfoType* retry, PduLengthType* availableDataPtr)

[SWS_LDCOM_00036]: Rte_LdComCbkCopyTxData_<sn>
BufReq_ReturnType Rte_LdComCbkCopyTxData_<sn>(const PduInfoType* info, RetryInfoType* retry, PduLengthType* availableDataPtr) shall be changed to
BufReq_ReturnType Rte_LdComCbkCopyTxData_<sn>(const PduInfoType* info, const RetryInfoType* retry, PduLengthType* availableDataPtr)

Specification of Lin
~~~~~~
PduInfoPtr needs to be const in Std_ReturnType Lin_SendFrame(uint8 Channel, const Lin_PduType* PduInfoPtr)

Specification of PduR
~~~~~~
* PduR_<User:LoTp>CopyTxData
add const to "RetryInfoType* retry"

Specification of J1939Nm
~~~~~~
Change parameter ’name’ of User_AddressClaimedIndication to type ’const uint8*’

Specification of SoAd
~~~~~~
=> everything already fixed with RfC 65633

Specification of SPIHandlerDriver
~~~~~~
==> nothing to change for SWS SPI

Specification of SynchronizedTimeBaseManager
~~~~~~

"StbM not affected. All issues listed in the WP-A attachment have been already implemented by IT 69124 in context of RfC 65633"

Specification of TcpIp
~~~~~~

~[SWS_TCPIP_00040] TcpIp_DhcpReadOption: change DataPtr from (IN) to (OUT)
~[SWS_TCPIP_00189] TcpIp_DhcpV6ReadOption: change DataPtr from (IN) to (OUT)
=> everything else already fixed with RfC 65633

Specification of TimeSyncOverFlexRay
~~~~~~

"Change SWS_FrTSyn_00064: parameter versioninfo of type Std_VersionInfoType* is marked wrongly as IN. Change to OUT"

Specification of EFX
~~~~~~~~~~~~~~~~~~~~

~ [SWS_Efx_00355] Efx_Debounce_u8_u8: Include constant for pointer Input-parameter as like below.
uint8 Efx_Debounce_u8_u8(boolean X, Efx_DebounceState_Type * State, const Efx_DebounceParam_Type * Param, sint32 dT)

~ [SWS_Efx_00376] Efx_MedianSort: The parameter <InType>* Array should be InOut instead of In parameter as like below.
Parameters (in): N Size of an array
Parameters (inout): Array Pointer to an array

~ [SWS_Efx_00309] Efx_RampCheckActivity: Include constant for pointer Input-parameter as like below.
boolean Efx_RampCheckActivity(const Efx_StateRamp_Type* State_cpst)

~ [SWS_Efx_00307] Efx_RampGetSwitchPos: Include constant for pointer Input-parameter as like below.
boolean Efx_RampGetSwitchPos(const Efx_StateRamp_Type* State_cpst)

~ [SWS_Efx_00193] Efx_Array_Average: Include constant for pointer Input-parameter as like below.
<OutType> Efx_Array_Average_<InTypeMn>_<OutTypeMn>(const <InType>* Array, uint16 Count)

Specification of MFL
~~~~~~~~~~~~~~~~~~~~~

~ [SWS_Mfl_00192] Mfl_Debounce_u8_u8: Include constant for pointer Input-parameter as like below.
boolean Mfl_Debounce_u8_u8(boolean X, Mfl_DebounceState_Type* State, const Mfl_DebounceParam_Type* Param, float32 dT)

~ [SWS_Mfl_00266] Mfl_DebounceInit: The parameter Mfl_DebounceState_Type* State should be Out instead of In parameter as like below.
Parameters (in): X Initial value for the input state
Parameters (out): State Pointer to structure for debouncing state variables

~ [SWS_Mfl_00246] Mfl_HystDeltaRight_f32_u8: Include constant for pointer Input-parameter as like below.
boolean Mfl_HystDeltaRight_f32_u8(float32 X, float32 Delta, float32 Rsp, const uint8* State)

~ [SWS_Mfl_00285] Mfl_MedianSort_f32_f32: The parameter Array should be InOut instead of In parameter as like below.
Parameters (in): N Size of an array
Parameters (inout): Array Pointer to an array

~ [SWS_Mfl_00037] Mfl_PT1SetState: The parameter State_cpst should be Out instead of In parameter as like below.
Parameters (in): X1_f32 Initial value for input state
Y1_f32 Initial value for output state
Parameters (out): State_cpst Pointer to internal state structure

~ [SWS_Mfl_00225] Mfl_RampCheckActivity: Include constant for pointer Input-parameter as like below.
boolean Mfl_RampCheckActivity(const Mfl_StateRamp_Type* State_cpst)

~ [SWS_Mfl_00223] Mfl_RampGetSwitchPos: Include constant for pointer Input-parameter as like below.
boolean Mfl_RampGetSwitchPos(const Mfl_StateRamp_Type* State_cpst)
–Last change on issue 68035 comment 135–
~~~~~~~~~~~~~~~~~~~~~

BW-C-Level:

Application	Specification	Bus
1	4	1

1.6 Specification Item SWS_Xfrm_00044

Trace References:

SRS_Xfrm_00002

Content:

Service name:	<Mip>_Inv_<transformerId>Xfrm_Inv_transformerId2	
Syntax:	uint8 <Mip>_Inv_<transformerId>(Rte_Cs_TransactionHandleType* TransactionHandle, const uint8* buffer, uint32 bufferLength, [Std_ReturnType* returnValue,] [<typeparamtype> * data_1,] ... [<typeparamtype> * data_n])	
Service ID[hex]:	0x04	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	bufferXfrm_Inv_transformerId2.buffer	Buffer allocated by the RTE, where the still transformed data are stored by the Rte
	bufferLengthXfrm_Inv_transformer Id2.bufferLength	Used length of the buffer
Parameters (inout):	None	
Parameters (out):	TransactionHandleXfrm_Inv_transformer Id2.TransactionHandle	Transaction handle according to [SWS_Rte_08732] (clientId and sequenceCounter) needed to differentiate between multiple requests.
	returnValueXfrm_Inv_transformer Id2.returnValue	Return value of the server runnable which needs to be transformed on server side for transmission to the calling client. This argument is only available for deserializers of the response of a Client/Server communication and if the ClientServerOperation has at least one PossibleError defined.
	data_1Xfrm_Inv_transformerId2.data_1	Client/Server operation argument which shall be transformed (in the same order as in the corresponding interface)
...Xfrm_Inv_transformerId2....	...	
data_nXfrm_Inv_transformerId2.data_n	Client/Server operation argument which shall be transformed (in the same order as in the corresponding interface)	
Return value:	uint8	0x00 (E_OK): Transformation successful 0x01 - 0xff: Specific errors

Description:	This function is the interface of the first transformer in a transformer chain of Client/Server communication (this is the last executed transformer on the receiving side!). It takes the constant buffer (IN parameter buffer) of length (IN parameter buffer Length which may be smaller than the maximum buffer size used by the RTE for buffer allocation) as input and outputs the operation arguments and optionally the return value (OUT parameters data_1, ..., data_n, and returnValue).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #68035: [diverse] Introduce rules defining which input parameters shall be passed per value and which ones per const reference

  Problem description:

  SWS_BSW_00186 especially states that input pointer parameters shall use the const qualifier (i.e., shall be P2CONST).

  In addition to that there shall be a SWS item that states that input parameters of integral and enum type shall be passed by value whereas input parameters of structure type shall be passed by reference.

  The various transformer SWS documents shall be adapted accordingly.
  –Last change on issue 68035 comment 4–

  Agreed solution:

  BSW UML model
  ————-
  The attachment "Changed Proposal in WP-A meeting" contains a list of changes to the APIs in the model (see column H). Afterwards all related documents (included in impact list) shall update their
  generated artifacts.

  General Requirements on Basic Software Modules
  ~~~~~~

  Introduce the following requirements prior to SRS_BSW_00371:

  SRS_BSW_xxxxx: Input parameters of scalar and enum types shall be passed as a value.
  Type: valid
  Description: All input parameters of scalar or enum type shall be passed as a value.
  Rationale:
  ~~~~~~

Use case: For example a function named <Mip>_SomeFunction with a return type of Std_ReturnType and a single parameter named SomeParameter of type uint8 is defined with the following signature:

Std_ReturnType <Mip>_SomeFunction(uint8 SomeParameter);
Dependencies: –
Supporting Material: —

SRS_BSW_yyyyy: Input parameters of structure type shall be passed as a reference to a constant structure
Type: valid
Description: All input parameters of structure type shall be passed as a reference constant structure
Rationale: Passing input parameters of structure type by value would result in additional run-time overhead due to efforts for copying the whole structure.
Use case: For example a function named <Mip>_SomeFunction with a return type of Std_ReturnType and a single parameter named SomeParameter of type SomeStructure (where SomeStructure is a struct) is defined with the following signature:

Std_ReturnType <Mip>_SomeFunction(P2CONST(SomeStructure, AUTOMATIC, <MIP>_APPL_DATA) SomeParameter);
Dependencies: –
Supporting Material: —

SRS_BSW_zzzzz: Input parameters of array type shall be passed as a reference to the constant array base type
Type: valid
Description: All input parameters of array type shall be passed as a reference to the constant array base type
Rationale: This effectively matches the behavior specified in the ISO-C:90 namely that a "declaration of a parameter as 'array of type' shall be adjusted to 'qualified pointer to
type'".
Use case: For example a function named <Mip>_SomeFunction with a return type of Std_ReturnType and a single parameter named SomeParameter of type array of uint8 is defined with the following signature:

Std_ReturnType <Mip>_SomeFunction(P2CONST(uint8, AUTOMATIC, <MIP>_APPL_DATA) SomeParameter);

Dependencies: –
Supporting Material: —

General Specification of Transformers
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

In SWS_Xfrm_00036 change

const <type>* dataElement

to

<paramtype> dataElement

and add the following to the where clause after the API table after the bullet "type is data type of the data element
"

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).

In SWS_Xfrm_00038 change

[<type> data_1,] ...
[<type> data_n]

to

[<paramtype> data_1,] ...
[<paramtype> data_n]

and add the following to the where clause after the API table after the bullet "type is data type of the data element
"

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

The following paragraph shall then be removed:

For the arguments of ClientServerOperation which are handed over to the transformer as data_1, ..., data_n the requirements to API parameters stated in chapter API Parameters of [5, SWS RTE] are valid (especially [SWS_Rte_01017], [SWS_Rte_01018] and [SWS_Rte_05107]).

In SWS_Xfrm_00040 change

[<originalData1>, ...
<originalDataN>]

to

[<paramtype> originalData1,] ...
[<paramtype> originalDataN]

and add the following to the where clause after the API table after the bullet "type is data type of the data element
"

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).

In SWS_Xfrm_00044 change

<type> *data_1, ...
<type> *data_n

to

[<paramtype> data_1,] ...
[<paramtype> data_n]

and add the following to the where clause after the API table after the bullet "type is data type of the data element
"

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).

The following paragraph shall then be removed:

For the arguments of ClientServerOperation which are handed over to the transformer as data_1, ..., data_n the requirements to API parameters stated in chapter API Parameters of [5, SWS RTE] are valid (especially [SWS_Rte_01017], [SWS_Rte_01018] and [SWS_Rte_05107]).

Speci?cation of SOME/IP Transformer
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

In SWS_SomeIpXf_00138 change

const <type>* dataElement

to

<paramtype> dataElement

and add the following to the where clause after the API table after the bullet "type is data type of the data element
"

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).

In SWS_SomeIpXf_00141 change

[<type> data_1,] ...
[<type> data_n]

to
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

[<paramtype> data_1,] ...
[<paramtype> data_n]

and add the following to the where clause after the API table after the bullet "type is data type of the data element
"

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).

The following paragraph shall then be removed:

For the arguments of ClientServerOperation which are handed over to the transformer as data_1, ..., data_n the requirements to API parameters stated in chapter API Parameters of [5, SWS RTE] are valid (especially [SWS_Rte_01017], [SWS_Rte_01018] and [SWS_Rte_05107]).

In SWS_SomeIpXf_00145 change

<type> *data_1, ...
<type> *data_n

to

[<paramtype> data_1,] ...
[<paramtype> data_n]

and add the following to the where clause after the API table after the bullet "type is data type of the data element
"

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).

The following paragraph shall then be removed:

For the arguments of ClientServerOperation which are handed over to the transformer as data_1, ..., data_n the requirements to API parameters stated in chapter API Parameters of [5, SWS RTE] are valid (especially [SWS_Rte_01017], [SWS_Rte_01018] and [SWS_Rte_05107]).

Specification of COM Based Transformer
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

In SWS_ComXf_00007 change

const <type>* dataElement

to

<paramtype> dataElement

and add the following to the where clause after the API table after the bullet "type is data type of the data element
"

<paramtype> is derived from <type> according to the parameter passing rules rules defined by the SRS BSW General (see SRS_BSW_xxxxx, SRS_BSW_yyyyy, and SRS_BSW_zzzzz) and SWS BSW General (see SWS_BSW_00186 and SWS_BSW_00187).

Specification of Time Sync over Ethernet
~~~~~~

In SWS_EthTSyn_00040 make the parameter DataPtr of EthTSyn_RxIndication const.

Specification of SWS FlexRay Interface
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Change SWS_FrIf_05073 from
FrIf_NumOfStartupFramesPtr (IN)
to
FrIf_NumOfStartupFramesPtr (OUT)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Specification of ADC
~~~~~~
~[SWS_Adc_00419] Adc_SetupResultBuffer: change Adc_ValueGroupType* to const Adc_ValueGroupType*
~[SWS_Adc_00369] Adc_ReadGroup: move Adc_ValueGroupType * from Parameters (in) to Parameters (out)

There is no need to change parameter from IN to INOUT in Adc_SetupResultBuffer

Specification of Com
~~~~~~
Change type of parameter MetaData of Com_TriggerIPDUSendWithMetaData from uint8* to const uint8*

Specification of ComM
~~~~~~
no change required

Specification of Dem
~~~~~~
no change required

Specification of DLT
~~~~~~
no change required

Specification of DoIP
~~~~~~
From:
Std_ReturnType <User>_DoIPRoutingActivationConfirmation(boolean* Confirmed, uint8* ConfirmationReqData, uint8* ConfirmationResData)
Std_ReturnType <User>_DoIPRoutingActivationAuthentication(boolean* Authentified, uint8* AuthenticationReqData, uint8* AuthenticationResData)

To:
Std_ReturnType <User>_DoIPRoutingActivationConfirmation(boolean* Confirmed,

const uint8* ConfirmationReqData, uint8* ConfirmationResData)
Std_ReturnType <User>_DoIPRoutingActivationAuthentication(boolean* Authenticified, const uint8* AuthenticationReqData, uint8* AuthenticationResData)

Specification of E2ELibrary
~~~~~~
no change required

Specification of Eth
~~~~~~
no change required

Specification of EthIf
~~~~~~
no change required

Specification of EthSwitchDriver
~~~~~~
no change required

Specification of ICUDriver
~~~~~~
SWS_Icu_00201: Icu_StartTimestamp
Parameter (IN): Icu_ValueType* BufferPtr shall be changed to Parameters (out) type

Specification of LdCom
~~~~~~
[SWS_LDCOM_00027]: LdCom_CopyTxData
BufReq_ReturnType LdCom_CopyTxData(PduIdType id, const PduInfoType* info, RetryInfoType* retry, PduLengthType* availableDataPtr) shall be changed to
BufReq_ReturnType LdCom_CopyTxData(PduIdType id, const PduInfoType* info, const RetryInfoType* retry, PduLengthType* availableDataPtr)

[SWS_LDCOM_00036]: Rte_LdComCbkCopyTxData_<sn>
BufReq_ReturnType Rte_LdComCbkCopyTxData_<sn>(const PduInfoType* info, RetryInfoType* retry, PduLengthType* availableDataPtr) shall be changed to
BufReq_ReturnType Rte_LdComCbkCopyTxData_<sn>(const PduInfoType* info,

const RetryInfoType* retry, PduLengthType* availableDataPtr)

Specification of Lin
~~~~~~
PduInfoPtr needs to be const in Std_ReturnType Lin_SendFrame(uint8 Channel, const Lin_PduType* PduInfoPtr)

Specification of PduR
~~~~~~
* PduR_<User:LoTp>CopyTxData
add const to "RetryInfoType* retry"

Specification of J1939Nm
~~~~~~
Change parameter ’name’ of User_AddressClaimedIndication to type ’const uint8*’

Specification of SoAd
~~~~~~
=> everything already fixed with RfC 65633

Specification of SPIHandlerDriver
~~~~~~
==> nothing to change for SWS SPI

Specification of SynchronizedTimeBaseManager
~~~~~~
"StbM not affected. All issues listed in the WP-A attachment have been already implemented by IT 69124 in context of RfC 65633"

Specification of TcpIp
~~~~~~
~[SWS_TCPIP_00040] TcpIp_DhcpReadOption: change DataPtr from (IN) to (OUT)
~[SWS_TCPIP_00189] TcpIp_DhcpV6ReadOption: change DataPtr from (IN) to (OUT)
=> everything else already fixed with RfC 65633
~~~~~~

Specification of TimeSyncOverFlexRay
~~~~~~
"Change SWS_FrTSyn_00064: parameter versioninfo of type Std_VersionInfoType* is marked wrongly as IN. Change to OUT"

Specification of EFX
~~~~~~~~~~~~~~~~~~~~~~
~ [SWS_Efx_00355] Efx_Debounce_u8_u8: Include constant for pointer Input-parameter as like below.
uint8 Efx_Debounce_u8_u8(boolean X, Efx_DebounceState_Type * State, const Efx_DebounceParam_Type * Param, sint32 dT)

~ [SWS_Efx_00376] Efx_MedianSort: The parameter <InType>* Array should be InOut instead of In parameter as like below.
Parameters (in): N Size of an array
Parameters (inout): Array Pointer to an array

~ [SWS_Efx_00309] Efx_RampCheckActivity: Include constant for pointer Input-parameter as like below.
boolean Efx_RampCheckActivity(const Efx_StateRamp_Type* State_cpst)

~ [SWS_Efx_00307] Efx_RampGetSwitchPos: Include constant for pointer Input-parameter as like below.
boolean Efx_RampGetSwitchPos(const Efx_StateRamp_Type* State_cpst)

~ [SWS_Efx_00193] Efx_Array_Average: Include constant for pointer Input-parameter as like below.
<OutType> Efx_Array_Average_<InTypeMn>_<OutTypeMn>(const <InType>* Array, uint16 Count)

Specification of MFL
~~~~~~~~~~~~~~~~~~~~~~
~ [SWS_Mfl_00192] Mfl_Debounce_u8_u8: Include constant for pointer Input-parameter as like below.
boolean Mfl_Debounce_u8_u8(boolean X, Mfl_DebounceState_Type* State, const Mfl_DebounceParam_Type* Param, float32 dT)

~ [SWS_Mfl_00266] Mfl_DebounceInit: The parameter Mfl_DebounceState_Type* State should be Out instead of In parameter as like below.
~~~~~~~~~~~~~~~~~~~~~~

Parameters (in): X Initial value for the input state
Parameters (out): State Pointer to structure for debouncing state variables

~ [SWS_Mfl_00246] Mfl_HystDeltaRight_f32_u8: Include constant for pointer Input-parameter as like below.
boolean Mfl_HystDeltaRight_f32_u8(float32 X, float32 Delta, float32 Rsp, const uint8* State)

~ [SWS_Mfl_00285] Mfl_MedianSort_f32_f32: The parameter Array should be InOut instead of In parameter as like below.
Parameters (in): N Size of an array
Parameters (inout): Array Pointer to an array

~ [SWS_Mfl_00037] Mfl_PT1SetState: The parameter State_cpst should be Out instead of In parameter as like below.
Parameters (in): X1_f32 Initial value for input state
Y1_f32 Initial value for output state
Parameters (out): State_cpst Pointer to internal state structure

~ [SWS_Mfl_00225] Mfl_RampCheckActivity: Include constant for pointer Input-parameter as like below.
boolean Mfl_RampCheckActivity(const Mfl_StateRamp_Type* State_cpst)

~ [SWS_Mfl_00223] Mfl_RampGetSwitchPos: Include constant for pointer Input-parameter as like below.
boolean Mfl_RampGetSwitchPos(const Mfl_StateRamp_Type* State_cpst)
–Last change on issue 68035 comment 135–

BW-C-Level:

Application	Specification	Bus
1	4	1