Document Title	SWS_EthernetInterface: 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_EthernetInterface

1.1 Specification Item ECUC_EthIf_00001

Trace References:

none

Content:

Container Name	EthIfGeneralEthIfGeneral
Description	This container contains the general configuration parameters of the Ethernet Interface.
Configuration Parameters	

Included parameters:

Included Parameters	
Parameter Name	SWS Item ID
EthIfDevErrorDetect	ECUC_EthIf_00004
EthIfEnableRxInterrupt	ECUC_EthIf_00005
EthIfEnableSignalQualityApi	ECUC_EthIf_00076
EthIfEnableTxInterrupt	ECUC_EthIf_00006
EthIfEnableWEthApi	ECUC_EthIf_00075
EthIfGetAndResetMeasurementDataApi	ECUC_EthIf_00072
EthIfGetBaudRate	ECUC_EthIf_00034
EthIfGetCounterState	ECUC_EthIf_00035
EthIfGetCtrlIdxList	ECUC_EthIf_00070
EthIfGetTransceiverWakeupModeApi	ECUC_EthIf_00041
EthIfGetVlanIdSupport	ECUC_EthIf_00071
EthIfGlobalTimeSupport	ECUC_EthIf_00039
EthIfMainFunctionPeriod	ECUC_EthIf_00023
EthIfMainFunctionStatePeriod	ECUC_EthIf_00056
EthIfMaxTrcvsTotal	ECUC_EthIf_00003
EthIfPortStartupActiveTime	ECUC_EthIf_00055
EthIfPublicCddHeaderFile	ECUC_EthIf_00024
EthIfRxIndicationIterations	ECUC_EthIf_00030
EthIfSetForwardingModeApi	ECUC_EthIf_00062
EthIfSignalQualityCheckPeriod	ECUC_EthIf_00077
EthIfStartAutoNegotiation	ECUC_EthIf_00033
EthIfSwitchManagementSupport	ECUC_EthIf_00064
EthIfSwitchOffPortTimeDelay	ECUC_EthIf_00054
EthIfTrcvLinkStateChgMainReload	ECUC_EthIf_00009
EthIfVerifyConfigApi	ECUC_EthIf_00063

Included Parameters	
Parameter Name	SWS Item ID
EthIfVersionInfoApi	ECUC_EthIf_00007
EthIfVersionInfoApiMacro	ECUC_EthIf_00008
EthIfWakeUpSupport	ECUC_EthIf_00040

Included containers:

No Included Containers

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77676: [EthIf][EthSwt][EthTrcv] Complete handling to determine signal quality

  Problem description:

  AR4.3.0 introduce APIs EthSwt_GetPortSignalQuality and EthTrcv_GetPhySignalQuality. This API's return the result of the signal quality meassured by a dedicated Ethernet transceiver. But there is a lack of how the meassurement for the signal quality shall be triggert respectively handled.

  Agreed solution:

  === EthTrcv ===
  ~[SWS_EthTrcv_91001] EthTrcv_GetPhySignalQuality
  ~ uint8* SignalQualityPtr change to uint32* SignalQualityPtr
  ~ change the description of parameter SignalQualityPtr to "Pointer to the memory where the signal quality shall be stored."

  === EthSwt ===
  ~[SWS_EthSwt_91014] EthSwt_GetPortSignalQuality
  ~ uint8* SignalQualityPtr change to uint32* SignalQualityPtr
  ~ change the description of parameter SignalQualityPtr to "Pointer to the memory where the signal quality shall be stored."
  ~ change the description: "...If no transceiver is referenced the signal quality shall be set to 0xFFFFFFFF."
  ~[SWS_EthSwt_00293] change "0xFF" to "0xFFFFFFFF"

  === EthIf ===
  ~ch.8.2
  +[EthIf_xxxxx1] EthIf_SignalQualityResultType
  Type Structure
  Element uint32 HighestSignalQuality the highest signal quality of a link since last

clear
Element uint32 LowestSignalQuality the lowest link signal quality of a link since last clear
Element uint32 ActualSignalQuality the actual signal quality

~ch. 8.3
~ [SWS_EthIf_91019] set EthIf_GetPhySignalQuality to "deprecated"

+[EthIf_xxxxx2] EthIf_GetTrcvSignalQuality
Syntax: Std_ReturnType EthIf_GetTrcvSignalQuality(
uint8 TrcvIdx,
EthIf_SignalQualityResultType* ResultPtr)
Sync/Async: Synchronous
Reentrancy: Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.
Parameters (in): TrcvIdx Index of the transceiver within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): ResultPtr Pointer to the memory where the signal quality in percent shall be stored.
Return value: Std_ReturnType
E_OK: The signal quality retrieved successfully
E_NOT_OK: The signal quality not retrieved successfully
Description: Retrieves the signal quality of the link of the given Ethernet transceiver

+[EthIf_xxxxx3] EthIf_GetSwitchPortSignalQuality
Syntax: Std_ReturnType EthIf_GetSwitchPortSignalQuality(
uint8 SwitchIdx,
uint8 SwitchPortIdx,
EthIf_SignalQualityResultType* ResultPtr)
Sync/Async: Synchronous
Reentrancy: Reentrant for different Ethernet switch indexes and Ethernet Switch port indexes. Non reentrant for the same SwitchPortIdx.
Parameters (in): SwitchIdx Index of the Ethernet switch within the context of the Ethernet Interface
SwitchPortIdx Index of the Ethernet switch port within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): ResultPtr Pointer to the memory where the signal quality in percent shall be stored.
Return value: Std_ReturnType
E_OK: The signal quality retrieved successfully
E_NOT_OK: The signal quality not retrieved successfully
Description: Retrieves the signal quality of the link of the given Ethernet switch port

+[EthIf_xxxxx4] EthIf_ClearTrcvSignalQuality
Syntax: Std_ReturnType EthIf_ClearTrcvSignalQuality(
uint8 TrcvIdx)
Sync/Async: Synchronous
Reentrancy: Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.
Parameters (in): TrcvIdx Index of the transceiver within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): None
Return value: Std_ReturnType
E_OK: The signal quality cleared successfully
E_NOT_OK: The signal quality cleared not successfully
Description: Clear the stored signal quality of the link of the given Ethernet transceiver

+[EthIf_xxxxx5] EthIf_ClearSwitchPortSignalQuality
Syntax: Std_ReturnType EthIf_ClearSwitchPortSignalQuality(
uint8 SwitchIdx,
uint8 SwitchPortIdx)
Sync/Async: Synchronous
Reentrancy: Reentrant for different Ethernet switch indexes and Ethernet Switch port indexes. Non reentrant for the same SwitchPortIdx.
Parameters (in): SwitchIdx Index of the Ethernet switch within the context of the Ethernet Interface
SwitchPortIdx Index of the Ethernet switch port within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): None
Return value: Std_ReturnType
E_OK: The signal quality cleared successfully
E_NOT_OK: The signal quality cleared not successfully
Description: Clear the stored signal quality of the link of the given Ethernet switch port

~ch. 8.5 Scheduled functions
~ SWS_EthIf_91104 EthIf_MainFunctionState
~Description: The function is polling different communication hardware (Ethernet transceiver, Ethernet switch ports) related information, e.g. link state, signal quality.

+[EthIf_xxxx12] The EthIf_MainFunctionState shall poll Ethernet communication hardware related information with the period of EthIfMainFunctionStatePeriod.

+[EthIf_xxxxx6] For each Ethernet switch port where a link state of ETHSWT_LINK_STATE_ACTIVE is yielded and references an Ethernet Transceiver the function shall poll the signal quality by calling EthSwt_GetPortSignalQuality().

+[EthIf_xxxxx7] For each Ethernet transceiver where a link state of ETHTRCV_LINK_STATE_ACTIVE is yielded the function shall poll the signal quality by calling EthTrcv_GetPhySignalQuality().

+[EthIf_xxxxx9] The obtained signal quality value shall be stored as type of EthIf_SignalQualityResultType. The value shall always be stored as ActualSignalQuality. If the obtained signal quality is higher than the stored highest signal quality (HighestSignalQuality), then HighestSignalQuality shall be updated with the obtained signal quality. If the obtained signal quality is lower than the lowest signal quality (LowestSignalQuality), then LowestSignalQuality shall be updated with the obtained signal quality.

~ch. 10 Configuration specification
add the following parameters to section 10.1.2 EthIfGeneral
+ [ECUC_EthIf_xxxxx1] EthIfEnableSignalQualityApi
+ Description: enable/disable the APIs read and clear the signal quality
+ Multiplicity : 1
+ Type : EcucBooleanParamDef
+ Scope / Dependency : scope: local

+ [ECUC_EthIf_xxxxx2] EthIfSignalQualityCheckPeriod
+ Description : Specifies the period in units of seconds in which the signal quality it polled in the context of EthIf_MainfunctionState. The value shall be an integral multiple of EthIfMainFunctionStatePeriod.
+ Multiplicity : 0..1
+ Type : EcucFloatParamDef
+ Scope / Dependency : scope: local
dependency: If this parameter is defined, the EthIf_MainFunctionState shall be generated and parameter EthIfEnableSignalQualityApi shall be set to TRUE.
–Last change on issue 77676 comment 38–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.2 Specification Item ECUC_EthIf_00032

Trace References:

none

Content:

Name	EthIfCtrlMtuEthIfController.EthIfCtrlMtu		
Description	Specifies the maximum transmission unit (MTU) of the EthIfCtrl in [bytes]. Note: in In case a VLAN tag is used for the EthIfCtrl, the MTU is frame length of the Ethernet frame will increase by 4 bytes smaller than the maximum payload size of an Ethernet frame which can be transmitted on the networkbytes.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	64 .. 9000		
Default value	–		
Post-Build Variant Value	true		
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU dependency: EthIfVlanId		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77375: [EthIf] The note regarding MTU is misleading

  Problem description:

  The note in ECUC_EthIf_00032 is misleading:
  "
  Note: in case a VLAN tag is used for the EthIfCtrl, the MTU is 4 bytes smaller than the maximum payload size of an Ethernet frame which can be transmitted on the network.
  "
  MTU (payload) is independ of the VLAN.

  Agreed solution:

  Change note in ECUC_EthIf_00032
  from

  Note: in case a VLAN tag is used for the EthIfCtrl, the MTU is 4 bytes smaller than the maximum payload size of an Ethernet frame which can be transmitted on the network.

to

Note: In case a VLAN tag is used for the EthIfCtrl, the frame length of the Ethernet frame will increase by 4 bytes.
–Last change on issue 77375 comment 1–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.3 Specification Item ECUC_EthIf_00054

Trace References:

none

Content:

Name	EthIfSwitchOffPortTimeDelayEthIfGeneral.EthIfSwitchOffPortTimeDelay		
Parent Container	EthIfGeneral		
Description	Denote the time delay after the mode "ETHTRCV_MODE_DOWN" of a EthIfSwitchPortGroup will be executed. This is only used for EthIfSwtPortGroups which are not referenced by a any EthIfControlleror the reference is of type "link-information". The time delay shall be greater than the UdpNm timings, because UdpNm shall finish its shutdown handling. (Repeat Message State, Prepare Bus-Sleep state, Bus-Sleep state).		
Multiplicity	0..1		
Type	EcucFloatParamDef		
Range	[0.001 .. 65.535]		
Default value	–		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	–	
Scope / Dependency	scope: local dependency: EthIfSwitchOffPortTimeDelay > (UdpNmTimeoutTime + UdpNmWait BusSleepTime)		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77795: [EthIf] Clarify the use of EthIfSwitchPortGroups with reference semantic "link-info"

  Problem description:

  Some requirements regarding the explicit switching of EthIfSwitchPortGroups refer to EthIfSwtPortGroups with reference semantic "link-info". This is not correct since EthIfSwtPortGroups with reference semantic "link-info" are never switched.

  Agreed solution:

  === EthIf ===
  ~ch. 7.1.1.1 Link state accumulation of EthIfSwitchPortGroup
  "... The actual state is reported to the EthSM state machine and additionally if the reference is type of "link-information" to the BswM by calling BswM_EthIf_PortGroupLinkStateChg...." change to
  "
  ... The actual state of EthIfSwtPortGroups referenced by a EthIfController is reported to the EthSM by calling EthSM_TrcvLinkStateChg. The actual state of EthIfSwtPortGroups which are not referenced by any EthIfController is reported to the BswM by calling BswM_EthIf_PortGroupLinkStateChg....
  "

  ~ECUC_EthIf_00054
  Description
  "...This is only used for EthIfSwtPortGroups which not referenced by a EthIfController or the reference is of type "link-information"...." change to
  "...This is only used for EthIfSwtPortGroups which are not referenced by any EthIfController..."

  ~ECUC_EthIf_00055
  Description
  "...This is only used for ports in EthIfSwtPortGroups which are not referenced by a EthIfController or the reference is of type "link-information"." change to
  "...This is only used for ports in EthIfSwtPortGroups which are not referenced by any EthIfController."
  –Last change on issue 77795 comment 1–

  BW-C-Level:

Application	Specification	Bus
1	1	1

1.4 Specification Item ECUC_EthIf_00055

Trace References:

none

Content:

Name	EthIfPortStartupActiveTimeEthIfGeneral.EthIfPortStartupActiveTime		
Parent Container	EthIfGeneral		
Description	Denote the time delay after the mode "ETHTRCV_MODE_ACTIVE" of all EthIfSwitchPorts are requested via EthIf_StartAllPorts. This is only used for ports in EthIfSwtPortGroups which are not referenced by a any EthIf Controlleror the reference is of type "link-information".		
Multiplicity	0..1		
Type	EcucFloatParamDef		
Range	[0.001 .. 65.535]		
Default value	–		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	–	
Scope / Dependency	scope: local		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77795: [EthIf] Clarify the use of EthIfSwitchPortGroups with reference semantic "link-info"

  Problem description:

  Some requirements regarding the explicit switching of EthIfSwitchPortGroups refer to EthIfSwtPortGroups with reference semantic "link-info". This is not correct since EthIfSwtPortGroups with reference semantic "link-info" are never switched.

  Agreed solution:

  === EthIf ===
  ~ch. 7.1.1.1 Link state accumulation of EthIfSwitchPortGroup
  "... The actual state is reported to the EthSM state machine and additionally if the reference is type of "link-information" to the BswM by calling BswM_EthIf_PortGroupLinkStateChg...." change to

"
... The actual state of EthIfSwtPortGroups referenced by a EthIfController is reported to the EthSM by calling EthSM_TrcvLinkStateChg. The actual state of EthIfSwtPortGroups which are not referenced by any EthIfController is reported to the BswM by calling BswM_EthIf_PortGroupLinkStateChg....
"

~ECUC_EthIf_00054
Description
"...This is only used for EthIfSwtPortGroups which not referenced by a EthIfController or the reference is of type "link-information"...." change to
"...This is only used for EthIfSwtPortGroups which are not referenced by any EthIfController..."

~ECUC_EthIf_00055
Description
"...This is only used for ports in EthIfSwtPortGroups which are not referenced by a EthIfController or the reference is of type "link-information"." change to
"...This is only used for ports in EthIfSwtPortGroups which are not referenced by any EthIfController."
–Last change on issue 77795 comment 1–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.5 Specification Item ECUC_EthIf_00076

Trace References:

none

Content:

Name	EthIfEnableSignalQualityApiEthIfGeneral.EthIfEnableSignalQualityApi
Parent Container	EthIfGeneral
Description	Enable/disable the APIs read and clear the signal quality.
Multiplicity	1
Type	EcucBooleanParamDef
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 #77676: [EthIf][EthSwt][EthTrcv] Complete handling to determine signal quality

  Problem description:

  AR4.3.0 introduce APIs EthSwt_GetPortSignalQuality and EthTrcv_GetPhySignalQuality. This API's return the result of the signal quality meassured by a dedicated Ethernet transceiver. But there is a lack of how the meassurement for the signal quality shall be triggert respectively handled.

  Agreed solution:

  === EthTrcv ===
  ~[SWS_EthTrcv_91001] EthTrcv_GetPhySignalQuality
  ~ uint8* SignalQualityPtr change to uint32* SignalQualityPtr
  ~ change the description of parameter SignalQualityPtr to "Pointer to the memory where the signal quality shall be stored."

  === EthSwt ===
  ~[SWS_EthSwt_91014] EthSwt_GetPortSignalQuality
  ~ uint8* SignalQualityPtr change to uint32* SignalQualityPtr
  ~ change the description of parameter SignalQualityPtr to "Pointer to the memory where the signal quality shall be stored."
  ~ change the description: "...If no transceiver is referenced the signal quality shall be set to 0xFFFFFFFF."
  ~[SWS_EthSwt_00293] change "0xFF" to "0xFFFFFFFF"

  === EthIf ===
  ~ch.8.2
  +[EthIf_xxxxx1] EthIf_SignalQualityResultType
  Type Structure
  Element uint32 HighestSignalQuality the highest signal quality of a link since last clear
  Element uint32 LowestSignalQuality the lowest link signal quality of a link since last clear

Element uint32 ActualSignalQuality the actual signal quality

~ch. 8.3
~ [SWS_EthIf_91019] set EthIf_GetPhySignalQuality to "deprecated"

+[EthIf_xxxxx2] EthIf_GetTrcvSignalQuality
Syntax: Std_ReturnType EthIf_GetTrcvSignalQuality(
uint8 TrcvIdx,
EthIf_SignalQualityResultType* ResultPtr)
Sync/Async: Synchronous
Reentrancy: Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.
Parameters (in): TrcvIdx Index of the transceiver within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): ResultPtr Pointer to the memory where the signal quality in percent shall be stored.
Return value: Std_ReturnType
E_OK: The signal quality retrieved successfully
E_NOT_OK: The signal quality not retrieved successfully
Description: Retrieves the signal quality of the link of the given Ethernet transceiver

+[EthIf_xxxxx3] EthIf_GetSwitchPortSignalQuality
Syntax: Std_ReturnType EthIf_GetSwitchPortSignalQuality(
uint8 SwitchIdx,
uint8 SwitchPortIdx,
EthIf_SignalQualityResultType* ResultPtr)
Sync/Async: Synchronous
Reentrancy: Reentrant for different Ethernet switch indexes and Ethernet Switch port indexes. Non reentrant for the same SwitchPortIdx.
Parameters (in): SwitchIdx Index of the Ethernet switch within the context of the Ethernet Interface
SwitchPortIdx Index of the Ethernet switch port within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): ResultPtr Pointer to the memory where the signal quality in percent shall be stored.
Return value: Std_ReturnType
E_OK: The signal quality retrieved successfully
E_NOT_OK: The signal quality not retrieved successfully
Description: Retrieves the signal quality of the link of the given Ethernet switch port

+[EthIf_xxxxx4] EthIf_ClearTrcvSignalQuality
Syntax: Std_ReturnType EthIf_ClearTrcvSignalQuality(

uint8 TrcvIdx)
Sync/Async: Synchronous
Reentrancy: Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.
Parameters (in): TrcvIdx Index of the transceiver within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): None
Return value: Std_ReturnType
E_OK: The signal quality cleared successfully
E_NOT_OK: The signal quality cleared not successfully
Description: Clear the stored signal quality of the link of the given Ethernet transceiver

+[EthIf_xxxxx5] EthIf_ClearSwitchPortSignalQuality
Syntax: Std_ReturnType EthIf_ClearSwitchPortSignalQuality(
uint8 SwitchIdx,
uint8 SwitchPortIdx)
Sync/Async: Synchronous
Reentrancy: Reentrant for different Ethernet switch indexes and Ethernet Switch port indexes. Non reentrant for the same SwitchPortIdx.
Parameters (in): SwitchIdx Index of the Ethernet switch within the context of the Ethernet Interface
SwitchPortIdx Index of the Ethernet switch port within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): None
Return value: Std_ReturnType
E_OK: The signal quality cleared successfully
E_NOT_OK: The signal quality cleared not successfully
Description: Clear the stored signal quality of the link of the given Ethernet switch port

~ch. 8.5 Scheduled functions
~ SWS_EthIf_91104 EthIf_MainFunctionState
~Description: The function is polling different communication hardware (Ethernet transceiver, Ethernet switch ports) related information, e.g. link state, signal quality.

+[EthIf_xxxx12] The EthIf_MainFunctionState shall poll Ethernet communication hardware related information with the period of EthIfMainFunctionStatePeriod.

+[EthIf_xxxxx6] For each Ethernet switch port where a link state of ETHSWT_LINK_STATE_ACTIVE is yielded and references an Ethernet Transceiver the function shall poll the signal quality by calling EthSwt_GetPortSignalQuality().

+[EthIf_xxxxx7] For each Ethernet transceiver where a link state of ETHTRCV_LINK_STATE_ACTIVE is yielded the function shall poll the signal quality by calling EthTrcv_GetPhySignalQuality().

+[EthIf_xxxxx9] The obtained signal quality value shall be stored as type of EthIf_SignalQualityResultType. The value shall always be stored as ActualSignalQuality. If the obtained signal quality is higher than the stored highest signal quality (HighestSignalQuality), then HighestSignalQuality shall be updated with the obtained signal quality. If the obtained signal quality is lower than the lowest signal quality (LowestSignalQuality), then LowestSignalQuality shall be updated with the obtained signal quality.

~ch. 10 Configuration specification
add the following parameters to section 10.1.2 EthIfGeneral
+ [ECUC_EthIf_xxxxx1] EthIfEnableSignalQualityApi
+ Description: enable/disable the APIs read and clear the signal quality
+ Multiplicity : 1
+ Type : EcucBooleanParamDef
+ Scope / Dependency : scope: local

+ [ECUC_EthIf_xxxxx2] EthIfSignalQualityCheckPeriod
+ Description : Specifies the period in units of seconds in which the signal quality it polled in the context of EthIf_MainfunctionState. The value shall be an integral multiple of EthIfMainFunctionStatePeriod.
+ Multiplicity : 0..1
+ Type : EcucFloatParamDef
+ Scope / Dependency : scope: local
dependency: If this parameter is defined, the EthIf_MainFunctionState shall be generated and parameter EthIfEnableSignalQualityApi shall be set to TRUE.
–Last change on issue 77676 comment 38–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.6 Specification Item ECUC_EthIf_00077

Trace References:

none

Content:

Name	EthIfSignalQualityCheckPeriodEthIfGeneral.EthIfSignalQualityCheckPeriod		
Parent Container	EthIfGeneral		
Description	Specifies the period in units of seconds in which the signal quality it polled in the context of EthIf_MainfunctionState. The value shall be an integral multiple of EthIfMainFunctionStatePeriod.		
Multiplicity	0..1		
Type	EcucFloatParamDef		
Range	[-INF .. INF]		
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 dependency: If this parameter is defined, the EthIf_MainFunctionState shall be generated and parameter EthIfEnableSignalQualityApi shall be set to TRUE.		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77676: [EthIf][EthSwt][EthTrcv] Complete handling to determine signal quality

  Problem description:

  AR4.3.0 introduce APIs EthSwt_GetPortSignalQuality and EthTrcv_GetPhySignalQuality. This API's return the result of the signal quality meassured by a dedicated Ethernet transceiver. But there is a lack of how the meassurement for the signal quality shall be triggert respectively handled.

  Agreed solution:

  === EthTrcv ===
  ~[SWS_EthTrcv_91001] EthTrcv_GetPhySignalQuality
  ~ uint8* SignalQualityPtr change to uint32* SignalQualityPtr
  ~ change the description of parameter SignalQualityPtr to "Pointer to the memory where the signal quality shall be stored."

  === EthSwt ===
  ~[SWS_EthSwt_91014] EthSwt_GetPortSignalQuality
  ~ uint8* SignalQualityPtr change to uint32* SignalQualityPtr
  ~ change the description of parameter SignalQualityPtr to "Pointer to the memory where the signal quality shall be stored."
  ~ change the description: "...If no transceiver is referenced the signal quality shall

be set to 0xFFFFFFFF."
~[SWS_EthSwt_00293] change "0xFF" to "0xFFFFFFFF"

=== EthIf ===
~ch.8.2
+[EthIf_xxxxx1] EthIf_SignalQualityResultType
Type Structure
Element uint32 HighestSignalQuality the highest signal quality of a link since last clear
Element uint32 LowestSignalQuality the lowest link signal quality of a link since last clear
Element uint32 ActualSignalQuality the actual signal quality

~ch. 8.3
~ [SWS_EthIf_91019] set EthIf_GetPhySignalQuality to "deprecated"

+[EthIf_xxxxx2] EthIf_GetTrcvSignalQuality
Syntax: Std_ReturnType EthIf_GetTrcvSignalQuality(
uint8 TrcvIdx,
EthIf_SignalQualityResultType* ResultPtr)
Sync/Async: Synchronous
Reentrancy: Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.
Parameters (in): TrcvIdx Index of the transceiver within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): ResultPtr Pointer to the memory where the signal quality in percent shall be stored.
Return value: Std_ReturnType
E_OK: The signal quality retrieved successfully
E_NOT_OK: The signal quality not retrieved successfully
Description: Retrieves the signal quality of the link of the given Ethernet transceiver

+[EthIf_xxxxx3] EthIf_GetSwitchPortSignalQuality
Syntax: Std_ReturnType EthIf_GetSwitchPortSignalQuality(
uint8 SwitchIdx,
uint8 SwitchPortIdx,
EthIf_SignalQualityResultType* ResultPtr)
Sync/Async: Synchronous
Reentrancy: Reentrant for different Ethernet switch indexes and Ethernet Switch port indexes. Non reentrant for the same SwitchPortIdx.
Parameters (in): SwitchIdx Index of the Ethernet switch within the context of the Ethernet Interface

SwitchPortIdx Index of the Ethernet switch port within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): ResultPtr Pointer to the memory where the signal quality in percent shall be stored.
Return value: Std_ReturnType
E_OK: The signal quality retrieved successfully
E_NOT_OK: The signal quality not retrieved successfully
Description: Retrieves the signal quality of the link of the given Ethernet switch port

+[EthIf_xxxxx4] EthIf_ClearTrcvSignalQuality
Syntax: Std_ReturnType EthIf_ClearTrcvSignalQuality(
uint8 TrcvIdx)
Sync/Async: Synchronous
Reentrancy: Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.
Parameters (in): TrcvIdx Index of the transceiver within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): None
Return value: Std_ReturnType
E_OK: The signal quality cleared successfully
E_NOT_OK: The signal quality cleared not successfully
Description: Clear the stored signal quality of the link of the given Ethernet transceiver

+[EthIf_xxxxx5] EthIf_ClearSwitchPortSignalQuality
Syntax: Std_ReturnType EthIf_ClearSwitchPortSignalQuality(
uint8 SwitchIdx,
uint8 SwitchPortIdx)
Sync/Async: Synchronous
Reentrancy: Reentrant for different Ethernet switch indexes and Ethernet Switch port indexes. Non reentrant for the same SwitchPortIdx.
Parameters (in): SwitchIdx Index of the Ethernet switch within the context of the Ethernet Interface
SwitchPortIdx Index of the Ethernet switch port within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): None
Return value: Std_ReturnType
E_OK: The signal quality cleared successfully
E_NOT_OK: The signal quality cleared not successfully
Description: Clear the stored signal quality of the link of the given Ethernet switch port

~ch. 8.5 Scheduled functions
~ SWS_EthIf_91104 EthIf_MainFunctionState
~Description: The function is polling different communication hardware (Ethernet transceiver, Ethernet switch ports) related information, e.g. link state, signal quality.

+[EthIf_xxxx12] The EthIf_MainFunctionState shall poll Ethernet communication hardware related information with the period of EthIfMainFunctionStatePeriod.

+[EthIf_xxxxx6] For each Ethernet switch port where a link state of ETHSWT_LINK_STATE_ACTIVE is yielded and references an Ethernet Transceiver the function shall poll the signal quality by calling EthSwt_GetPortSignalQuality().

+[EthIf_xxxxx7] For each Ethernet transceiver where a link state of ETHTRCV_LINK_STATE_ACTIVE is yielded the function shall poll the signal quality by calling EthTrcv_GetPhySignalQuality().

+[EthIf_xxxxx9] The obtained signal quality value shall be stored as type of EthIf_SignalQualityResultType. The value shall always be stored as ActualSignalQuality. If the obtained signal quality is higher than the stored highest signal quality (HighestSignalQuality), then HighestSignalQuality shall be updated with the obtained signal quality. If the obtained signal quality is lower than the lowest signal quality (LowestSignalQuality), then LowestSignalQuality shall be updated with the obtained signal quality.

~ch. 10 Configuration specification
add the following parameters to section 10.1.2 EthIfGeneral
+ [ECUC_EthIf_xxxxx1] EthIfEnableSignalQualityApi
+ Description: enable/disable the APIs read and clear the signal quality
+ Multiplicity : 1
+ Type : EcucBooleanParamDef
+ Scope / Dependency : scope: local

+ [ECUC_EthIf_xxxxx2] EthIfSignalQualityCheckPeriod
+ Description : Specifies the period in units of seconds in which the signal quality it polled in the context of EthIf_MainfunctionState. The value shall be an integral multiple of EthIfMainFunctionStatePeriod.
+ Multiplicity : 0..1
+ Type : EcucFloatParamDef
+ Scope / Dependency : scope: local
dependency: If this parameter is defined, the EthIf_MainFunctionState shall be

generated and parameter EthIfEnableSignalQualityApi shall be set to TRUE.
–Last change on issue 77676 comment 38–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.7 Specification Item SWS_EthIf_00017

Trace References:

none

Content:

Type or error	Relevance	Related error code	Value [hex]
Invalid controller index	Default Development Error	ETHIF_E_INV_CTRL_IDX	0x01
Invalid transceiver index	Default Development Error	ETHIF_E_INV_TRCV_IDX	0x02
Invalid switch index	Development Error	ETHIF_E_INV_SWT_IDX	0x03
Invalid port group index	Default Development Error	ETHIF_E_INV_PORT_GROUP_IDX	0x03 0x04
EthIf module was not initialized	Default Development Error	ETHIF_E_NOT_INITIALIZED UNINIT	0x04 0x05
Invalid pointer in parameter list	Default Development Error	ETHIF_E_PARAM_POINTER	0x05 0x06
Invalid parameter	Default Development Error	ETHIF_E_INV_PARAM	0x06 0x07
Initialization failure	Default Development Error	ETHIF_E_INIT_FAILED	0x07 0x08

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of ’Runtime errors’

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another.

Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
- The review task of the ITs shall be done by the WP to which the specification "belongs".

*** BSW UML Model ***
SWS_CanNm:
———-
Chapter 8.6.1 Optional Interfaces:
Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

SWS_LinIf:
———-
SWS_LinIf_00359: add Det_ReportRuntimeError

SWS_UdpNm:
———-
Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	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

1.8 Specification Item SWS_EthIf_00027

Trace References:

none

Content:

Caveat: The API has to be called during initialization.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #75637: [diverse] Remove requirements like initializiation requirements regarding the <Module>_Init API

  Problem description:

  Generally a Caveat can never start a requirement. For this all basic Software module shall be checked and cleaned up for this formulation.

  original finding was:

  According to the discussion of RfC# 67678 we detect requirements regarding the initializiation of the certain api's, e.g.:
  "
  ...
  [SWS_EthIf_00213] ?
  Caveat: The function requires previous initialization (EthIf_Init). ?()
  "
  Such requirment are not implementation relevant. Please remove this and add as note.
  –Last change on issue 75637 comment 2–

  Agreed solution:

  General:
  - Remove requirements which state that previous initialization of <Modul>_Init() is required, which are formulated like a note, e.g. : [SWS_EthIf_00213] ?
  Caveat: The function requires previous initialization (EthIf_Init). ?()
  - Add general requirement as note to ch. 8.3 Function definitions
  - Remove or reformulate all other requirements, which start with caveat. Most of them are simply a note or implementation hint.

  === EhtIf ===
  ~8.3 Function definitions
  +Note: All functions in this chapter requires previous initialization (EthIf_Init), except the following ones:
  EthIf_Init, EthIf_GetVersionInfo

  ~8.3.2 EthIf_SetControllerMode
  -[SWS_EthIf_00038]

  ~8.3.3 EthIf_GetControllerMode
  -[SWS_EthIf_00044]

~8.3.8 EthIf_ CheckWakeup
-[SWS_EthIf_00249]

~8.3.9 EthIf_GetPhysAddr
-[SWS_EthIf_00066]

~8.3.10 EthIf_SetPhysAddr
-[SWS_EthIf_00138]

~8.3.11 EthIf_UpdatePhysAddrFilter
-[SWS_EthIf_00144]

~8.3.12 EthIf_GetPortMacAddr
-[SWS_EthIf_00195]

~8.3.13 EthIf_GetArlTable
-[SWS_EthIf_00201]

8.3.14 EthIf_GetBufferLevel
-[SWS_EthIf_00207]

~8.3.15 EthIf_GetDropCount
-[SWS_EthIf_00213]

~8.3.16 EthIf_StoreConfiguration
-[SWS_EthIf_00218]

~8.3.17 EthIf_ResetConfiguration
-[SWS_EthIf_00223]

8.3.18 EthIf_GetCurrentTime
-[SWS_EthIf_00159]

~8.3.19 EthIf_EnableEgressTimeStamp
-[SWS_EthIf_00165]

~8.3.20 EthIf_GetEgressTimeStamp
-[SWS_EthIf_00171]

~8.3.21 EthIf_GetIngressTimeStamp
-[SWS_EthIf_00177]

~8.3.22 EthIf_SetCorrectionTime
-[SWS_EthIf_00183]

~8.3.23 EthIf_SetGlobalTime
-[SWS_EthIf_00189]

~8.3.24 EthIf_ProvideTxBuffer
-[SWS_EthIf_00074]

~8.3.25 EthIf_Transmit
-[SWS_EthIf_00081]

=== CanIf ===
General note for below items which contain phrases like "The CAN Interface module must be initialized after Power ON": this is not needed anymore cause already stated in SWS_CANIF_00661. So e.g. in cases where result from CanDrv is just passed through the whole requirement can be deleted.

Also it is assumed, that the mindset like defined in SWS_CANIF_00661 is there for all upper layer functions like <User_TxConfirmation> which might be called by CanIf.

Because if CanDrv would not be already initialised it would return E_NOT_OK which is passed through:
-SWS_CANIF_00312
-SWS_CANIF_00316

Because already guarded by SWS_CANIF_00661:
-SWS_CANIF_00334
-SWS_CANIF_00339
-SWS_CANIF_00344
-SWS_CANIF_00349

~SWS_CANIF_00661
All CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall not execute their normal operation and return E_NOT_OK unless the CanIf has been initialized with a preceding call of CanIf_Init().

+SWS_CANIF_????1
If the switch CANIF_PUBLIC_DEV_ERROR_DETECT is enabled, all CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall report to the DET (using CANIF_E_UNINIT) unless the CanIf has been initialized with a preceding call of CanIf_Init().

-SWS_CANIF_00323 change to
Note: During the call of CanIf_Transmit() the buffer of PduInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00329
CanIf_ReadRxPduData() shall not be used for CanIfRxSduId, which are defined to receive multiple CAN-Ids (range reception).

Add note below SWS_CANIF_00329
Note: During the call of of CanIf_ReadRxPduData() the buffer of CanIfRxInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00356
CanIf_SetDynamicTxId() shall not be interrupted by CanIf_Transmit(), if the same L-SDU ID is handled.

-SWS_CANIF_00407 change to
Note: The call context of CanIf_CheckValidation() is either on interrupt level (interrupt mode) or on task level (polling mode).
Caveat: The corresponding CAN controller and transceiver must be switched on via CanTrcv_SetOpMode(Transceiver, CANTRCV_TRCVMODE_NORMAL) and Can_SetControllerMode(Controller, CAN_CS_STARTED) and the corresponding mode indications must have been called.

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."

-SWS_CANIF_00737 CanIf_GetTxConfirmationState()
-SWS_CANIF_00870 CanIf_SetBaudrate()

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
-SWS_CANIF_00401 CanIf_CheckWakeup()
-SWS_CANIF_00413 CanIf_TxConfirmation()
-SWS_CANIF_00422 CanIf_RxIndication()
-SWS_CANIF_00432 CanIf_ControllerBusOff()
-SWS_CANIF_00818 CanIf_ConfirmPnAvailability()
-SWS_CANIF_00807 CanIf_ClearTrcvWufFlagIndication()
-SWS_CANIF_00811 CanIf_CheckTrcvWakeFlagIndication()
-SWS_CANIF_00703 CanIf_ControllerModeIndication()
-SWS_CANIF_00709 CanIf_TrcvModeIndication()
-SWS_CANIF_00887 <User_TriggerTransmit>()
-SWS_CANIF_00437 <User_TxConfirmation>()

-SWS_CANIF_00440 change to
Note: Until <User_RxIndication>() returns, CanIf will not access PduInfoPtr. The PduInfoPtr is only valid and can be used by upper layers, until the indication returns. CanIf guarantees that the number of configured bytes for this PduInfoPtr is valid.
Note: The call context of <User_RxIndication>() is either on interrupt level (interrupt mode) or on task level (polling mode).

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00822 <User_ConfirmPnAvailability>()
-SWS_CANIF_00793 <User_ClearTrcvWufFlagIndication>()
-SWS_CANIF_00799 <User_CheckTrcvWakeFlagIndication>()

Remove all following requirements and instead add there two Notes with tem-

plate <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00688 <User_ControllerModeIndication>()

=== ComM ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (ComM_Init), except the following ones:
ComM_Init, ComM_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the ComM module is initialized correctly.
-[SWS_ComM_00805] Caveats of ComM_Nm_NetworkStartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00806] Caveats of ComM_Nm_NetworkMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00808] Caveats of ComM_Nm_PrepareBusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00810] Caveats of ComM_Nm_BusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00812] Caveats of ComM_Nm_RestartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00814] Caveats of ComM_EcuM_WakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00965] Caveats of ComM_EcuM_PNCWakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00816] Caveats of ComM_BusSM_ModeIndication(): The Communication Manager Module is initialized correctly.()

=== LdCom ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (LdCom_Init), except the following ones:
LdCom_Init, LdCom_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the LdCom module is initialized correctly.
–Last change on issue 75637 comment 23–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.9 Specification Item SWS_EthIf_00036

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZED UNINIT otherwise (if DET is disabled) return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of ’Runtime errors’

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification "belongs".

  *** BSW UML Model ***
  SWS_CanNm:
  ———-

  Chapter 8.6.1 Optional Interfaces:
  Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

SWS_LinIf:
———-
SWS_LinIf_00359: add Det_ReportRuntimeError

SWS_UdpNm:
———-
Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.10 Specification Item SWS_EthIf_00038

Trace References:

none

Content:

Caveat: The function requires previous initialization (EthIf_Init).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #75637: [diverse] Remove requirements like initializiation requirements regarding the <Module>_Init API

  Problem description:

  Generally a Caveat can never start a requirement. For this all basic Software module shall be checked and cleaned up for this formulation.

  original finding was:

  According to the discussion of RfC# 67678 we detect requirements regarding the initializiation of the certain api’s, e.g.:

"
...
[SWS_EthIf_00213] ?
Caveat: The function requires previous initialization (EthIf_Init). ?()
"
Such requirment are not implementation relevant. Please remove this and add as note.
–Last change on issue 75637 comment 2–

Agreed solution:

General:
- Remove requirements which state that previous initialization of <Modul>_Init() is required, which are formulated like a note, e.g. : [SWS_EthIf_00213] ?
Caveat: The function requires previous initialization (EthIf_Init). ?()
- Add general requirement as note to ch. 8.3 Function definitions
- Remove or reformulate all other requirements, which start with caveat. Most of them are simply a note or implementation hint.

=== EhtIf ===
~8.3 Function definitions
+Note: All functions in this chapter requires previous initialization (EthIf_Init), except the following ones:
EthIf_Init, EthIf_GetVersionInfo

~8.3.2 EthIf_SetControllerMode
-[SWS_EthIf_00038]

~8.3.3 EthIf_GetControllerMode
-[SWS_EthIf_00044]

~8.3.8 EthIf_ CheckWakeup
-[SWS_EthIf_00249]

~8.3.9 EthIf_GetPhysAddr
-[SWS_EthIf_00066]

~8.3.10 EthIf_SetPhysAddr
-[SWS_EthIf_00138]

~8.3.11 EthIf_UpdatePhysAddrFilter
-[SWS_EthIf_00144]

~8.3.12 EthIf_GetPortMacAddr
-[SWS_EthIf_00195]

~8.3.13 EthIf_GetArlTable
-[SWS_EthIf_00201]

8.3.14 EthIf_GetBufferLevel
-[SWS_EthIf_00207]

~8.3.15 EthIf_GetDropCount
-[SWS_EthIf_00213]

~8.3.16 EthIf_StoreConfiguration
-[SWS_EthIf_00218]

~8.3.17 EthIf_ResetConfiguration
-[SWS_EthIf_00223]

8.3.18 EthIf_GetCurrentTime
-[SWS_EthIf_00159]

~8.3.19 EthIf_EnableEgressTimeStamp
-[SWS_EthIf_00165]

~8.3.20 EthIf_GetEgressTimeStamp
-[SWS_EthIf_00171]

~8.3.21 EthIf_GetIngressTimeStamp
-[SWS_EthIf_00177]

~8.3.22 EthIf_SetCorrectionTime
-[SWS_EthIf_00183]

~8.3.23 EthIf_SetGlobalTime
-[SWS_EthIf_00189]

~8.3.24 EthIf_ProvideTxBuffer
-[SWS_EthIf_00074]

~8.3.25 EthIf_Transmit
-[SWS_EthIf_00081]

=== CanIf ===
General note for below items which contain phrases like "The CAN Interface module must be initialized after Power ON": this is not needed anymore cause already stated in SWS_CANIF_00661. So e.g. in cases where result from CanDrv is just passed through the whole requirement can be deleted.

Also it is assumed, that the mindset like defined in SWS_CANIF_00661 is there for all upper layer functions like <User_TxConfirmation> which might be called by CanIf.

Because if CanDrv would not be already initialised it would return E_NOT_OK which is passed through:
-SWS_CANIF_00312
-SWS_CANIF_00316

Because already guarded by SWS_CANIF_00661:
-SWS_CANIF_00334
-SWS_CANIF_00339
-SWS_CANIF_00344
-SWS_CANIF_00349

~SWS_CANIF_00661
All CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall not execute their normal operation and return E_NOT_OK unless the CanIf has been initialized with a preceding call of CanIf_Init().

+SWS_CANIF_????1
If the switch CANIF_PUBLIC_DEV_ERROR_DETECT is enabled, all CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall report to the DET (using CANIF_E_UNINIT) unless the CanIf has been initialized with a preceding call of CanIf_Init().

-SWS_CANIF_00323 change to
Note: During the call of CanIf_Transmit() the buffer of PduInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00329
CanIf_ReadRxPduData() shall not be used for CanIfRxSduId, which are defined to receive multiple CAN-Ids (range reception).

Add note below SWS_CANIF_00329
Note: During the call of of CanIf_ReadRxPduData() the buffer of CanIfRxInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00356
CanIf_SetDynamicTxId() shall not be interrupted by CanIf_Transmit(), if the same L-SDU ID is handled.

-SWS_CANIF_00407 change to
Note: The call context of CanIf_CheckValidation() is either on interrupt level (interrupt mode) or on task level (polling mode).
Caveat: The corresponding CAN controller and transceiver must be switched on via CanTrcv_SetOpMode(Transceiver, CANTRCV_TRCVMODE_NORMAL) and Can_SetControllerMode(Controller, CAN_CS_STARTED) and the corresponding mode indications must have been called.

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
-SWS_CANIF_00737 CanIf_GetTxConfirmationState()
-SWS_CANIF_00870 CanIf_SetBaudrate()

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
-SWS_CANIF_00401 CanIf_CheckWakeup()
-SWS_CANIF_00413 CanIf_TxConfirmation()
-SWS_CANIF_00422 CanIf_RxIndication()
-SWS_CANIF_00432 CanIf_ControllerBusOff()
-SWS_CANIF_00818 CanIf_ConfirmPnAvailability()

-SWS_CANIF_00807 CanIf_ClearTrcvWufFlagIndication()
-SWS_CANIF_00811 CanIf_CheckTrcvWakeFlagIndication()
-SWS_CANIF_00703 CanIf_ControllerModeIndication()
-SWS_CANIF_00709 CanIf_TrcvModeIndication()
-SWS_CANIF_00887 <User_TriggerTransmit>()
-SWS_CANIF_00437 <User_TxConfirmation>()

-SWS_CANIF_00440 change to
Note: Until <User_RxIndication>() returns, CanIf will not access PduInfoPtr. The PduInfoPtr is only valid and can be used by upper layers, until the indication returns. CanIf guarantees that the number of configured bytes for this PduInfoPtr is valid.
Note: The call context of <User_RxIndication>() is either on interrupt level (interrupt mode) or on task level (polling mode).

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00822 <User_ConfirmPnAvailability>()
-SWS_CANIF_00793 <User_ClearTrcvWufFlagIndication>()
-SWS_CANIF_00799 <User_CheckTrcvWakeFlagIndication>()

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00688 <User_ControllerModeIndication>()

=== ComM ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (ComM_Init), except the following ones:
ComM_Init, ComM_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the ComM module is initialized correctly.
-[SWS_ComM_00805] Caveats of ComM_Nm_NetworkStartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00806] Caveats of ComM_Nm_NetworkMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00808] Caveats of ComM_Nm_PrepareBusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00810] Caveats of ComM_Nm_BusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00812] Caveats of ComM_Nm_RestartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00814] Caveats of ComM_EcuM_WakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00965] Caveats of ComM_EcuM_PNCWakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00816] Caveats of ComM_BusSM_ModeIndication(): The Communication Manager Module is initialized correctly.()

=== LdCom ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (LdCom_Init), except the following ones:
LdCom_Init, LdCom_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the LdCom module is initialized correctly.
–Last change on issue 75637 comment 23–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.11 Specification Item SWS_EthIf_00041

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZED UNINIT otherwise (if DET is disabled) return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification "belongs".

  *** BSW UML Model ***
  SWS_CanNm:
  ———-
  Chapter 8.6.1 Optional Interfaces:
  Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

  SWS_LinIf:
  ———-
  SWS_LinIf_00359: add Det_ReportRuntimeError

  SWS_UdpNm:
  ———-
  Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

  *** ECUC XML ***
  Not affected. No configuration of runtime error reporting required (see SWS BSW

General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.12 Specification Item SWS_EthIf_00044

Trace References:

none

Content:

Caveat: The function requires previous initialization (EthIf_Init).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #75637: [diverse] Remove requirements like initializiation requirements regarding the <Module>_Init API

  Problem description:

  Generally a Caveat can never start a requirement. For this all basic Software module shall be checked and cleaned up for this formulation.

  original finding was:

  According to the discussion of RfC# 67678 we detect requirements regarding the initializiation of the certain api's, e.g.:
  "
  ...
  [SWS_EthIf_00213] ?
  Caveat: The function requires previous initialization (EthIf_Init). ?()
  "
  Such requirment are not implementation relevant. Please remove this and add as note.
  –Last change on issue 75637 comment 2–

  Agreed solution:

  General:
  - Remove requirements which state that previous initialization of <Modul>_Init() is required, which are formulated like a note, e.g. : [SWS_EthIf_00213] ?

Caveat: The function requires previous initialization (EthIf_Init). ?()
- Add general requirement as note to ch. 8.3 Function definitions
- Remove or reformulate all other requirements, which start with caveat. Most of them are simply a note or implementation hint.

=== EhtIf ===
~8.3 Function definitions
+Note: All functions in this chapter requires previous initialization (EthIf_Init), except the following ones:
EthIf_Init, EthIf_GetVersionInfo

~8.3.2 EthIf_SetControllerMode
-[SWS_EthIf_00038]

~8.3.3 EthIf_GetControllerMode
-[SWS_EthIf_00044]

~8.3.8 EthIf_ CheckWakeup
-[SWS_EthIf_00249]

~8.3.9 EthIf_GetPhysAddr
-[SWS_EthIf_00066]

~8.3.10 EthIf_SetPhysAddr
-[SWS_EthIf_00138]

~8.3.11 EthIf_UpdatePhysAddrFilter
-[SWS_EthIf_00144]

~8.3.12 EthIf_GetPortMacAddr
-[SWS_EthIf_00195]

~8.3.13 EthIf_GetArlTable
-[SWS_EthIf_00201]

8.3.14 EthIf_GetBufferLevel
-[SWS_EthIf_00207]

~8.3.15 EthIf_GetDropCount
-[SWS_EthIf_00213]

~8.3.16 EthIf_StoreConfiguration

-[SWS_EthIf_00218]

~8.3.17 EthIf_ResetConfiguration
-[SWS_EthIf_00223]

8.3.18 EthIf_GetCurrentTime
-[SWS_EthIf_00159]

~8.3.19 EthIf_EnableEgressTimeStamp
-[SWS_EthIf_00165]

~8.3.20 EthIf_GetEgressTimeStamp
-[SWS_EthIf_00171]

~8.3.21 EthIf_GetIngressTimeStamp
-[SWS_EthIf_00177]

~8.3.22 EthIf_SetCorrectionTime
-[SWS_EthIf_00183]

~8.3.23 EthIf_SetGlobalTime
-[SWS_EthIf_00189]

~8.3.24 EthIf_ProvideTxBuffer
-[SWS_EthIf_00074]

~8.3.25 EthIf_Transmit
-[SWS_EthIf_00081]

=== CanIf ===
General note for below items which contain phrases like "The CAN Interface module must be initialized after Power ON": this is not needed anymore cause already stated in SWS_CANIF_00661. So e.g. in cases where result from CanDrv is just passed through the whole requirement can be deleted.

Also it is assumed, that the mindset like defined in SWS_CANIF_00661 is there for all upper layer functions like <User_TxConfirmation> which might be called by CanIf.

Because if CanDrv would not be already initialised it would return E_NOT_OK

which is passed through:
-SWS_CANIF_00312
-SWS_CANIF_00316

Because already guarded by SWS_CANIF_00661:
-SWS_CANIF_00334
-SWS_CANIF_00339
-SWS_CANIF_00344
-SWS_CANIF_00349

~SWS_CANIF_00661
All CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall not execute their normal operation and return E_NOT_OK unless the CanIf has been initialized with a preceding call of CanIf_Init().

+SWS_CANIF_????1
If the switch CANIF_PUBLIC_DEV_ERROR_DETECT is enabled, all CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall report to the DET (using CANIF_E_UNINIT) unless the CanIf has been initialized with a preceding call of CanIf_Init().

-SWS_CANIF_00323 change to
Note: During the call of CanIf_Transmit() the buffer of PduInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00329
CanIf_ReadRxPduData() shall not be used for CanIfRxSduId, which are defined to receive multiple CAN-Ids (range reception).

Add note below SWS_CANIF_00329
Note: During the call of of CanIf_ReadRxPduData() the buffer of CanIfRxInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00356
CanIf_SetDynamicTxId() shall not be interrupted by CanIf_Transmit(), if the same L-SDU ID is handled.

-SWS_CANIF_00407 change to
Note: The call context of CanIf_CheckValidation() is either on interrupt level (interrupt mode) or on task level (polling mode).
Caveat: The corresponding CAN controller and transceiver must be switched on via CanTrcv_SetOpMode(Transceiver, CANTRCV_TRCVMODE_NORMAL) and Can_SetControllerMode(Controller, CAN_CS_STARTED) and the corresponding mode indications must have been called.

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
-SWS_CANIF_00737 CanIf_GetTxConfirmationState()
-SWS_CANIF_00870 CanIf_SetBaudrate()

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
-SWS_CANIF_00401 CanIf_CheckWakeup()
-SWS_CANIF_00413 CanIf_TxConfirmation()
-SWS_CANIF_00422 CanIf_RxIndication()
-SWS_CANIF_00432 CanIf_ControllerBusOff()
-SWS_CANIF_00818 CanIf_ConfirmPnAvailability()
-SWS_CANIF_00807 CanIf_ClearTrcvWufFlagIndication()
-SWS_CANIF_00811 CanIf_CheckTrcvWakeFlagIndication()
-SWS_CANIF_00703 CanIf_ControllerModeIndication()
-SWS_CANIF_00709 CanIf_TrcvModeIndication()
-SWS_CANIF_00887 <User_TriggerTransmit>()
-SWS_CANIF_00437 <User_TxConfirmation>()

-SWS_CANIF_00440 change to
Note: Until <User_RxIndication>() returns, CanIf will not access PduInfoPtr. The PduInfoPtr is only valid and can be used by upper layers, until the indication returns. CanIf guarantees that the number of configured bytes for this PduInfoPtr is valid.
Note: The call context of <User_RxIndication>() is either on interrupt level (interrupt

mode) or on task level (polling mode).

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00822 <User_ConfirmPnAvailability>()
-SWS_CANIF_00793 <User_ClearTrcvWufFlagIndication>()
-SWS_CANIF_00799 <User_CheckTrcvWakeFlagIndication>()

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00688 <User_ControllerModeIndication>()

=== ComM ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (ComM_Init), except the following ones:
ComM_Init, ComM_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the ComM module is initialized correctly.
-[SWS_ComM_00805] Caveats of ComM_Nm_NetworkStartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00806] Caveats of ComM_Nm_NetworkMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00808] Caveats of ComM_Nm_PrepareBusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00810] Caveats of ComM_Nm_BusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00812] Caveats of ComM_Nm_RestartIndication: The ComM module is initialized correctly.()

-[SWS_ComM_00814] Caveats of ComM_EcuM_WakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00965] Caveats of ComM_EcuM_PNCWakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00816] Caveats of ComM_BusSM_ModeIndication(): The Communication Manager Module is initialized correctly.()

=== LdCom ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (LdCom_Init), except the following ones:
LdCom_Init, LdCom_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the LdCom module is initialized correctly.
–Last change on issue 75637 comment 23–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.13 Specification Item SWS_EthIf_00063

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZEDUNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

Notes:
- It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
- The review task of the ITs shall be done by the WP to which the specification "belongs".

*** BSW UML Model ***
SWS_CanNm:
———-
Chapter 8.6.1 Optional Interfaces:
Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

SWS_LinIf:
———-
SWS_LinIf_00359: add Det_ReportRuntimeError

SWS_UdpNm:
———-
Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.14 Specification Item SWS_EthIf_00066

Trace References:

none

Content:

Caveat: The function requires previous initialization (EthIf_Init).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #75637: [diverse] Remove requirements like initializiation requirements regarding the <Module>_Init API

  Problem description:

  Generally a Caveat can never start a requirement. For this all basic Software module shall be checked and cleaned up for this formulation.

  original finding was:

  According to the discussion of RfC# 67678 we detect requirements regarding the initializiation of the certain api's, e.g.:
  "
  ...
  [SWS_EthIf_00213] ?
  Caveat: The function requires previous initialization (EthIf_Init). ?()
  "
  Such requirment are not implementation relevant. Please remove this and add as note.
  –Last change on issue 75637 comment 2–

  Agreed solution:

  General:
  - Remove requirements which state that previous initialization of <Modul>_Init() is required, which are formulated like a note, e.g. : [SWS_EthIf_00213] ?
  Caveat: The function requires previous initialization (EthIf_Init). ?()
  - Add general requirement as note to ch. 8.3 Function definitions
  - Remove or reformulate all other requirements, which start with caveat. Most of them are simply a note or implementation hint.

  === EhtIf ===
  ~8.3 Function definitions
  +Note: All functions in this chapter requires previous initialization (EthIf_Init), except the following ones:
  EthIf_Init, EthIf_GetVersionInfo

  ~8.3.2 EthIf_SetControllerMode

-[SWS_EthIf_00038]

~8.3.3 EthIf_GetControllerMode
-[SWS_EthIf_00044]

~8.3.8 EthIf_ CheckWakeup
-[SWS_EthIf_00249]

~8.3.9 EthIf_GetPhysAddr
-[SWS_EthIf_00066]

~8.3.10 EthIf_SetPhysAddr
-[SWS_EthIf_00138]

~8.3.11 EthIf_UpdatePhysAddrFilter
-[SWS_EthIf_00144]

~8.3.12 EthIf_GetPortMacAddr
-[SWS_EthIf_00195]

~8.3.13 EthIf_GetArlTable
-[SWS_EthIf_00201]

8.3.14 EthIf_GetBufferLevel
-[SWS_EthIf_00207]

~8.3.15 EthIf_GetDropCount
-[SWS_EthIf_00213]

~8.3.16 EthIf_StoreConfiguration
-[SWS_EthIf_00218]

~8.3.17 EthIf_ResetConfiguration
-[SWS_EthIf_00223]

8.3.18 EthIf_GetCurrentTime
-[SWS_EthIf_00159]

~8.3.19 EthIf_EnableEgressTimeStamp
-[SWS_EthIf_00165]

~8.3.20 EthIf_GetEgressTimeStamp
-[SWS_EthIf_00171]

~8.3.21 EthIf_GetIngressTimeStamp
-[SWS_EthIf_00177]

~8.3.22 EthIf_SetCorrectionTime
-[SWS_EthIf_00183]

~8.3.23 EthIf_SetGlobalTime
-[SWS_EthIf_00189]

~8.3.24 EthIf_ProvideTxBuffer
-[SWS_EthIf_00074]

~8.3.25 EthIf_Transmit
-[SWS_EthIf_00081]

=== CanIf ===
General note for below items which contain phrases like "The CAN Interface module must be initialized after Power ON": this is not needed anymore cause already stated in SWS_CANIF_00661. So e.g. in cases where result from CanDrv is just passed through the whole requirement can be deleted.

Also it is assumed, that the mindset like defined in SWS_CANIF_00661 is there for all upper layer functions like <User_TxConfirmation> which might be called by CanIf.

Because if CanDrv would not be already initialised it would return E_NOT_OK which is passed through:
-SWS_CANIF_00312
-SWS_CANIF_00316

Because already guarded by SWS_CANIF_00661:
-SWS_CANIF_00334
-SWS_CANIF_00339
-SWS_CANIF_00344
-SWS_CANIF_00349

~SWS_CANIF_00661

All CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall not execute their normal operation and return E_NOT_OK unless the CanIf has been initialized with a preceding call of CanIf_Init().

+SWS_CANIF_????1
If the switch CANIF_PUBLIC_DEV_ERROR_DETECT is enabled, all CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall report to the DET (using CANIF_E_UNINIT) unless the CanIf has been initialized with a preceding call of CanIf_Init().

-SWS_CANIF_00323 change to
Note: During the call of CanIf_Transmit() the buffer of PduInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00329
CanIf_ReadRxPduData() shall not be used for CanIfRxSduId, which are defined to receive multiple CAN-Ids (range reception).

Add note below SWS_CANIF_00329
Note: During the call of of CanIf_ReadRxPduData() the buffer of CanIfRxInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00356
CanIf_SetDynamicTxId() shall not be interrupted by CanIf_Transmit(), if the same L-SDU ID is handled.

-SWS_CANIF_00407 change to
Note: The call context of CanIf_CheckValidation() is either on interrupt level (interrupt mode) or on task level (polling mode).
Caveat: The corresponding CAN controller and transceiver must be switched on via CanTrcv_SetOpMode(Transceiver, CANTRCV_TRCVMODE_NORMAL) and Can_SetControllerMode(Controller, CAN_CS_STARTED) and the corresponding mode indications must have been called.

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
-SWS_CANIF_00737 CanIf_GetTxConfirmationState()
-SWS_CANIF_00870 CanIf_SetBaudrate()

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
-SWS_CANIF_00401 CanIf_CheckWakeup()
-SWS_CANIF_00413 CanIf_TxConfirmation()
-SWS_CANIF_00422 CanIf_RxIndication()
-SWS_CANIF_00432 CanIf_ControllerBusOff()
-SWS_CANIF_00818 CanIf_ConfirmPnAvailability()
-SWS_CANIF_00807 CanIf_ClearTrcvWufFlagIndication()
-SWS_CANIF_00811 CanIf_CheckTrcvWakeFlagIndication()
-SWS_CANIF_00703 CanIf_ControllerModeIndication()
-SWS_CANIF_00709 CanIf_TrcvModeIndication()
-SWS_CANIF_00887 <User_TriggerTransmit>()
-SWS_CANIF_00437 <User_TxConfirmation>()

-SWS_CANIF_00440 change to
Note: Until <User_RxIndication>() returns, CanIf will not access PduInfoPtr. The PduInfoPtr is only valid and can be used by upper layers, until the indication returns. CanIf guarantees that the number of configured bytes for this PduInfoPtr is valid.
Note: The call context of <User_RxIndication>() is either on interrupt level (interrupt mode) or on task level (polling mode).

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00822 <User_ConfirmPnAvailability>()
-SWS_CANIF_00793 <User_ClearTrcvWufFlagIndication>()

-SWS_CANIF_00799 <User_CheckTrcvWakeFlagIndication>()

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00688 <User_ControllerModeIndication>()

=== ComM ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (ComM_Init), except the following ones:
ComM_Init, ComM_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the ComM module is initialized correctly.
-[SWS_ComM_00805] Caveats of ComM_Nm_NetworkStartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00806] Caveats of ComM_Nm_NetworkMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00808] Caveats of ComM_Nm_PrepareBusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00810] Caveats of ComM_Nm_BusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00812] Caveats of ComM_Nm_RestartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00814] Caveats of ComM_EcuM_WakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00965] Caveats of ComM_EcuM_PNCWakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00816] Caveats of ComM_BusSM_ModeIndication(): The Communication Manager Module is initialized correctly.()

=== LdCom ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (LdCom_Init), except the following ones:
LdCom_Init, LdCom_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the LdCom module is initialized correctly.
–Last change on issue 75637 comment 23–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.15 Specification Item SWS_EthIf_00067

Trace References:

none

Content:

Service name:	EthIf_ProvideTxBufferEthIf_ProvideTxBuffer	
Syntax:	BufReq_ReturnType EthIf_ProvideTxBuffer(uint8 CtrlIdx, Eth_FrameType FrameType, uint8 Priority, Eth_BufIdxType* BufIdxPtr, uint8** BufPtr, uint16* LenBytePtr)	
Service ID[hex]:	0x09	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	CtrlIdxEthIf_ProvideTxBuffer.CtrlIdx	Index of the Ethernet controller within the context of the Ethernet Interface
	FrameTypeEthIf_ProvideTxBuffer.FrameType	Ethernet Frame Type (EtherType)
	PriorityEthIf_ProvideTxBuffer.Priority	Priority value which shall be used for the 3-bit PCP field of the VLAN tag
Parameters (inout):	LenBytePtrEthIf_ProvideTxBuffer.LenBytePtr	in: desired length in bytes, out: granted length in bytes
Parameters (out):	BufIdxPtrEthIf_ProvideTxBuffer.BufIdxPtr	Index to the granted buffer resource. To be used for subsequent requests
	BufPtrEthIf_ProvideTxBuffer.BufPtr	Pointer to the granted buffer
Return value:	BufReq_ReturnType	BUFREQ_OK: success BUFREQ_E_NOT_OK: development error detected BUFREQ_E_BUSY: all buffers in use BUFREQ_E_OVFL: requested buffer too large
Description:	Provides access to a transmit buffer of the specified Ethernet controller.	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #78021: [EthIf] BufReq_ReturnType - BUFREQ_E_OVFL is not handled by the EthIf API - EthIf_ProvideTxBuffer

  Problem description:

  Name: Sathish Madanmohan
  Phone: +44 1904 562639
  Role: Specialist Software Engineer

  Description/Motivation:

  BufReq_ReturnType - BUFREQ_E_OVFL is not handled by the EthIf API - EthIf_ProvideTxBuffer.

  for e.g. If the layer above EthIf requests for a buffer whose length is larger than the actual buffer size then the Eth MCAL returns 'BUFREQ_E_OVFL' to the EthIf. This must then be communicated to the above TCP layer. But there is no description found in the EthIf specification about BUFREQ_E_OVFL. This makes the buffer overflow functionality incomplete.

  There is no way this overflow criteria can be communicated to the layers above EthIf !

  This must be fixed.

  Was there already a decision?

  No

  Agreed solution:

  ~SWS_EthIf_00067
  old:
  Return value: BufReq_ReturnType BUFREQ_OK: success
  BUFREQ_E_NOT_OK: development error detected
  BUFREQ_E_BUSY: all buffers in use
  new:
  Return value: BufReq_ReturnType BUFREQ_OK: success
  BUFREQ_E_NOT_OK: development error detected

BUFREQ_E_BUSY: all buffers in use
BUFREQ_E_OVFL: requested buffer too large
–Last change on issue 78021 comment 2–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.16 Specification Item SWS_EthIf_00069

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZED UNINIT and return BUFREQ_E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification "belongs".

  *** BSW UML Model ***
  SWS_CanNm:
  ———-

Chapter 8.6.1 Optional Interfaces:
Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

SWS_LinIf:
———-
SWS_LinIf_00359: add Det_ReportRuntimeError

SWS_UdpNm:
———-
Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.17 Specification Item SWS_EthIf_00074

Trace References:

none

Content:

Caveat: The function requires previous initialization (EthIf_Init).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #75637: [diverse] Remove requirements like initializiation requirements regarding the <Module>_Init API

  Problem description:

  Generally a Caveat can never start a requirement. For this all basic Software module shall be checked and cleaned up for this formulation.

  original finding was:

According to the discussion of RfC# 67678 we detect requirements regarding the initializiation of the certain api's, e.g.:
"
...
[SWS_EthIf_00213] ?
Caveat: The function requires previous initialization (EthIf_Init). ?()
"
Such requirment are not implementation relevant. Please remove this and add as note.
–Last change on issue 75637 comment 2–

Agreed solution:

General:
- Remove requirements which state that previous initialization of <Modul>_Init() is required, which are formulated like a note, e.g. : [SWS_EthIf_00213] ?
Caveat: The function requires previous initialization (EthIf_Init). ?()
- Add general requirement as note to ch. 8.3 Function definitions
- Remove or reformulate all other requirements, which start with caveat. Most of them are simply a note or implementation hint.

=== EhtIf ===
~8.3 Function definitions
+Note: All functions in this chapter requires previous initialization (EthIf_Init), except the following ones:
EthIf_Init, EthIf_GetVersionInfo

~8.3.2 EthIf_SetControllerMode
-[SWS_EthIf_00038]

~8.3.3 EthIf_GetControllerMode
-[SWS_EthIf_00044]

~8.3.8 EthIf_ CheckWakeup
-[SWS_EthIf_00249]

~8.3.9 EthIf_GetPhysAddr
-[SWS_EthIf_00066]

~8.3.10 EthIf_SetPhysAddr
-[SWS_EthIf_00138]

~8.3.11 EthIf_UpdatePhysAddrFilter

-[SWS_EthIf_00144]

~8.3.12 EthIf_GetPortMacAddr
-[SWS_EthIf_00195]

~8.3.13 EthIf_GetArlTable
-[SWS_EthIf_00201]

8.3.14 EthIf_GetBufferLevel
-[SWS_EthIf_00207]

~8.3.15 EthIf_GetDropCount
-[SWS_EthIf_00213]

~8.3.16 EthIf_StoreConfiguration
-[SWS_EthIf_00218]

~8.3.17 EthIf_ResetConfiguration
-[SWS_EthIf_00223]

8.3.18 EthIf_GetCurrentTime
-[SWS_EthIf_00159]

~8.3.19 EthIf_EnableEgressTimeStamp
-[SWS_EthIf_00165]

~8.3.20 EthIf_GetEgressTimeStamp
-[SWS_EthIf_00171]

~8.3.21 EthIf_GetIngressTimeStamp
-[SWS_EthIf_00177]

~8.3.22 EthIf_SetCorrectionTime
-[SWS_EthIf_00183]

~8.3.23 EthIf_SetGlobalTime
-[SWS_EthIf_00189]

~8.3.24 EthIf_ProvideTxBuffer
-[SWS_EthIf_00074]

~8.3.25 EthIf_Transmit
-[SWS_EthIf_00081]

=== CanIf ===
General note for below items which contain phrases like "The CAN Interface module must be initialized after Power ON": this is not needed anymore cause already stated in SWS_CANIF_00661. So e.g. in cases where result from CanDrv is just passed through the whole requirement can be deleted.

Also it is assumed, that the mindset like defined in SWS_CANIF_00661 is there for all upper layer functions like <User_TxConfirmation> which might be called by CanIf.

Because if CanDrv would not be already initialised it would return E_NOT_OK which is passed through:
-SWS_CANIF_00312
-SWS_CANIF_00316

Because already guarded by SWS_CANIF_00661:
-SWS_CANIF_00334
-SWS_CANIF_00339
-SWS_CANIF_00344
-SWS_CANIF_00349

~SWS_CANIF_00661
All CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall not execute their normal operation and return E_NOT_OK unless the CanIf has been initialized with a preceding call of CanIf_Init().

+SWS_CANIF_????1
If the switch CANIF_PUBLIC_DEV_ERROR_DETECT is enabled, all CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall report to the DET (using CANIF_E_UNINIT) unless the CanIf has been initialized with a preceding call of CanIf_Init().

-SWS_CANIF_00323 change to
Note: During the call of CanIf_Transmit() the buffer of PduInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context.

After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00329
CanIf_ReadRxPduData() shall not be used for CanIfRxSduId, which are defined to receive multiple CAN-Ids (range reception).

Add note below SWS_CANIF_00329
Note: During the call of of CanIf_ReadRxPduData() the buffer of CanIfRxInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00356
CanIf_SetDynamicTxId() shall not be interrupted by CanIf_Transmit(), if the same L-SDU ID is handled.

-SWS_CANIF_00407 change to
Note: The call context of CanIf_CheckValidation() is either on interrupt level (interrupt mode) or on task level (polling mode).
Caveat: The corresponding CAN controller and transceiver must be switched on via CanTrcv_SetOpMode(Transceiver, CANTRCV_TRCVMODE_NORMAL) and Can_SetControllerMode(Controller, CAN_CS_STARTED) and the corresponding mode indications must have been called.

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
-SWS_CANIF_00737 CanIf_GetTxConfirmationState()
-SWS_CANIF_00870 CanIf_SetBaudrate()

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
-SWS_CANIF_00401 CanIf_CheckWakeup()
-SWS_CANIF_00413 CanIf_TxConfirmation()
-SWS_CANIF_00422 CanIf_RxIndication()

-SWS_CANIF_00432 CanIf_ControllerBusOff()
-SWS_CANIF_00818 CanIf_ConfirmPnAvailability()
-SWS_CANIF_00807 CanIf_ClearTrcvWufFlagIndication()
-SWS_CANIF_00811 CanIf_CheckTrcvWakeFlagIndication()
-SWS_CANIF_00703 CanIf_ControllerModeIndication()
-SWS_CANIF_00709 CanIf_TrcvModeIndication()
-SWS_CANIF_00887 <User_TriggerTransmit>()
-SWS_CANIF_00437 <User_TxConfirmation>()

-SWS_CANIF_00440 change to
Note: Until <User_RxIndication>() returns, CanIf will not access PduInfoPtr. The PduInfoPtr is only valid and can be used by upper layers, until the indication returns. CanIf guarantees that the number of configured bytes for this PduInfoPtr is valid.
Note: The call context of <User_RxIndication>() is either on interrupt level (interrupt mode) or on task level (polling mode).

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00822 <User_ConfirmPnAvailability>()
-SWS_CANIF_00793 <User_ClearTrcvWufFlagIndication>()
-SWS_CANIF_00799 <User_CheckTrcvWakeFlagIndication>()

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00688 <User_ControllerModeIndication>()

=== ComM ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (ComM_Init), except the following ones:

ComM_Init, ComM_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the ComM module is initialized correctly.
-[SWS_ComM_00805] Caveats of ComM_Nm_NetworkStartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00806] Caveats of ComM_Nm_NetworkMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00808] Caveats of ComM_Nm_PrepareBusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00810] Caveats of ComM_Nm_BusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00812] Caveats of ComM_Nm_RestartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00814] Caveats of ComM_EcuM_WakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00965] Caveats of ComM_EcuM_PNCWakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00816] Caveats of ComM_BusSM_ModeIndication(): The Communication Manager Module is initialized correctly.()

=== LdCom ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (LdCom_Init), except the following ones:
LdCom_Init, LdCom_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the LdCom module is initialized correctly.
–Last change on issue 75637 comment 23–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.18 Specification Item SWS_EthIf_00075

Trace References:

none

Content:

Service name:	EthIf_TransmitEthIf_Transmit	
Syntax:	Std_ReturnType EthIf_Transmit(uint8 CtrlIdx, Eth_BufIdxType* Type BufIdx, Eth_FrameType FrameType, boolean TxConfirmation, uint16* uint16 LenByte, const uint8* PhysAddrPtr)	
Service ID[hex]:	0x0a	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant for different buffer indexes and Ctrl indexes	
Parameters (in):	CtrlIdxEthIf_Transmit.CtrlIdx	Index of the Ethernet controller within the context of the Ethernet Interface
	BufIdxEthIf_Transmit.BufIdx	Index of the buffer resource
	FrameTypeEthIf_Transmit.FrameType	Ethernet frame type
	TxConfirmationEthIf_Transmit.TxConfirmation	Activates transmission confirmation
	LenByteEthIf_Transmit.LenByte	Data length in byte
	PhysAddrPtrEthIf_Transmit.PhysAddrPtr	Physical target address (MAC address) in network byte order

LenByteEthIf_Transmit.LenByte Data length in byte

Parameters (inout):	None	

BufIdxEthIf_Transmit.BufIdx Index of the buffer resource

Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: success E_NOT_OK: transmission failed
Description:	Triggers transmission of a previously filled transmit buffer	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76631: [EthIf] Parameters of EthIf_Transmit have wrong directions

  Problem description:

  With AUTOSAR 4.3.0 the syntax of EthIf_Transmit(). Both BufIdx and LenByte became pointers.

  Syntax based on AUTOSAR 4.2.2
  Std_ReturnType EthIf_Transmit(
  uint8 CtrlIdx,
  Eth_BufIdxType BufIdx,
  Eth_FrameType FrameType,
  boolean TxConfirmation,

uint16 LenByte,
const uint8* PhysAddrPtr
)

Syntax based on AUTOSAR 4.3.0
Std_ReturnType EthIf_Transmit(
uint8 CtrlIdx,
Eth_BufIdxType* BufIdx,
Eth_FrameType FrameType,
boolean TxConfirmation,
uint16* LenByte,
const uint8* PhysAddrPtr
)

The changes were done to meet the description of these parameter but it does not make sense to change the direction from functional point of view.

BufIdx is an OUT parameter of EthIf_ProvideTxBuffer() to be used as IN parameter for subsequent call of EthIf_Transmit().

LenByte defines the number of bytes to be transmitted with EthIf_Transmit(). A pointer (IN/OUT) does not make sense since the IN value would also be the OUT value.

Agreed solution:

Revert syntax changes of SWS_EthIf_00075 and correct the direction in the parameters parameter description for BufIdx and LenBytes. Both shall be IN parameters
Std_ReturnType EthIf_Transmit(
uint8 CtrlIdx,
Eth_BufIdxType BufIdx, [IN]
Eth_FrameType FrameType,
boolean TxConfirmation,
uint16 LenByte,[IN]
const uint8* PhysAddrPtr
)

~ Figure 7 in chapter 9.3 shall be corrected (syntax of EthIf_Transmit)
–Last change on issue 76631 comment 11–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.19 Specification Item SWS_EthIf_00077

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZED UNINIT otherwise (if DET is disabled) return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification "belongs".

  *** BSW UML Model ***
  SWS_CanNm:
  ———-
  Chapter 8.6.1 Optional Interfaces:
  Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

  SWS_LinIf:
  ———-
  SWS_LinIf_00359: add Det_ReportRuntimeError

  SWS_UdpNm:

———-

Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.20 Specification Item SWS_EthIf_00081

Trace References:

none

Content:

Caveat: The function requires previous buffer request (EthIf_ProvideTxBuffer).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #75637: [diverse] Remove requirements like initializiation requirements regarding the <Module>_Init API

  Problem description:

  Generally a Caveat can never start a requirement. For this all basic Software module shall be checked and cleaned up for this formulation.

  original finding was:

  According to the discussion of RfC# 67678 we detect requirements regarding the initializiation of the certain api's, e.g.:
  "
  ...
  [SWS_EthIf_00213] ?
  Caveat: The function requires previous initialization (EthIf_Init). ?()
  "
  Such requirment are not implementation relevant. Please remove this and add as

note.
–Last change on issue 75637 comment 2–

Agreed solution:

General:
- Remove requirements which state that previous initialization of <Modul>_Init() is required, which are formulated like a note, e.g. : [SWS_EthIf_00213] ?
Caveat: The function requires previous initialization (EthIf_Init). ?()
- Add general requirement as note to ch. 8.3 Function definitions
- Remove or reformulate all other requirements, which start with caveat. Most of them are simply a note or implementation hint.

=== EhtIf ===
~8.3 Function definitions
+Note: All functions in this chapter requires previous initialization (EthIf_Init), except the following ones:
EthIf_Init, EthIf_GetVersionInfo

~8.3.2 EthIf_SetControllerMode
-[SWS_EthIf_00038]

~8.3.3 EthIf_GetControllerMode
-[SWS_EthIf_00044]

~8.3.8 EthIf_ CheckWakeup
-[SWS_EthIf_00249]

~8.3.9 EthIf_GetPhysAddr
-[SWS_EthIf_00066]

~8.3.10 EthIf_SetPhysAddr
-[SWS_EthIf_00138]

~8.3.11 EthIf_UpdatePhysAddrFilter
-[SWS_EthIf_00144]

~8.3.12 EthIf_GetPortMacAddr
-[SWS_EthIf_00195]

~8.3.13 EthIf_GetArlTable
-[SWS_EthIf_00201]

8.3.14 EthIf_GetBufferLevel
-[SWS_EthIf_00207]

~8.3.15 EthIf_GetDropCount
-[SWS_EthIf_00213]

~8.3.16 EthIf_StoreConfiguration
-[SWS_EthIf_00218]

~8.3.17 EthIf_ResetConfiguration
-[SWS_EthIf_00223]

8.3.18 EthIf_GetCurrentTime
-[SWS_EthIf_00159]

~8.3.19 EthIf_EnableEgressTimeStamp
-[SWS_EthIf_00165]

~8.3.20 EthIf_GetEgressTimeStamp
-[SWS_EthIf_00171]

~8.3.21 EthIf_GetIngressTimeStamp
-[SWS_EthIf_00177]

~8.3.22 EthIf_SetCorrectionTime
-[SWS_EthIf_00183]

~8.3.23 EthIf_SetGlobalTime
-[SWS_EthIf_00189]

~8.3.24 EthIf_ProvideTxBuffer
-[SWS_EthIf_00074]

~8.3.25 EthIf_Transmit
-[SWS_EthIf_00081]

=== CanIf ===
General note for below items which contain phrases like "The CAN Interface module must be initialized after Power ON": this is not needed anymore cause already stated in SWS_CANIF_00661. So e.g. in cases where result from CanDrv is just passed through the whole requirement can be deleted.

Also it is assumed, that the mindset like defined in SWS_CANIF_00661 is there for all upper layer functions like <User_TxConfirmation> which might be called by CanIf.

Because if CanDrv would not be already initialised it would return E_NOT_OK which is passed through:
-SWS_CANIF_00312
-SWS_CANIF_00316

Because already guarded by SWS_CANIF_00661:
-SWS_CANIF_00334
-SWS_CANIF_00339
-SWS_CANIF_00344
-SWS_CANIF_00349

~SWS_CANIF_00661
All CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall not execute their normal operation and return E_NOT_OK unless the CanIf has been initialized with a preceding call of CanIf_Init().

+SWS_CANIF_????1
If the switch CANIF_PUBLIC_DEV_ERROR_DETECT is enabled, all CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall report to the DET (using CANIF_E_UNINIT) unless the CanIf has been initialized with a preceding call of CanIf_Init().

-SWS_CANIF_00323 change to
Note: During the call of CanIf_Transmit() the buffer of PduInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00329
CanIf_ReadRxPduData() shall not be used for CanIfRxSduId, which are defined to receive multiple CAN-Ids (range reception).

Add note below SWS_CANIF_00329
Note: During the call of of CanIf_ReadRxPduData() the buffer of CanIfRxInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00356
CanIf_SetDynamicTxId() shall not be interrupted by CanIf_Transmit(), if the same L-SDU ID is handled.

-SWS_CANIF_00407 change to
Note: The call context of CanIf_CheckValidation() is either on interrupt level (interrupt mode) or on task level (polling mode).
Caveat: The corresponding CAN controller and transceiver must be switched on via CanTrcv_SetOpMode(Transceiver, CANTRCV_TRCVMODE_NORMAL) and Can_SetControllerMode(Controller, CAN_CS_STARTED) and the corresponding mode indications must have been called.

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
-SWS_CANIF_00737 CanIf_GetTxConfirmationState()
-SWS_CANIF_00870 CanIf_SetBaudrate()

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
-SWS_CANIF_00401 CanIf_CheckWakeup()
-SWS_CANIF_00413 CanIf_TxConfirmation()
-SWS_CANIF_00422 CanIf_RxIndication()
-SWS_CANIF_00432 CanIf_ControllerBusOff()
-SWS_CANIF_00818 CanIf_ConfirmPnAvailability()
-SWS_CANIF_00807 CanIf_ClearTrcvWufFlagIndication()
-SWS_CANIF_00811 CanIf_CheckTrcvWakeFlagIndication()
-SWS_CANIF_00703 CanIf_ControllerModeIndication()
-SWS_CANIF_00709 CanIf_TrcvModeIndication()
-SWS_CANIF_00887 <User_TriggerTransmit>()
-SWS_CANIF_00437 <User_TxConfirmation>()

-SWS_CANIF_00440 change to
Note: Until <User_RxIndication>() returns, CanIf will not access PduInfoPtr. The PduInfoPtr is only valid and can be used by upper layers, until the indication returns. CanIf guarantees that the number of configured bytes for this PduInfoPtr is valid.
Note: The call context of <User_RxIndication>() is either on interrupt level (interrupt mode) or on task level (polling mode).

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00822 <User_ConfirmPnAvailability>()
-SWS_CANIF_00793 <User_ClearTrcvWufFlagIndication>()
-SWS_CANIF_00799 <User_CheckTrcvWakeFlagIndication>()

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00688 <User_ControllerModeIndication>()

=== ComM ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (ComM_Init), except the following ones:
ComM_Init, ComM_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the ComM module is initialized correctly.
-[SWS_ComM_00805] Caveats of ComM_Nm_NetworkStartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00806] Caveats of ComM_Nm_NetworkMode: The Communication

Manager Module is initialized correctly.()
-[SWS_ComM_00808] Caveats of ComM_Nm_PrepareBusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00810] Caveats of ComM_Nm_BusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00812] Caveats of ComM_Nm_RestartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00814] Caveats of ComM_EcuM_WakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00965] Caveats of ComM_EcuM_PNCWakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00816] Caveats of ComM_BusSM_ModeIndication(): The Communication Manager Module is initialized correctly.()

=== LdCom ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (LdCom_Init), except the following ones:
LdCom_Init, LdCom_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the LdCom module is initialized correctly.
–Last change on issue 75637 comment 23–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.21 Specification Item SWS_EthIf_00085

Trace References:

none

Content:

Service name:	EthIf_RxIndicationEthIf_RxIndication

Syntax:	void EthIf_RxIndication(uint8 CtrlIdx, Eth_FrameType FrameType, boolean IsBroadcast, const uint8* PhysAddrPtr, const Eth_DataType* DataPtr, uint16 LenByte)	
Service ID[hex]:	0x10	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	CtrlIdxEthIf_RxIndication.CtrlIdx	Index of the physical Ethernet controller within the context of the Ethernet Interface
	FrameTypeEthIf_RxIndication.FrameType	Frame type of received Ethernet frame
	IsBroadcastEthIf_RxIndication.IsBroadcast	parameter to indicate a broadcast frame
	PhysAddrPtrEthIf_RxIndication.PhysAddrPtr	Pointer to Physical source address (MAC address in network byte order) of received Ethernet frame
	DataPtrEthIf_RxIndication.DataPtr	Pointer to payload of received Ethernet frame.
	LenByteEthIf_RxIndication.LenByte	Length of the received frame bytes (bytes) of the payload in received frame.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	Handles a received frame received by the indexed controller	

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* Authenticated, uint8* AuthenticationReqData, uint8* AuthenticationResData)

To:
Std_ReturnType <User>_DoIPRoutingActivationConfirmation(boolean* Confirmed, const uint8* ConfirmationReqData, uint8* ConfirmationResData)
Std_ReturnType <User>_DoIPRoutingActivationAuthentication(boolean* Authenticated, 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 #76835: [ETHIF] EthIf_RxIndication description of LenByte parameter not meaningful

Problem description:

As per AUTOSAR_SWS_EthernetInterface.pdf for AUTOSAR 4.0.3
EthIf_RxIndication has DataPtr and LenByte, IN parameters as below :

DataPtr : Ethernet frame comprising the following elements in the listed order: Target MAC, Source MAC, VLAN tag (optional), Type, Payload.
LenByte : Length of the received frame bytes

As per AUTOSAR_SWS_EthernetInterface.pdf for AUTOSAR revision 4.1 and above
EthIf_RxIndication has DataPtr and LenByte, IN parameters as below :

DataPtr : Pointer to payload of received Ethernet frame.
LenByte : Length of the received frame bytes

The description of DataPtr has changed in AUTOSAR 4.1, corresponding description of LenByte of has not changed.
When DataPtr points to payload there is no meaning in LenByte providing the whole frame length which includes
length of Target MAC, Source MAC, VLAN tag (optional), Type and Payload.
–Last change on issue 76835 comment 1–

Agreed solution:

CP_SWS_EthernetInterface
[SWS_EthIf_00085]
LenByte : Length of the received frame bytes
to be modified to
LenByte :Length (bytes) of the payload in received frame.
–Last change on issue 76835 comment 13–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.22 Specification Item SWS_EthIf_00086

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZEDUNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification "belongs".

  *** BSW UML Model ***
  SWS_CanNm:
  ———-
  Chapter 8.6.1 Optional Interfaces:
  Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

  SWS_LinIf:
  ———-

SWS_LinIf_00359: add Det_ReportRuntimeError

SWS_UdpNm:
———-
Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.23 Specification Item SWS_EthIf_00089

Trace References:

none

Content:

Caveat: The function requires previous initialization (EthIf_Init).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #75637: [diverse] Remove requirements like initializiation requirements regarding the <Module>_Init API

  Problem description:

  Generally a Caveat can never start a requirement. For this all basic Software module shall be checked and cleaned up for this formulation.

  original finding was:

  According to the discussion of RfC# 67678 we detect requirements regarding the initializiation of the certain api's, e.g.:
  "
  ...
  [SWS_EthIf_00213] ?

Caveat: The function requires previous initialization (EthIf_Init). ?()
"
Such requirment are not implementation relevant. Please remove this and add as note.
–Last change on issue 75637 comment 2–

Agreed solution:

General:
- Remove requirements which state that previous initialization of <Modul>_Init() is required, which are formulated like a note, e.g. : [SWS_EthIf_00213] ?
Caveat: The function requires previous initialization (EthIf_Init). ?()
- Add general requirement as note to ch. 8.3 Function definitions
- Remove or reformulate all other requirements, which start with caveat. Most of them are simply a note or implementation hint.

=== EhtIf ===
~8.3 Function definitions
+Note: All functions in this chapter requires previous initialization (EthIf_Init), except the following ones:
EthIf_Init, EthIf_GetVersionInfo

~8.3.2 EthIf_SetControllerMode
-[SWS_EthIf_00038]

~8.3.3 EthIf_GetControllerMode
-[SWS_EthIf_00044]

~8.3.8 EthIf_ CheckWakeup
-[SWS_EthIf_00249]

~8.3.9 EthIf_GetPhysAddr
-[SWS_EthIf_00066]

~8.3.10 EthIf_SetPhysAddr
-[SWS_EthIf_00138]

~8.3.11 EthIf_UpdatePhysAddrFilter
-[SWS_EthIf_00144]

~8.3.12 EthIf_GetPortMacAddr
-[SWS_EthIf_00195]

~8.3.13 EthIf_GetArlTable
-[SWS_EthIf_00201]

8.3.14 EthIf_GetBufferLevel
-[SWS_EthIf_00207]

~8.3.15 EthIf_GetDropCount
-[SWS_EthIf_00213]

~8.3.16 EthIf_StoreConfiguration
-[SWS_EthIf_00218]

~8.3.17 EthIf_ResetConfiguration
-[SWS_EthIf_00223]

8.3.18 EthIf_GetCurrentTime
-[SWS_EthIf_00159]

~8.3.19 EthIf_EnableEgressTimeStamp
-[SWS_EthIf_00165]

~8.3.20 EthIf_GetEgressTimeStamp
-[SWS_EthIf_00171]

~8.3.21 EthIf_GetIngressTimeStamp
-[SWS_EthIf_00177]

~8.3.22 EthIf_SetCorrectionTime
-[SWS_EthIf_00183]

~8.3.23 EthIf_SetGlobalTime
-[SWS_EthIf_00189]

~8.3.24 EthIf_ProvideTxBuffer
-[SWS_EthIf_00074]

~8.3.25 EthIf_Transmit
-[SWS_EthIf_00081]

=== CanIf ===
General note for below items which contain phrases like "The CAN Interface module must be initialized after Power ON": this is not needed anymore cause already

stated in SWS_CANIF_00661. So e.g. in cases where result from CanDrv is just passed through the whole requirement can be deleted.

Also it is assumed, that the mindset like defined in SWS_CANIF_00661 is there for all upper layer functions like <User_TxConfirmation> which might be called by CanIf.

Because if CanDrv would not be already initialised it would return E_NOT_OK which is passed through:
-SWS_CANIF_00312
-SWS_CANIF_00316

Because already guarded by SWS_CANIF_00661:
-SWS_CANIF_00334
-SWS_CANIF_00339
-SWS_CANIF_00344
-SWS_CANIF_00349

~SWS_CANIF_00661
All CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall not execute their normal operation and return E_NOT_OK unless the CanIf has been initialized with a preceding call of CanIf_Init().

+SWS_CANIF_????1
If the switch CANIF_PUBLIC_DEV_ERROR_DETECT is enabled, all CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall report to the DET (using CANIF_E_UNINIT) unless the CanIf has been initialized with a preceding call of CanIf_Init().

-SWS_CANIF_00323 change to
Note: During the call of CanIf_Transmit() the buffer of PduInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00329
CanIf_ReadRxPduData() shall not be used for CanIfRxSduId, which are defined to

receive multiple CAN-Ids (range reception).

Add note below SWS_CANIF_00329
Note: During the call of of CanIf_ReadRxPduData() the buffer of CanIfRxInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00356
CanIf_SetDynamicTxId() shall not be interrupted by CanIf_Transmit(), if the same L-SDU ID is handled.

-SWS_CANIF_00407 change to
Note: The call context of CanIf_CheckValidation() is either on interrupt level (interrupt mode) or on task level (polling mode).
Caveat: The corresponding CAN controller and transceiver must be switched on via CanTrcv_SetOpMode(Transceiver, CANTRCV_TRCVMODE_NORMAL) and Can_SetControllerMode(Controller, CAN_CS_STARTED) and the corresponding mode indications must have been called.

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
-SWS_CANIF_00737 CanIf_GetTxConfirmationState()
-SWS_CANIF_00870 CanIf_SetBaudrate()

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
-SWS_CANIF_00401 CanIf_CheckWakeup()
-SWS_CANIF_00413 CanIf_TxConfirmation()
-SWS_CANIF_00422 CanIf_RxIndication()
-SWS_CANIF_00432 CanIf_ControllerBusOff()
-SWS_CANIF_00818 CanIf_ConfirmPnAvailability()
-SWS_CANIF_00807 CanIf_ClearTrcvWufFlagIndication()
-SWS_CANIF_00811 CanIf_CheckTrcvWakeFlagIndication()
-SWS_CANIF_00703 CanIf_ControllerModeIndication()

-SWS_CANIF_00709 CanIf_TrcvModeIndication()
-SWS_CANIF_00887 <User_TriggerTransmit>()
-SWS_CANIF_00437 <User_TxConfirmation>()

-SWS_CANIF_00440 change to
Note: Until <User_RxIndication>() returns, CanIf will not access PduInfoPtr. The PduInfoPtr is only valid and can be used by upper layers, until the indication returns. CanIf guarantees that the number of configured bytes for this PduInfoPtr is valid.
Note: The call context of <User_RxIndication>() is either on interrupt level (interrupt mode) or on task level (polling mode).

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00822 <User_ConfirmPnAvailability>()
-SWS_CANIF_00793 <User_ClearTrcvWufFlagIndication>()
-SWS_CANIF_00799 <User_CheckTrcvWakeFlagIndication>()

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00688 <User_ControllerModeIndication>()

=== ComM ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (ComM_Init), except the following ones:
ComM_Init, ComM_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the ComM module is initialized correctly.

-[SWS_ComM_00805] Caveats of ComM_Nm_NetworkStartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00806] Caveats of ComM_Nm_NetworkMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00808] Caveats of ComM_Nm_PrepareBusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00810] Caveats of ComM_Nm_BusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00812] Caveats of ComM_Nm_RestartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00814] Caveats of ComM_EcuM_WakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00965] Caveats of ComM_EcuM_PNCWakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00816] Caveats of ComM_BusSM_ModeIndication(): The Communication Manager Module is initialized correctly.()

=== LdCom ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (LdCom_Init), except the following ones:
LdCom_Init, LdCom_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the LdCom module is initialized correctly.
–Last change on issue 75637 comment 23–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.24 Specification Item SWS_EthIf_00090

Trace References:

none

Content:

Caveat: The function shall be callable on interrupt level.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #75637: [diverse] Remove requirements like initializiation requirements regarding the <Module>_Init API

  Problem description:

  Generally a Caveat can never start a requirement. For this all basic Software module shall be checked and cleaned up for this formulation.

  original finding was:

  According to the discussion of RfC# 67678 we detect requirements regarding the initializiation of the certain api's, e.g.:
  "
  ...
  [SWS_EthIf_00213] ?
  Caveat: The function requires previous initialization (EthIf_Init). ?()
  "
  Such requirment are not implementation relevant. Please remove this and add as note.
  –Last change on issue 75637 comment 2–

  Agreed solution:

  General:
  - Remove requirements which state that previous initialization of <Modul>_Init() is required, which are formulated like a note, e.g. : [SWS_EthIf_00213] ?
  Caveat: The function requires previous initialization (EthIf_Init). ?()
  - Add general requirement as note to ch. 8.3 Function definitions
  - Remove or reformulate all other requirements, which start with caveat. Most of them are simply a note or implementation hint.

  === EhtIf ===
  ~8.3 Function definitions
  +Note: All functions in this chapter requires previous initialization (EthIf_Init), except the following ones:
  EthIf_Init, EthIf_GetVersionInfo

  ~8.3.2 EthIf_SetControllerMode
  -[SWS_EthIf_00038]

  ~8.3.3 EthIf_GetControllerMode
  -[SWS_EthIf_00044]

  ~8.3.8 EthIf_ CheckWakeup

-[SWS_EthIf_00249]

~8.3.9 EthIf_GetPhysAddr
-[SWS_EthIf_00066]

~8.3.10 EthIf_SetPhysAddr
-[SWS_EthIf_00138]

~8.3.11 EthIf_UpdatePhysAddrFilter
-[SWS_EthIf_00144]

~8.3.12 EthIf_GetPortMacAddr
-[SWS_EthIf_00195]

~8.3.13 EthIf_GetArlTable
-[SWS_EthIf_00201]

8.3.14 EthIf_GetBufferLevel
-[SWS_EthIf_00207]

~8.3.15 EthIf_GetDropCount
-[SWS_EthIf_00213]

~8.3.16 EthIf_StoreConfiguration
-[SWS_EthIf_00218]

~8.3.17 EthIf_ResetConfiguration
-[SWS_EthIf_00223]

8.3.18 EthIf_GetCurrentTime
-[SWS_EthIf_00159]

~8.3.19 EthIf_EnableEgressTimeStamp
-[SWS_EthIf_00165]

~8.3.20 EthIf_GetEgressTimeStamp
-[SWS_EthIf_00171]

~8.3.21 EthIf_GetIngressTimeStamp
-[SWS_EthIf_00177]

~8.3.22 EthIf_SetCorrectionTime
-[SWS_EthIf_00183]

~8.3.23 EthIf_SetGlobalTime
-[SWS_EthIf_00189]

~8.3.24 EthIf_ProvideTxBuffer
-[SWS_EthIf_00074]

~8.3.25 EthIf_Transmit
-[SWS_EthIf_00081]

=== CanIf ===
General note for below items which contain phrases like "The CAN Interface module must be initialized after Power ON": this is not needed anymore cause already stated in SWS_CANIF_00661. So e.g. in cases where result from CanDrv is just passed through the whole requirement can be deleted.

Also it is assumed, that the mindset like defined in SWS_CANIF_00661 is there for all upper layer functions like <User_TxConfirmation> which might be called by CanIf.

Because if CanDrv would not be already initialised it would return E_NOT_OK which is passed through:
-SWS_CANIF_00312
-SWS_CANIF_00316

Because already guarded by SWS_CANIF_00661:
-SWS_CANIF_00334
-SWS_CANIF_00339
-SWS_CANIF_00344
-SWS_CANIF_00349

~SWS_CANIF_00661
All CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall not execute their normal operation and return E_NOT_OK unless the CanIf has been initialized with a preceding call of CanIf_Init().

+SWS_CANIF_????1

If the switch CANIF_PUBLIC_DEV_ERROR_DETECT is enabled, all CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall report to the DET (using CANIF_E_UNINIT) unless the CanIf has been initialized with a preceding call of CanIf_Init().

-SWS_CANIF_00323 change to
Note: During the call of CanIf_Transmit() the buffer of PduInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00329
CanIf_ReadRxPduData() shall not be used for CanIfRxSduId, which are defined to receive multiple CAN-Ids (range reception).

Add note below SWS_CANIF_00329
Note: During the call of of CanIf_ReadRxPduData() the buffer of CanIfRxInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00356
CanIf_SetDynamicTxId() shall not be interrupted by CanIf_Transmit(), if the same L-SDU ID is handled.

-SWS_CANIF_00407 change to
Note: The call context of CanIf_CheckValidation() is either on interrupt level (interrupt mode) or on task level (polling mode).
Caveat: The corresponding CAN controller and transceiver must be switched on via CanTrcv_SetOpMode(Transceiver, CANTRCV_TRCVMODE_NORMAL) and Can_SetControllerMode(Controller, CAN_CS_STARTED) and the corresponding mode indications must have been called.

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
-SWS_CANIF_00737 CanIf_GetTxConfirmationState()
-SWS_CANIF_00870 CanIf_SetBaudrate()

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
-SWS_CANIF_00401 CanIf_CheckWakeup()
-SWS_CANIF_00413 CanIf_TxConfirmation()
-SWS_CANIF_00422 CanIf_RxIndication()
-SWS_CANIF_00432 CanIf_ControllerBusOff()
-SWS_CANIF_00818 CanIf_ConfirmPnAvailability()
-SWS_CANIF_00807 CanIf_ClearTrcvWufFlagIndication()
-SWS_CANIF_00811 CanIf_CheckTrcvWakeFlagIndication()
-SWS_CANIF_00703 CanIf_ControllerModeIndication()
-SWS_CANIF_00709 CanIf_TrcvModeIndication()
-SWS_CANIF_00887 <User_TriggerTransmit>()
-SWS_CANIF_00437 <User_TxConfirmation>()

-SWS_CANIF_00440 change to
Note: Until <User_RxIndication>() returns, CanIf will not access PduInfoPtr. The PduInfoPtr is only valid and can be used by upper layers, until the indication returns. CanIf guarantees that the number of configured bytes for this PduInfoPtr is valid.
Note: The call context of <User_RxIndication>() is either on interrupt level (interrupt mode) or on task level (polling mode).

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00822 <User_ConfirmPnAvailability>()
-SWS_CANIF_00793 <User_ClearTrcvWufFlagIndication>()
-SWS_CANIF_00799 <User_CheckTrcvWakeFlagIndication>()

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."

"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00688 <User_ControllerModeIndication>()

=== ComM ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (ComM_Init), except the following ones:
ComM_Init, ComM_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the ComM module is initialized correctly.
-[SWS_ComM_00805] Caveats of ComM_Nm_NetworkStartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00806] Caveats of ComM_Nm_NetworkMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00808] Caveats of ComM_Nm_PrepareBusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00810] Caveats of ComM_Nm_BusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00812] Caveats of ComM_Nm_RestartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00814] Caveats of ComM_EcuM_WakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00965] Caveats of ComM_EcuM_PNCWakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00816] Caveats of ComM_BusSM_ModeIndication(): The Communication Manager Module is initialized correctly.()

=== LdCom ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (LdCom_Init), except the following ones:
LdCom_Init, LdCom_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the LdCom module is initialized correctly.
–Last change on issue 75637 comment 23–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.25 Specification Item SWS_EthIf_00092

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZEDUNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of ’Runtime errors’

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification "belongs".

  *** BSW UML Model ***
  SWS_CanNm:
  ———-
  Chapter 8.6.1 Optional Interfaces:
  Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

  SWS_LinIf:
  ———-

SWS_LinIf_00359: add Det_ReportRuntimeError

SWS_UdpNm:
———-
Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.26 Specification Item SWS_EthIf_00095

Trace References:

none

Content:

Caveat: The function requires previous initialization (EthIf_Init).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #75637: [diverse] Remove requirements like initializiation requirements regarding the <Module>_Init API

  Problem description:

  Generally a Caveat can never start a requirement. For this all basic Software module shall be checked and cleaned up for this formulation.

  original finding was:

  According to the discussion of RfC# 67678 we detect requirements regarding the initializiation of the certain api's, e.g.:
  "
  ...
  [SWS_EthIf_00213] ?

Caveat: The function requires previous initialization (EthIf_Init). ?()
"
Such requirment are not implementation relevant. Please remove this and add as note.
–Last change on issue 75637 comment 2–

Agreed solution:

General:
- Remove requirements which state that previous initialization of <Modul>_Init() is required, which are formulated like a note, e.g. : [SWS_EthIf_00213] ?
Caveat: The function requires previous initialization (EthIf_Init). ?()
- Add general requirement as note to ch. 8.3 Function definitions
- Remove or reformulate all other requirements, which start with caveat. Most of them are simply a note or implementation hint.

=== EhtIf ===
~8.3 Function definitions
+Note: All functions in this chapter requires previous initialization (EthIf_Init), except the following ones:
EthIf_Init, EthIf_GetVersionInfo

~8.3.2 EthIf_SetControllerMode
-[SWS_EthIf_00038]

~8.3.3 EthIf_GetControllerMode
-[SWS_EthIf_00044]

~8.3.8 EthIf_ CheckWakeup
-[SWS_EthIf_00249]

~8.3.9 EthIf_GetPhysAddr
-[SWS_EthIf_00066]

~8.3.10 EthIf_SetPhysAddr
-[SWS_EthIf_00138]

~8.3.11 EthIf_UpdatePhysAddrFilter
-[SWS_EthIf_00144]

~8.3.12 EthIf_GetPortMacAddr
-[SWS_EthIf_00195]

~8.3.13 EthIf_GetArlTable
-[SWS_EthIf_00201]

8.3.14 EthIf_GetBufferLevel
-[SWS_EthIf_00207]

~8.3.15 EthIf_GetDropCount
-[SWS_EthIf_00213]

~8.3.16 EthIf_StoreConfiguration
-[SWS_EthIf_00218]

~8.3.17 EthIf_ResetConfiguration
-[SWS_EthIf_00223]

8.3.18 EthIf_GetCurrentTime
-[SWS_EthIf_00159]

~8.3.19 EthIf_EnableEgressTimeStamp
-[SWS_EthIf_00165]

~8.3.20 EthIf_GetEgressTimeStamp
-[SWS_EthIf_00171]

~8.3.21 EthIf_GetIngressTimeStamp
-[SWS_EthIf_00177]

~8.3.22 EthIf_SetCorrectionTime
-[SWS_EthIf_00183]

~8.3.23 EthIf_SetGlobalTime
-[SWS_EthIf_00189]

~8.3.24 EthIf_ProvideTxBuffer
-[SWS_EthIf_00074]

~8.3.25 EthIf_Transmit
-[SWS_EthIf_00081]

=== CanIf ===
General note for below items which contain phrases like "The CAN Interface module must be initialized after Power ON": this is not needed anymore cause already

stated in SWS_CANIF_00661. So e.g. in cases where result from CanDrv is just passed through the whole requirement can be deleted.

Also it is assumed, that the mindset like defined in SWS_CANIF_00661 is there for all upper layer functions like <User_TxConfirmation> which might be called by CanIf.

Because if CanDrv would not be already initialised it would return E_NOT_OK which is passed through:
-SWS_CANIF_00312
-SWS_CANIF_00316

Because already guarded by SWS_CANIF_00661:
-SWS_CANIF_00334
-SWS_CANIF_00339
-SWS_CANIF_00344
-SWS_CANIF_00349

~SWS_CANIF_00661
All CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall not execute their normal operation and return E_NOT_OK unless the CanIf has been initialized with a preceding call of CanIf_Init().

+SWS_CANIF_????1
If the switch CANIF_PUBLIC_DEV_ERROR_DETECT is enabled, all CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall report to the DET (using CANIF_E_UNINIT) unless the CanIf has been initialized with a preceding call of CanIf_Init().

-SWS_CANIF_00323 change to
Note: During the call of CanIf_Transmit() the buffer of PduInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00329
CanIf_ReadRxPduData() shall not be used for CanIfRxSduId, which are defined to

receive multiple CAN-Ids (range reception).

Add note below SWS_CANIF_00329
Note: During the call of of CanIf_ReadRxPduData() the buffer of CanIfRxInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00356
CanIf_SetDynamicTxId() shall not be interrupted by CanIf_Transmit(), if the same L-SDU ID is handled.

-SWS_CANIF_00407 change to
Note: The call context of CanIf_CheckValidation() is either on interrupt level (interrupt mode) or on task level (polling mode).
Caveat: The corresponding CAN controller and transceiver must be switched on via CanTrcv_SetOpMode(Transceiver, CANTRCV_TRCVMODE_NORMAL) and Can_SetControllerMode(Controller, CAN_CS_STARTED) and the corresponding mode indications must have been called.

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
-SWS_CANIF_00737 CanIf_GetTxConfirmationState()
-SWS_CANIF_00870 CanIf_SetBaudrate()

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
-SWS_CANIF_00401 CanIf_CheckWakeup()
-SWS_CANIF_00413 CanIf_TxConfirmation()
-SWS_CANIF_00422 CanIf_RxIndication()
-SWS_CANIF_00432 CanIf_ControllerBusOff()
-SWS_CANIF_00818 CanIf_ConfirmPnAvailability()
-SWS_CANIF_00807 CanIf_ClearTrcvWufFlagIndication()
-SWS_CANIF_00811 CanIf_CheckTrcvWakeFlagIndication()
-SWS_CANIF_00703 CanIf_ControllerModeIndication()

-SWS_CANIF_00709 CanIf_TrcvModeIndication()
-SWS_CANIF_00887 <User_TriggerTransmit>()
-SWS_CANIF_00437 <User_TxConfirmation>()

-SWS_CANIF_00440 change to
Note: Until <User_RxIndication>() returns, CanIf will not access PduInfoPtr. The PduInfoPtr is only valid and can be used by upper layers, until the indication returns. CanIf guarantees that the number of configured bytes for this PduInfoPtr is valid.
Note: The call context of <User_RxIndication>() is either on interrupt level (interrupt mode) or on task level (polling mode).

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00822 <User_ConfirmPnAvailability>()
-SWS_CANIF_00793 <User_ClearTrcvWufFlagIndication>()
-SWS_CANIF_00799 <User_CheckTrcvWakeFlagIndication>()

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00688 <User_ControllerModeIndication>()

=== ComM ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (ComM_Init), except the following ones:
ComM_Init, ComM_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the ComM module is initialized correctly.

-[SWS_ComM_00805] Caveats of ComM_Nm_NetworkStartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00806] Caveats of ComM_Nm_NetworkMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00808] Caveats of ComM_Nm_PrepareBusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00810] Caveats of ComM_Nm_BusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00812] Caveats of ComM_Nm_RestartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00814] Caveats of ComM_EcuM_WakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00965] Caveats of ComM_EcuM_PNCWakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00816] Caveats of ComM_BusSM_ModeIndication(): The Communication Manager Module is initialized correctly.()

=== LdCom ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (LdCom_Init), except the following ones:
LdCom_Init, LdCom_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the LdCom module is initialized correctly.
–Last change on issue 75637 comment 23–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.27 Specification Item SWS_EthIf_00096

Trace References:

none

Content:

Caveat: The function shall be callable on interrupt level.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #75637: [diverse] Remove requirements like initializiation requirements regarding the <Module>_Init API

  Problem description:

  Generally a Caveat can never start a requirement. For this all basic Software module shall be checked and cleaned up for this formulation.

  original finding was:

  According to the discussion of RfC# 67678 we detect requirements regarding the initializiation of the certain api's, e.g.:
  "
  ...
  [SWS_EthIf_00213] ?
  Caveat: The function requires previous initialization (EthIf_Init). ?()
  "
  Such requirment are not implementation relevant. Please remove this and add as note.
  –Last change on issue 75637 comment 2–

  Agreed solution:

  General:
  - Remove requirements which state that previous initialization of <Modul>_Init() is required, which are formulated like a note, e.g. : [SWS_EthIf_00213] ?
  Caveat: The function requires previous initialization (EthIf_Init). ?()
  - Add general requirement as note to ch. 8.3 Function definitions
  - Remove or reformulate all other requirements, which start with caveat. Most of them are simply a note or implementation hint.

  === EhtIf ===
  ~8.3 Function definitions
  +Note: All functions in this chapter requires previous initialization (EthIf_Init), except the following ones:
  EthIf_Init, EthIf_GetVersionInfo

  ~8.3.2 EthIf_SetControllerMode
  -[SWS_EthIf_00038]

  ~8.3.3 EthIf_GetControllerMode
  -[SWS_EthIf_00044]

  ~8.3.8 EthIf_ CheckWakeup

-[SWS_EthIf_00249]

~8.3.9 EthIf_GetPhysAddr
-[SWS_EthIf_00066]

~8.3.10 EthIf_SetPhysAddr
-[SWS_EthIf_00138]

~8.3.11 EthIf_UpdatePhysAddrFilter
-[SWS_EthIf_00144]

~8.3.12 EthIf_GetPortMacAddr
-[SWS_EthIf_00195]

~8.3.13 EthIf_GetArlTable
-[SWS_EthIf_00201]

8.3.14 EthIf_GetBufferLevel
-[SWS_EthIf_00207]

~8.3.15 EthIf_GetDropCount
-[SWS_EthIf_00213]

~8.3.16 EthIf_StoreConfiguration
-[SWS_EthIf_00218]

~8.3.17 EthIf_ResetConfiguration
-[SWS_EthIf_00223]

8.3.18 EthIf_GetCurrentTime
-[SWS_EthIf_00159]

~8.3.19 EthIf_EnableEgressTimeStamp
-[SWS_EthIf_00165]

~8.3.20 EthIf_GetEgressTimeStamp
-[SWS_EthIf_00171]

~8.3.21 EthIf_GetIngressTimeStamp
-[SWS_EthIf_00177]

~8.3.22 EthIf_SetCorrectionTime
-[SWS_EthIf_00183]

~8.3.23 EthIf_SetGlobalTime
-[SWS_EthIf_00189]

~8.3.24 EthIf_ProvideTxBuffer
-[SWS_EthIf_00074]

~8.3.25 EthIf_Transmit
-[SWS_EthIf_00081]

=== CanIf ===
General note for below items which contain phrases like "The CAN Interface module must be initialized after Power ON": this is not needed anymore cause already stated in SWS_CANIF_00661. So e.g. in cases where result from CanDrv is just passed through the whole requirement can be deleted.

Also it is assumed, that the mindset like defined in SWS_CANIF_00661 is there for all upper layer functions like <User_TxConfirmation> which might be called by CanIf.

Because if CanDrv would not be already initialised it would return E_NOT_OK which is passed through:
-SWS_CANIF_00312
-SWS_CANIF_00316

Because already guarded by SWS_CANIF_00661:
-SWS_CANIF_00334
-SWS_CANIF_00339
-SWS_CANIF_00344
-SWS_CANIF_00349

~SWS_CANIF_00661
All CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall not execute their normal operation and return E_NOT_OK unless the CanIf has been initialized with a preceding call of CanIf_Init().

+SWS_CANIF_????1

If the switch CANIF_PUBLIC_DEV_ERROR_DETECT is enabled, all CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall report to the DET (using CANIF_E_UNINIT) unless the CanIf has been initialized with a preceding call of CanIf_Init().

-SWS_CANIF_00323 change to
Note: During the call of CanIf_Transmit() the buffer of PduInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00329
CanIf_ReadRxPduData() shall not be used for CanIfRxSduId, which are defined to receive multiple CAN-Ids (range reception).

Add note below SWS_CANIF_00329
Note: During the call of of CanIf_ReadRxPduData() the buffer of CanIfRxInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00356
CanIf_SetDynamicTxId() shall not be interrupted by CanIf_Transmit(), if the same L-SDU ID is handled.

-SWS_CANIF_00407 change to
Note: The call context of CanIf_CheckValidation() is either on interrupt level (interrupt mode) or on task level (polling mode).
Caveat: The corresponding CAN controller and transceiver must be switched on via CanTrcv_SetOpMode(Transceiver, CANTRCV_TRCVMODE_NORMAL) and Can_SetControllerMode(Controller, CAN_CS_STARTED) and the corresponding mode indications must have been called.

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
-SWS_CANIF_00737 CanIf_GetTxConfirmationState()
-SWS_CANIF_00870 CanIf_SetBaudrate()

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
-SWS_CANIF_00401 CanIf_CheckWakeup()
-SWS_CANIF_00413 CanIf_TxConfirmation()
-SWS_CANIF_00422 CanIf_RxIndication()
-SWS_CANIF_00432 CanIf_ControllerBusOff()
-SWS_CANIF_00818 CanIf_ConfirmPnAvailability()
-SWS_CANIF_00807 CanIf_ClearTrcvWufFlagIndication()
-SWS_CANIF_00811 CanIf_CheckTrcvWakeFlagIndication()
-SWS_CANIF_00703 CanIf_ControllerModeIndication()
-SWS_CANIF_00709 CanIf_TrcvModeIndication()
-SWS_CANIF_00887 <User_TriggerTransmit>()
-SWS_CANIF_00437 <User_TxConfirmation>()

-SWS_CANIF_00440 change to
Note: Until <User_RxIndication>() returns, CanIf will not access PduInfoPtr. The PduInfoPtr is only valid and can be used by upper layers, until the indication returns. CanIf guarantees that the number of configured bytes for this PduInfoPtr is valid.
Note: The call context of <User_RxIndication>() is either on interrupt level (interrupt mode) or on task level (polling mode).

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00822 <User_ConfirmPnAvailability>()
-SWS_CANIF_00793 <User_ClearTrcvWufFlagIndication>()
-SWS_CANIF_00799 <User_CheckTrcvWakeFlagIndication>()

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."

"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00688 <User_ControllerModeIndication>()

=== ComM ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (ComM_Init), except the following ones:
ComM_Init, ComM_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the ComM module is initialized correctly.
-[SWS_ComM_00805] Caveats of ComM_Nm_NetworkStartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00806] Caveats of ComM_Nm_NetworkMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00808] Caveats of ComM_Nm_PrepareBusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00810] Caveats of ComM_Nm_BusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00812] Caveats of ComM_Nm_RestartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00814] Caveats of ComM_EcuM_WakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00965] Caveats of ComM_EcuM_PNCWakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00816] Caveats of ComM_BusSM_ModeIndication(): The Communication Manager Module is initialized correctly.()

=== LdCom ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (LdCom_Init), except the following ones:
LdCom_Init, LdCom_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the LdCom module is initialized correctly.
–Last change on issue 75637 comment 23–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.28 Specification Item SWS_EthIf_00097

Trace References:

none

Content:

Service name:	EthIf_MainFunctionRxEthIf_MainFunctionRx
Syntax:	void EthIf_MainFunctionRx(void)
Service ID[hex]:	0x20
Description:	The function checks for new received frames and issues transmission confirmations reception indications in polling mode. It checks also for transceiver state changes.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76647: [EthIf] Findings in EthIf

  Problem description:

  Name: Stellwag
  Phone:
  Role: WP-A review (as part of V2x review)

  Figure 1/Introduction: should be enhanced with the wireless modules. Also the V2x should be mentioned as upper layer.

  Chapter 4.1: What does "The Ethernet Interface module is only able to handle a single thread of execution. The execution must not be pre-empted by itself." mean? Only one "caller" at a time? If the EthIf is using a callback/callout the called function is not allowed to use the EthIf? Please detail/explain.

  Chapter 4.1: "The Ethernet Interface module is only able to handle a single thread of execution. The execution must not be pre-empted by itself." is no limitation - please reword or remove

  Chapter 5/page 14: add V2xGn to the list of users of EthIf; remove the "empty" list

SWS_EthIf_00226: Why is EthIf unconditionally using the EthSwt..Cfg.h file? What is really required? Btw: This dependency is not shown in the diagram in chapter 5.1.1 ...

SWS_EthIf_00261: please replace SWS_EthIf_xxx21 with the real number

Chapter 7.2.1: There is no "Default error" in AUTOSAR. Please replace with "development error"

SWS_EthIf_00149: If we follow SWS_EthIf_00153 then this type is missing in SWS_EthIf_00023 .... but maybe SWS_EthIf_00023 only means those types starting with Eth_ ...

SWS_EthIf_00150: This type is availibe but seems not be used in any API ... please clarify or remove

SWS_EthIf_00039: This function is marked as non-reentrant. If the EthIf has access to more drivers (e.g. Eth and WEth): Who coordinates the users that no parallel calls are done? This problem applies to many APIs.

SWS_EthIf_00190: Does this function only works when a swicth is used? What about cases without a switch but some dedicated controllers? What about wirless access? Please either update or change name to EthIf_GetPortMacAddrFromSwitch()

SWS_EthIf_00214: This API stores "values" "somewhere" - is there any service to "restore" this values? SWS_EthIf_00219 just "resets" the values ...

SWS_EthIf_00095: "Caveat: The function requires previous initialization (EthIf_Init)" is no requirement for me, please remove. There are more requirements starting with "Caveat: ..." ...

SWS_EthIf_00098: This contradicts SWS_BSW_00037, please remove (same applies to SWS_EthIf_00278)

SWS_EthIf_00099: ETHIF_ENABLE_RX_INTERRUPT is not defined in the SWS ...

SWS_EthIf_91051: What is the difference to SWS_EthIf_00097? Why do we need N receiever main functions? Where is the period configured? (it's not in ECUC_EthIf_00023...)

SWS_EthIf_00100: ETHIF_ENABLE_TX_INTERRUPT is not defined in the

SWS ...

Chapter 8.5: Why are different tables are used for scheduled functions? This just confuses readers, I suggest to use the form from the SWS template.

SWS_EthIf_00102: This table is empty. Either add functions required by the EthIf or remove table.

SWS_EthIf_91009: I think this is not an "Asynchronous" function but a synchronouse one.

Agreed solution:

In chapter 1:
- change "This indicates the main task of the Ethernet Interface:
Provide to upper layers a hardware independent interface to the Ethernet Communication System comprising multiple different Ethernet controllers and transceivers. This interface shall be uniform for all Ethernet controllers and transceivers. Thus, the upper layers (TCP/IP, EthSM, CDD) may access the underlying bus system in a uniform manner."
to
"This indicates the main task of the Ethernet Interface:
Provide to upper layers a hardware independent interface to the Ethernet Communication System comprising multiple different wired or wireless Ethernet controllers and transceivers. This interface shall be uniform for all Ethernet controllers and transceivers. Thus, the upper layers (TCP/IP, EthSM, CDD, V2x modules) may access the underlying bus system in a uniform manner."
- Add a upper layer "V2x modules" to figure 1
- Add WEth and WEthTrcv as lower layer modules to figure 1

replace
SWS_EthIf_00114
The function shall change the state of the component from ETHIF_STATE_UNINIT to ETHIF_STATE_INIT
by
SWS_EthIf_00114
The function shall change the state of the component from uninitialized to initialized.

Chapter 4.1:
- remove "The Ethernet Interface module is only able to handle a single thread of execution. The execution must not be pre-empted by itself."

Chapter 5:

- add V2xGn to the list of users of EthIf; remove the "empty" list of "Modules used by the Ethernet Interface module:"

SWS_EthIf_00150:
- remove SWS_EthIf_00150

~SWS_EthIf_00153 from:
The types specified in SWS_EthernetInterface shall be declared in Eth_GeneralTypes.h.()
to
General Eth types specified in SWS_EthernetInterface shall be declared in Eth_GeneralTypes.h.()

8.3.35 EthIf_GetBufWTxParams
replace SWS-ID by new ID: SWS_EthIf_91009 -> SWS_EthIf_xxxxx

~ SWS_EthIf_00097
remove from Description: " It checks also for transceiver state changes."
replace from Description: "transmission confirmations" by "reception indications"

~SWS_EthIf_00099
replace ETHIF_ENABLE_RX_INTERRUPT by EthIfEnableRxInterrupt

~SWS_EthIf_00100
replace ETHIF_ENABLE_TX_INTERRUPT by EthIfEnableTxInterrupt

remove SWS_EthIf_00102

Please remove the requirement id and convert it to a note:
SWS_EthIf_00027, SWS_EthIf_00038, SWS_EthIf_00044, SWS_EthIf_00066, SWS_EthIf_00138, SWS_EthIf_00144, SWS_EthIf_00195, SWS_EthIf_00201, SWS_EthIf_00207, SWS_EthIf_00218, SWS_EthIf_00223, SWS_EthIf_00159, SWS_EthIf_00165, SWS_EthIf_00171, SWS_EthIf_00177, SWS_EthIf_00074, SWS_EthIf_00089, SWS_EthIf_00095

Please remove the requirement id and maybe keep it as a note:
SWS_EthIf_00249, SWS_EthIf_00284, SWS_EthIf_00081, SWS_EthIf_00090, SWS_EthIf_00096, SWS_EthIf_00292.
–Last change on issue 76647 comment 37–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.29 Specification Item SWS_EthIf_00098

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZEDUNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification "belongs".

  *** BSW UML Model ***
  SWS_CanNm:
  ———-

  Chapter 8.6.1 Optional Interfaces:
  Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

SWS_LinIf:
———-
SWS_LinIf_00359: add Det_ReportRuntimeError

SWS_UdpNm:
———-
Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.30 Specification Item SWS_EthIf_00103

Trace References:

none

Content:

API function	Description
BswM_EthIf_PortGroupLinkStateChg	Function called by EthIf to indicate the link state change of a certain Ethernet switch port group.
Eth_GetControllerMode	Obtains the state of the indexed controller
Eth_GetPhysAddr	Obtains the physical source address used by the indexed controller
Eth_ProvideTxBuffer	Provides access to a transmit buffer of the FIFO related to the specified priority
Eth_ReadMii	Reads a transceiver register
Eth_Receive	Receive a frame from the related fifo.
Eth_SetControllerMode	Enables / disables the indexed controller
Eth_Transmit	Triggers transmission of a previously filled transmit buffer
Eth_TxConfirmation	Triggers frame transmission confirmation
Eth_WriteMii	Configures a transceiver register or triggers a function offered by the receiver
EthSM_CtrlModeIndication	Called when mode has been read out. Either triggered by previous EthIf_GetControllerMode or by EthIf_SetController Mode call. Can directly be called within the trigger functions.

API function	Description
EthSwt_EnableTimeStamping	Activates egress time stamping on a dedicated message object on all ports of a Switch but on uplink ports between cascaded switcheswhere EthSwtPortTimeStampSupport is set to TRUE. The selective activation of dedicated message objects for time stamping reduces the number of notification calls only to the required calls. Some HW does store once the egress time stamp marker and some HW needs it always before transmission. There will be no disabled functionality, due to the fact, that the message type is always "time stamped" by network design.
EthSwt_SetMgmtInfo	Extends the Ethernet frame prepared previously by EthSwt_EthTxPrepareFrame() with the management information to achieve transmission only on specific ports.
EthTrcv_GetBaudRate	Obtains the baud rate of the indexed transceiver
EthTrcv_GetDuplexMode	Obtains the duplex mode of the indexed transceiver
EthTrcv_GetLinkState	Obtains the link state of the indexed transceiver
EthTrcv_GetTransceiverMode	Obtains the state of the indexed transceiver
EthTrcv_SetTransceiverMode	Enables / disables the indexed transceiver
EthTrcv_StartAutoNegotiation	Restarts the negotiation of the transmission parameters used by the indexed transceiver
WEth_GetBufWRxParams	Read out values related to the receive direction for a received packet. For example, this could be RSSI or Channel belonging to one single packet.This API is valid only within the context of WEth_Receive
WEth_GetBufWTxParams	Read out values related to the transmit direction for a transmitted packet. For example, this could be transaction ID belonging to one single packet.This API is valid only within the context of WEth_TxConfirmation.
WEth_SetBufWTxParams	Set values related to the transmit direction for a specific buffer (packet to be sent). For example, this can be the desired transmit power or the channel belonging to one single packet.
WEthTrcv_GetChanRxParams	Read values related to the receive direction of the transceiver. For example, this could be a Channel Busy Ratio (CBR) or the average Channel Idle Time (CIT).
WEthTrcv_SetChanRxParams	Set values related to the receive direction of a transceiver's wireless channel.For example, this could be a channel parameter like the frequency.
WEthTrcv_SetChanTxParams	Set values related to the transmit direction of a transceiver's wireless channel. For example, this could be the bitrate of a channel.
WEthTrcv_SetRadioParams	Set values related to a transceiver's wireless radio. For example, this could be the selection of the radio settings (channel, ...).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76196: [EthSwt] Sequence charts needed to explain Init, Communication Initialization and Link State Change

  Problem description:

The Sequence charts for Initialization of the module, Communication Initialization and passing of the Link State are important to understand the interaction between Eth, EthTrcv, EthSwt and EthIf and need to be made available.

Agreed solution:

- delete figure 5 Module Initialization Sequence diagram as the sequence of this initialization is dependent on HW and straight forward diagrams imply a easy implementation rule which is simply not true and will never work. Initialization is triggered by EcuM (partly via BsWM) and not part of EthIf or EthSwt-specification.

[SWS_EthIf_00103]
add BswM_EthIf_PortGroupLinkStateChg to expected optional interfaces table and description for BswM_EthIf_PortGroupLinkStateChg:
Indicates the link state change of a certain Ethernet switch port group.

For Communication Initialization of one single Switch Port and of a PortGroup and ProtGroups with PNC is attached to RfC # 76196. Same applies for GetLinkState of one Switch Port.
all attached diagrams shall be added to chapter 9 Sequence diagrams as separate subchapter with the name of the attachments.
All diagrams and the BSW-UML diagram used are in https://svn.autosar.org/repos/work/09_WorkPackages/WP-A2/70_InternalDocuments/EthSwt-Discussions/Sequence20diagrams20and20inittialization/
–Last change on issue 76196 comment 45–

BW-C-Level:

Application	Specification	Bus
1	1	1

- RfC #77050: [EthSwt] Uplink ports shall not be excluded from timestamping

Problem description:

[SWS_EthSwt_00244] currently limits timestamping to
"all ... ports except the ports where EthSwtPortRole is set to ETHSWT_UP_LINK_PORT".

This limitation is problematic for cascaded switches since
no timestamps are taken when SYNC messages exit the first
switch and enter the second one.

It may be possible to omit timestamping on the link
between the cascaded switches if the timestamp counters in both switches are synchronized.

It is then possible to use the ingress timestamp on the first switch in combination with any egress timestamp on the second switch.

However, without a synchronization of the timestamp counters in both switches it is necessary to consider ingress and egress (on the uplink port) timestamps on the first switch and ingress (on the uplink port) and egress timestamps on the second switch.

Since it is unclear if all switch devices support timestamp counter synchronization and since this feature may result in additional hardware requirements (e.g., common clock source, reset lines etc.) it may be a show-stopper to omit timestamping at the uplink ports.

I recommend to either just remove this restriction or to make it configurable in case that the hardware supports synchronized counters.

Agreed solution:

~[SWS_EthSwt_00244][If EthSwt_EnableTimeStamp is called for a SwitchIdx, the switch driver shall enable the time-stamping for all his ports where EthSwtPortTimeStampSupport is set to TRUE.]

~[SWS_EthSwt_00378] If EthSwt_PortEnableTimeStamp is called for a PortIdx, the switch driver shall enable the time-stamping for this port if EthSwtPortTimeStampSupport is set to TRUE for this port.

~SWS_EthSwt_91011 EthSwt_EnableTimeStamping
Description: Activates egress time stamping on a dedicated message object on all ports of a Switch where EthSwtPortTimeStampSupport is set to TRUE. The selective activation of dedicated message objects for time stamping reduces the number of notification calls only to the required calls. Some HW does store once the egress time stamp marker and some HW needs it always before transmission. There will be no disabled functionality, due to the fact, that the message type is always "time stamped" by network design.

~SWS_EthSwt_91028 EthSwt_PortEnableTimeStamp
Description: Activates egress time stamping on a dedicated message object on a dedicated port of a Switch if EthSwtPortTimeStampSupport is set to TRUE for this port. The selective activation of dedicated message objects for time stamping reduces the number of notification calls only to the required calls. Some HW does store once the egress time stamp marker and some HW needs it always before transmission. There will be no disabled functionality, due to the fact, that the message type is always "time stamped" by network design.
–Last change on issue 77050 comment 9–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.31 Specification Item SWS_EthIf_00104

Trace References:

none

Content:

Service name:	<User>_RxIndicationUser_RxIndication	
Syntax:	void <User>_RxIndication(uint8 CtrlIdx, Eth_FrameType FrameType, boolean IsBroadcast, const uint8* PhysAddrPtr, const uint8* DataPtr, uint16 LenByte)	
Service ID[hex]:	–	
Sync/Async:	–	
Reentrancy:	Dont care	
Parameters (in):	CtrlIdxUser_RxIndication.CtrlIdx	Index of the Ethernet controller within the context of the Ethernet Interface
	FrameTypeUser_RxIndication.FrameType	frame type of received Ethernet frame
	IsBroadcastUser_RxIndication.IsBroadcast	parameter to indicate a broadcast frame
	PhysAddrPtrUser_RxIndication.PhysAddrPtr	pointer to Physical source address (MAC address in network byte order) of received Ethernet frame
	DataPtrUser_RxIndication.DataPtr	Pointer to payload of the received Ethernet frame (i.e. Ethernet header is not provided).
	LenByteUser_RxIndication.LenByte	Length of received data.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	Indicates the reception of an Ethernet frame	

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* Authenticated, uint8* AuthenticationReqData, uint8* AuthenticationResData)

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

1.32 Specification Item SWS_EthIf_00124

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZEDUNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of ’Runtime errors’

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

Notes:
- It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
- The review task of the ITs shall be done by the WP to which the specification "belongs".

*** BSW UML Model ***
SWS_CanNm:
———-
Chapter 8.6.1 Optional Interfaces:
Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

SWS_LinIf:
———-
SWS_LinIf_00359: add Det_ReportRuntimeError

SWS_UdpNm:
———-
Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.33 Specification Item SWS_EthIf_00135

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZEDUNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification "belongs".

  *** BSW UML Model ***
  SWS_CanNm:
  ———-
  Chapter 8.6.1 Optional Interfaces:
  Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

  SWS_LinIf:
  ———-
  SWS_LinIf_00359: add Det_ReportRuntimeError

  SWS_UdpNm:
  ———-
  Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

  *** ECUC XML ***

Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.34 Specification Item SWS_EthIf_00138

Trace References:

none

Content:

Caveat: The function requires previous initialization (EthIf_Init).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #75637: [diverse] Remove requirements like initializiation requirements regarding the <Module>_Init API

  Problem description:

  Generally a Caveat can never start a requirement. For this all basic Software module shall be checked and cleaned up for this formulation.

  original finding was:

  According to the discussion of RfC# 67678 we detect requirements regarding the initializiation of the certain api's, e.g.:
  "
  ...
  [SWS_EthIf_00213] ?
  Caveat: The function requires previous initialization (EthIf_Init). ?()
  "
  Such requirment are not implementation relevant. Please remove this and add as note.
  –Last change on issue 75637 comment 2–

  Agreed solution:

  General:
  - Remove requirements which state that previous initialization of <Modul>_Init() is

required, which are formulated like a note, e.g. : [SWS_EthIf_00213] ?
Caveat: The function requires previous initialization (EthIf_Init). ?()
- Add general requirement as note to ch. 8.3 Function definitions
- Remove or reformulate all other requirements, which start with caveat. Most of them are simply a note or implementation hint.

=== EhtIf ===
~8.3 Function definitions
+Note: All functions in this chapter requires previous initialization (EthIf_Init), except the following ones:
EthIf_Init, EthIf_GetVersionInfo

~8.3.2 EthIf_SetControllerMode
-[SWS_EthIf_00038]

~8.3.3 EthIf_GetControllerMode
-[SWS_EthIf_00044]

~8.3.8 EthIf_ CheckWakeup
-[SWS_EthIf_00249]

~8.3.9 EthIf_GetPhysAddr
-[SWS_EthIf_00066]

~8.3.10 EthIf_SetPhysAddr
-[SWS_EthIf_00138]

~8.3.11 EthIf_UpdatePhysAddrFilter
-[SWS_EthIf_00144]

~8.3.12 EthIf_GetPortMacAddr
-[SWS_EthIf_00195]

~8.3.13 EthIf_GetArlTable
-[SWS_EthIf_00201]

8.3.14 EthIf_GetBufferLevel
-[SWS_EthIf_00207]

~8.3.15 EthIf_GetDropCount
-[SWS_EthIf_00213]

~8.3.16 EthIf_StoreConfiguration
-[SWS_EthIf_00218]

~8.3.17 EthIf_ResetConfiguration
-[SWS_EthIf_00223]

8.3.18 EthIf_GetCurrentTime
-[SWS_EthIf_00159]

~8.3.19 EthIf_EnableEgressTimeStamp
-[SWS_EthIf_00165]

~8.3.20 EthIf_GetEgressTimeStamp
-[SWS_EthIf_00171]

~8.3.21 EthIf_GetIngressTimeStamp
-[SWS_EthIf_00177]

~8.3.22 EthIf_SetCorrectionTime
-[SWS_EthIf_00183]

~8.3.23 EthIf_SetGlobalTime
-[SWS_EthIf_00189]

~8.3.24 EthIf_ProvideTxBuffer
-[SWS_EthIf_00074]

~8.3.25 EthIf_Transmit
-[SWS_EthIf_00081]

=== CanIf ===
General note for below items which contain phrases like "The CAN Interface module must be initialized after Power ON": this is not needed anymore cause already stated in SWS_CANIF_00661. So e.g. in cases where result from CanDrv is just passed through the whole requirement can be deleted.

Also it is assumed, that the mindset like defined in SWS_CANIF_00661 is there for all upper layer functions like <User_TxConfirmation> which might be called by CanIf.

Because if CanDrv would not be already initialised it would return E_NOT_OK which is passed through:
-SWS_CANIF_00312
-SWS_CANIF_00316

Because already guarded by SWS_CANIF_00661:
-SWS_CANIF_00334
-SWS_CANIF_00339
-SWS_CANIF_00344
-SWS_CANIF_00349

~SWS_CANIF_00661
All CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall not execute their normal operation and return E_NOT_OK unless the CanIf has been initialized with a preceding call of CanIf_Init().

+SWS_CANIF_????1
If the switch CANIF_PUBLIC_DEV_ERROR_DETECT is enabled, all CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall report to the DET (using CANIF_E_UNINIT) unless the CanIf has been initialized with a preceding call of CanIf_Init().

-SWS_CANIF_00323 change to
Note: During the call of CanIf_Transmit() the buffer of PduInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00329
CanIf_ReadRxPduData() shall not be used for CanIfRxSduId, which are defined to receive multiple CAN-Ids (range reception).

Add note below SWS_CANIF_00329
Note: During the call of of CanIf_ReadRxPduData() the buffer of CanIfRxInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00356
CanIf_SetDynamicTxId() shall not be interrupted by CanIf_Transmit(), if the same L-SDU ID is handled.

-SWS_CANIF_00407 change to
Note: The call context of CanIf_CheckValidation() is either on interrupt level (interrupt mode) or on task level (polling mode).
Caveat: The corresponding CAN controller and transceiver must be switched on via CanTrcv_SetOpMode(Transceiver, CANTRCV_TRCVMODE_NORMAL) and Can_SetControllerMode(Controller, CAN_CS_STARTED) and the corresponding mode indications must have been called.

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
-SWS_CANIF_00737 CanIf_GetTxConfirmationState()
-SWS_CANIF_00870 CanIf_SetBaudrate()

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
-SWS_CANIF_00401 CanIf_CheckWakeup()
-SWS_CANIF_00413 CanIf_TxConfirmation()
-SWS_CANIF_00422 CanIf_RxIndication()
-SWS_CANIF_00432 CanIf_ControllerBusOff()
-SWS_CANIF_00818 CanIf_ConfirmPnAvailability()
-SWS_CANIF_00807 CanIf_ClearTrcvWufFlagIndication()
-SWS_CANIF_00811 CanIf_CheckTrcvWakeFlagIndication()
-SWS_CANIF_00703 CanIf_ControllerModeIndication()
-SWS_CANIF_00709 CanIf_TrcvModeIndication()
-SWS_CANIF_00887 <User_TriggerTransmit>()
-SWS_CANIF_00437 <User_TxConfirmation>()

-SWS_CANIF_00440 change to
Note: Until <User_RxIndication>() returns, CanIf will not access PduInfoPtr. The PduInfoPtr is only valid and can be used by upper layers, until the indication returns. CanIf guarantees that the number of configured bytes for this PduInfoPtr is valid.

Note: The call context of <User_RxIndication>() is either on interrupt level (interrupt mode) or on task level (polling mode).

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00822 <User_ConfirmPnAvailability>()
-SWS_CANIF_00793 <User_ClearTrcvWufFlagIndication>()
-SWS_CANIF_00799 <User_CheckTrcvWakeFlagIndication>()

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00688 <User_ControllerModeIndication>()

=== ComM ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (ComM_Init), except the following ones:
ComM_Init, ComM_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the ComM module is initialized correctly.
-[SWS_ComM_00805] Caveats of ComM_Nm_NetworkStartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00806] Caveats of ComM_Nm_NetworkMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00808] Caveats of ComM_Nm_PrepareBusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00810] Caveats of ComM_Nm_BusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00812] Caveats of ComM_Nm_RestartIndication: The ComM

module is initialized correctly.()
-[SWS_ComM_00814] Caveats of ComM_EcuM_WakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00965] Caveats of ComM_EcuM_PNCWakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00816] Caveats of ComM_BusSM_ModeIndication(): The Communication Manager Module is initialized correctly.()

=== LdCom ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (LdCom_Init), except the following ones:
LdCom_Init, LdCom_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the LdCom module is initialized correctly.
–Last change on issue 75637 comment 23–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.35 Specification Item SWS_EthIf_00141

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZEDUNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

Agreed solution:

solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

Notes:
- It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
- The review task of the ITs shall be done by the WP to which the specification "belongs".

*** BSW UML Model ***
SWS_CanNm:
———-
Chapter 8.6.1 Optional Interfaces:
Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

SWS_LinIf:
———-
SWS_LinIf_00359: add Det_ReportRuntimeError

SWS_UdpNm:
———-
Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.36 Specification Item SWS_EthIf_00144

Trace References:

none

Content:

Caveat: The function requires previous initialization (EthIf_Init).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #75637: [diverse] Remove requirements like initializiation requirements regarding the <Module>_Init API

  Problem description:

  Generally a Caveat can never start a requirement. For this all basic Software module shall be checked and cleaned up for this formulation.

  original finding was:

  According to the discussion of RfC# 67678 we detect requirements regarding the initializiation of the certain api's, e.g.:
  "
  ...
  [SWS_EthIf_00213] ?
  Caveat: The function requires previous initialization (EthIf_Init). ?()
  "
  Such requirment are not implementation relevant. Please remove this and add as note.
  –Last change on issue 75637 comment 2–

  Agreed solution:

  General:
  - Remove requirements which state that previous initialization of <Modul>_Init() is required, which are formulated like a note, e.g. : [SWS_EthIf_00213] ?
  Caveat: The function requires previous initialization (EthIf_Init). ?()
  - Add general requirement as note to ch. 8.3 Function definitions
  - Remove or reformulate all other requirements, which start with caveat. Most of them are simply a note or implementation hint.

  === EhtIf ===
  ~8.3 Function definitions
  +Note: All functions in this chapter requires previous initialization (EthIf_Init), except the following ones:
  EthIf_Init, EthIf_GetVersionInfo

~8.3.2 EthIf_SetControllerMode
-[SWS_EthIf_00038]

~8.3.3 EthIf_GetControllerMode
-[SWS_EthIf_00044]

~8.3.8 EthIf_ CheckWakeup
-[SWS_EthIf_00249]

~8.3.9 EthIf_GetPhysAddr
-[SWS_EthIf_00066]

~8.3.10 EthIf_SetPhysAddr
-[SWS_EthIf_00138]

~8.3.11 EthIf_UpdatePhysAddrFilter
-[SWS_EthIf_00144]

~8.3.12 EthIf_GetPortMacAddr
-[SWS_EthIf_00195]

~8.3.13 EthIf_GetArlTable
-[SWS_EthIf_00201]

8.3.14 EthIf_GetBufferLevel
-[SWS_EthIf_00207]

~8.3.15 EthIf_GetDropCount
-[SWS_EthIf_00213]

~8.3.16 EthIf_StoreConfiguration
-[SWS_EthIf_00218]

~8.3.17 EthIf_ResetConfiguration
-[SWS_EthIf_00223]

8.3.18 EthIf_GetCurrentTime
-[SWS_EthIf_00159]

~8.3.19 EthIf_EnableEgressTimeStamp
-[SWS_EthIf_00165]

~8.3.20 EthIf_GetEgressTimeStamp

-[SWS_EthIf_00171]

~8.3.21 EthIf_GetIngressTimeStamp
-[SWS_EthIf_00177]

~8.3.22 EthIf_SetCorrectionTime
-[SWS_EthIf_00183]

~8.3.23 EthIf_SetGlobalTime
-[SWS_EthIf_00189]

~8.3.24 EthIf_ProvideTxBuffer
-[SWS_EthIf_00074]

~8.3.25 EthIf_Transmit
-[SWS_EthIf_00081]

=== CanIf ===
General note for below items which contain phrases like "The CAN Interface module must be initialized after Power ON": this is not needed anymore cause already stated in SWS_CANIF_00661. So e.g. in cases where result from CanDrv is just passed through the whole requirement can be deleted.

Also it is assumed, that the mindset like defined in SWS_CANIF_00661 is there for all upper layer functions like <User_TxConfirmation> which might be called by CanIf.

Because if CanDrv would not be already initialised it would return E_NOT_OK which is passed through:
-SWS_CANIF_00312
-SWS_CANIF_00316

Because already guarded by SWS_CANIF_00661:
-SWS_CANIF_00334
-SWS_CANIF_00339
-SWS_CANIF_00344
-SWS_CANIF_00349

~SWS_CANIF_00661
All CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall not execute their normal operation and return E_NOT_OK unless the CanIf has been initialized with a preceding call of CanIf_Init().

+SWS_CANIF_????1
If the switch CANIF_PUBLIC_DEV_ERROR_DETECT is enabled, all CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall report to the DET (using CANIF_E_UNINIT) unless the CanIf has been initialized with a preceding call of CanIf_Init().

-SWS_CANIF_00323 change to
Note: During the call of CanIf_Transmit() the buffer of PduInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00329
CanIf_ReadRxPduData() shall not be used for CanIfRxSduId, which are defined to receive multiple CAN-Ids (range reception).

Add note below SWS_CANIF_00329
Note: During the call of of CanIf_ReadRxPduData() the buffer of CanIfRxInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00356
CanIf_SetDynamicTxId() shall not be interrupted by CanIf_Transmit(), if the same L-SDU ID is handled.

-SWS_CANIF_00407 change to
Note: The call context of CanIf_CheckValidation() is either on interrupt level (interrupt mode) or on task level (polling mode).
Caveat: The corresponding CAN controller and transceiver must be switched on via CanTrcv_SetOpMode(Transceiver, CANTRCV_TRCVMODE_NORMAL) and Can_SetControllerMode(Controller, CAN_CS_STARTED) and the corresponding mode indications must have been called.

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
-SWS_CANIF_00737 CanIf_GetTxConfirmationState()
-SWS_CANIF_00870 CanIf_SetBaudrate()

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
-SWS_CANIF_00401 CanIf_CheckWakeup()
-SWS_CANIF_00413 CanIf_TxConfirmation()
-SWS_CANIF_00422 CanIf_RxIndication()
-SWS_CANIF_00432 CanIf_ControllerBusOff()
-SWS_CANIF_00818 CanIf_ConfirmPnAvailability()
-SWS_CANIF_00807 CanIf_ClearTrcvWufFlagIndication()
-SWS_CANIF_00811 CanIf_CheckTrcvWakeFlagIndication()
-SWS_CANIF_00703 CanIf_ControllerModeIndication()
-SWS_CANIF_00709 CanIf_TrcvModeIndication()
-SWS_CANIF_00887 <User_TriggerTransmit>()
-SWS_CANIF_00437 <User_TxConfirmation>()

-SWS_CANIF_00440 change to
Note: Until <User_RxIndication>() returns, CanIf will not access PduInfoPtr. The PduInfoPtr is only valid and can be used by upper layers, until the indication returns. CanIf guarantees that the number of configured bytes for this PduInfoPtr is valid.
Note: The call context of <User_RxIndication>() is either on interrupt level (interrupt mode) or on task level (polling mode).

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00822 <User_ConfirmPnAvailability>()

-SWS_CANIF_00793 <User_ClearTrcvWufFlagIndication>()
-SWS_CANIF_00799 <User_CheckTrcvWakeFlagIndication>()

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00688 <User_ControllerModeIndication>()

=== ComM ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (ComM_Init), except the following ones:
ComM_Init, ComM_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the ComM module is initialized correctly.
-[SWS_ComM_00805] Caveats of ComM_Nm_NetworkStartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00806] Caveats of ComM_Nm_NetworkMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00808] Caveats of ComM_Nm_PrepareBusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00810] Caveats of ComM_Nm_BusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00812] Caveats of ComM_Nm_RestartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00814] Caveats of ComM_EcuM_WakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00965] Caveats of ComM_EcuM_PNCWakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00816] Caveats of ComM_BusSM_ModeIndication(): The Communication Manager Module is initialized correctly.()

=== LdCom ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (LdCom_Init), except the following ones:
LdCom_Init, LdCom_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the LdCom module is initialized correctly.
–Last change on issue 75637 comment 23–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.37 Specification Item SWS_EthIf_00155

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZEDUNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of ’Runtime errors’

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification "belongs".

  *** BSW UML Model ***

SWS_CanNm:
———-
Chapter 8.6.1 Optional Interfaces:
Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

SWS_LinIf:
———-
SWS_LinIf_00359: add Det_ReportRuntimeError

SWS_UdpNm:
———-
Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.38 Specification Item SWS_EthIf_00159

Trace References:

none

Content:

Caveat: The function requires previous initialization (EthIf_Init).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #75637: [diverse] Remove requirements like initializiation requirements regarding the <Module>_Init API

  Problem description:

  Generally a Caveat can never start a requirement. For this all basic Software module shall be checked and cleaned up for this formulation.

original finding was:

According to the discussion of RfC# 67678 we detect requirements regarding the initializiation of the certain api's, e.g.:
"
...
[SWS_EthIf_00213] ?
Caveat: The function requires previous initialization (EthIf_Init). ?()
"
Such requirment are not implementation relevant. Please remove this and add as note.
–Last change on issue 75637 comment 2–

Agreed solution:

General:
- Remove requirements which state that previous initialization of <Modul>_Init() is required, which are formulated like a note, e.g. : [SWS_EthIf_00213] ?
Caveat: The function requires previous initialization (EthIf_Init). ?()
- Add general requirement as note to ch. 8.3 Function definitions
- Remove or reformulate all other requirements, which start with caveat. Most of them are simply a note or implementation hint.

=== EhtIf ===
~8.3 Function definitions
+Note: All functions in this chapter requires previous initialization (EthIf_Init), except the following ones:
EthIf_Init, EthIf_GetVersionInfo

~8.3.2 EthIf_SetControllerMode
-[SWS_EthIf_00038]

~8.3.3 EthIf_GetControllerMode
-[SWS_EthIf_00044]

~8.3.8 EthIf_ CheckWakeup
-[SWS_EthIf_00249]

~8.3.9 EthIf_GetPhysAddr
-[SWS_EthIf_00066]

~8.3.10 EthIf_SetPhysAddr
-[SWS_EthIf_00138]

~8.3.11 EthIf_UpdatePhysAddrFilter
-[SWS_EthIf_00144]

~8.3.12 EthIf_GetPortMacAddr
-[SWS_EthIf_00195]

~8.3.13 EthIf_GetArlTable
-[SWS_EthIf_00201]

8.3.14 EthIf_GetBufferLevel
-[SWS_EthIf_00207]

~8.3.15 EthIf_GetDropCount
-[SWS_EthIf_00213]

~8.3.16 EthIf_StoreConfiguration
-[SWS_EthIf_00218]

~8.3.17 EthIf_ResetConfiguration
-[SWS_EthIf_00223]

8.3.18 EthIf_GetCurrentTime
-[SWS_EthIf_00159]

~8.3.19 EthIf_EnableEgressTimeStamp
-[SWS_EthIf_00165]

~8.3.20 EthIf_GetEgressTimeStamp
-[SWS_EthIf_00171]

~8.3.21 EthIf_GetIngressTimeStamp
-[SWS_EthIf_00177]

~8.3.22 EthIf_SetCorrectionTime
-[SWS_EthIf_00183]

~8.3.23 EthIf_SetGlobalTime
-[SWS_EthIf_00189]

~8.3.24 EthIf_ProvideTxBuffer
-[SWS_EthIf_00074]

~8.3.25 EthIf_Transmit
-[SWS_EthIf_00081]

=== CanIf ===
General note for below items which contain phrases like "The CAN Interface module must be initialized after Power ON": this is not needed anymore cause already stated in SWS_CANIF_00661. So e.g. in cases where result from CanDrv is just passed through the whole requirement can be deleted.

Also it is assumed, that the mindset like defined in SWS_CANIF_00661 is there for all upper layer functions like <User_TxConfirmation> which might be called by CanIf.

Because if CanDrv would not be already initialised it would return E_NOT_OK which is passed through:
-SWS_CANIF_00312
-SWS_CANIF_00316

Because already guarded by SWS_CANIF_00661:
-SWS_CANIF_00334
-SWS_CANIF_00339
-SWS_CANIF_00344
-SWS_CANIF_00349

~SWS_CANIF_00661
All CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall not execute their normal operation and return E_NOT_OK unless the CanIf has been initialized with a preceding call of CanIf_Init().

+SWS_CANIF_????1
If the switch CANIF_PUBLIC_DEV_ERROR_DETECT is enabled, all CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall report to the DET (using CANIF_E_UNINIT) unless the CanIf has been initialized with a preceding call of CanIf_Init().

-SWS_CANIF_00323 change to

Note: During the call of CanIf_Transmit() the buffer of PduInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00329
CanIf_ReadRxPduData() shall not be used for CanIfRxSduId, which are defined to receive multiple CAN-Ids (range reception).

Add note below SWS_CANIF_00329
Note: During the call of of CanIf_ReadRxPduData() the buffer of CanIfRxInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00356
CanIf_SetDynamicTxId() shall not be interrupted by CanIf_Transmit(), if the same L-SDU ID is handled.

-SWS_CANIF_00407 change to
Note: The call context of CanIf_CheckValidation() is either on interrupt level (interrupt mode) or on task level (polling mode).
Caveat: The corresponding CAN controller and transceiver must be switched on via CanTrcv_SetOpMode(Transceiver, CANTRCV_TRCVMODE_NORMAL) and Can_SetControllerMode(Controller, CAN_CS_STARTED) and the corresponding mode indications must have been called.

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
-SWS_CANIF_00737 CanIf_GetTxConfirmationState()
-SWS_CANIF_00870 CanIf_SetBaudrate()

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
-SWS_CANIF_00401 CanIf_CheckWakeup()

-SWS_CANIF_00413 CanIf_TxConfirmation()
-SWS_CANIF_00422 CanIf_RxIndication()
-SWS_CANIF_00432 CanIf_ControllerBusOff()
-SWS_CANIF_00818 CanIf_ConfirmPnAvailability()
-SWS_CANIF_00807 CanIf_ClearTrcvWufFlagIndication()
-SWS_CANIF_00811 CanIf_CheckTrcvWakeFlagIndication()
-SWS_CANIF_00703 CanIf_ControllerModeIndication()
-SWS_CANIF_00709 CanIf_TrcvModeIndication()
-SWS_CANIF_00887 <User_TriggerTransmit>()
-SWS_CANIF_00437 <User_TxConfirmation>()

-SWS_CANIF_00440 change to
Note: Until <User_RxIndication>() returns, CanIf will not access PduInfoPtr. The PduInfoPtr is only valid and can be used by upper layers, until the indication returns. CanIf guarantees that the number of configured bytes for this PduInfoPtr is valid.
Note: The call context of <User_RxIndication>() is either on interrupt level (interrupt mode) or on task level (polling mode).

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00822 <User_ConfirmPnAvailability>()
-SWS_CANIF_00793 <User_ClearTrcvWufFlagIndication>()
-SWS_CANIF_00799 <User_CheckTrcvWakeFlagIndication>()

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00688 <User_ControllerModeIndication>()

=== ComM ===
~ ch. 8.3 Function definition

+Note: All functions in this chapter requires previous initialization (ComM_Init), except the following ones:
ComM_Init, ComM_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the ComM module is initialized correctly.
-[SWS_ComM_00805] Caveats of ComM_Nm_NetworkStartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00806] Caveats of ComM_Nm_NetworkMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00808] Caveats of ComM_Nm_PrepareBusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00810] Caveats of ComM_Nm_BusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00812] Caveats of ComM_Nm_RestartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00814] Caveats of ComM_EcuM_WakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00965] Caveats of ComM_EcuM_PNCWakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00816] Caveats of ComM_BusSM_ModeIndication(): The Communication Manager Module is initialized correctly.()

=== LdCom ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (LdCom_Init), except the following ones:
LdCom_Init, LdCom_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the LdCom module is initialized correctly.
–Last change on issue 75637 comment 23–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.39 Specification Item SWS_EthIf_00161

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service Eth If_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZEDUNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification "belongs".

  *** BSW UML Model ***
  SWS_CanNm:
  ———-
  Chapter 8.6.1 Optional Interfaces:
  Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

  SWS_LinIf:
  ———-
  SWS_LinIf_00359: add Det_ReportRuntimeError

  SWS_UdpNm:
  ———-

Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.40 Specification Item SWS_EthIf_00165

Trace References:

none

Content:

Caveat: The function requires previous initialization (EthIf_Init).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #75637: [diverse] Remove requirements like initializiation requirements regarding the <Module>_Init API

  Problem description:

  Generally a Caveat can never start a requirement. For this all basic Software module shall be checked and cleaned up for this formulation.

  original finding was:

  According to the discussion of RfC# 67678 we detect requirements regarding the initializiation of the certain api's, e.g.:
  "
  ...
  [SWS_EthIf_00213] ?
  Caveat: The function requires previous initialization (EthIf_Init). ?()
  "
  Such requirment are not implementation relevant. Please remove this and add as

note.
–Last change on issue 75637 comment 2–

Agreed solution:

General:
- Remove requirements which state that previous initialization of <Modul>_Init() is required, which are formulated like a note, e.g. : [SWS_EthIf_00213] ?
Caveat: The function requires previous initialization (EthIf_Init). ?()
- Add general requirement as note to ch. 8.3 Function definitions
- Remove or reformulate all other requirements, which start with caveat. Most of them are simply a note or implementation hint.

=== EhtIf ===
~8.3 Function definitions
+Note: All functions in this chapter requires previous initialization (EthIf_Init), except the following ones:
EthIf_Init, EthIf_GetVersionInfo

~8.3.2 EthIf_SetControllerMode
-[SWS_EthIf_00038]

~8.3.3 EthIf_GetControllerMode
-[SWS_EthIf_00044]

~8.3.8 EthIf_ CheckWakeup
-[SWS_EthIf_00249]

~8.3.9 EthIf_GetPhysAddr
-[SWS_EthIf_00066]

~8.3.10 EthIf_SetPhysAddr
-[SWS_EthIf_00138]

~8.3.11 EthIf_UpdatePhysAddrFilter
-[SWS_EthIf_00144]

~8.3.12 EthIf_GetPortMacAddr
-[SWS_EthIf_00195]

~8.3.13 EthIf_GetArlTable
-[SWS_EthIf_00201]

8.3.14 EthIf_GetBufferLevel
-[SWS_EthIf_00207]

~8.3.15 EthIf_GetDropCount
-[SWS_EthIf_00213]

~8.3.16 EthIf_StoreConfiguration
-[SWS_EthIf_00218]

~8.3.17 EthIf_ResetConfiguration
-[SWS_EthIf_00223]

8.3.18 EthIf_GetCurrentTime
-[SWS_EthIf_00159]

~8.3.19 EthIf_EnableEgressTimeStamp
-[SWS_EthIf_00165]

~8.3.20 EthIf_GetEgressTimeStamp
-[SWS_EthIf_00171]

~8.3.21 EthIf_GetIngressTimeStamp
-[SWS_EthIf_00177]

~8.3.22 EthIf_SetCorrectionTime
-[SWS_EthIf_00183]

~8.3.23 EthIf_SetGlobalTime
-[SWS_EthIf_00189]

~8.3.24 EthIf_ProvideTxBuffer
-[SWS_EthIf_00074]

~8.3.25 EthIf_Transmit
-[SWS_EthIf_00081]

=== CanIf ===
General note for below items which contain phrases like "The CAN Interface module must be initialized after Power ON": this is not needed anymore cause already stated in SWS_CANIF_00661. So e.g. in cases where result from CanDrv is just passed through the whole requirement can be deleted.

Also it is assumed, that the mindset like defined in SWS_CANIF_00661 is there for all upper layer functions like <User_TxConfirmation> which might be called by CanIf.

Because if CanDrv would not be already initialised it would return E_NOT_OK which is passed through:
-SWS_CANIF_00312
-SWS_CANIF_00316

Because already guarded by SWS_CANIF_00661:
-SWS_CANIF_00334
-SWS_CANIF_00339
-SWS_CANIF_00344
-SWS_CANIF_00349

~SWS_CANIF_00661
All CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall not execute their normal operation and return E_NOT_OK unless the CanIf has been initialized with a preceding call of CanIf_Init().

+SWS_CANIF_????1
If the switch CANIF_PUBLIC_DEV_ERROR_DETECT is enabled, all CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall report to the DET (using CANIF_E_UNINIT) unless the CanIf has been initialized with a preceding call of CanIf_Init().

-SWS_CANIF_00323 change to
Note: During the call of CanIf_Transmit() the buffer of PduInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00329
CanIf_ReadRxPduData() shall not be used for CanIfRxSduId, which are defined to receive multiple CAN-Ids (range reception).

Add note below SWS_CANIF_00329
Note: During the call of of CanIf_ReadRxPduData() the buffer of CanIfRxInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00356
CanIf_SetDynamicTxId() shall not be interrupted by CanIf_Transmit(), if the same L-SDU ID is handled.

-SWS_CANIF_00407 change to
Note: The call context of CanIf_CheckValidation() is either on interrupt level (interrupt mode) or on task level (polling mode).
Caveat: The corresponding CAN controller and transceiver must be switched on via CanTrcv_SetOpMode(Transceiver, CANTRCV_TRCVMODE_NORMAL) and Can_SetControllerMode(Controller, CAN_CS_STARTED) and the corresponding mode indications must have been called.

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
-SWS_CANIF_00737 CanIf_GetTxConfirmationState()
-SWS_CANIF_00870 CanIf_SetBaudrate()

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
-SWS_CANIF_00401 CanIf_CheckWakeup()
-SWS_CANIF_00413 CanIf_TxConfirmation()
-SWS_CANIF_00422 CanIf_RxIndication()
-SWS_CANIF_00432 CanIf_ControllerBusOff()
-SWS_CANIF_00818 CanIf_ConfirmPnAvailability()
-SWS_CANIF_00807 CanIf_ClearTrcvWufFlagIndication()
-SWS_CANIF_00811 CanIf_CheckTrcvWakeFlagIndication()
-SWS_CANIF_00703 CanIf_ControllerModeIndication()
-SWS_CANIF_00709 CanIf_TrcvModeIndication()
-SWS_CANIF_00887 <User_TriggerTransmit>()
-SWS_CANIF_00437 <User_TxConfirmation>()

-SWS_CANIF_00440 change to
Note: Until <User_RxIndication>() returns, CanIf will not access PduInfoPtr. The PduInfoPtr is only valid and can be used by upper layers, until the indication returns. CanIf guarantees that the number of configured bytes for this PduInfoPtr is valid.
Note: The call context of <User_RxIndication>() is either on interrupt level (interrupt mode) or on task level (polling mode).

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00822 <User_ConfirmPnAvailability>()
-SWS_CANIF_00793 <User_ClearTrcvWufFlagIndication>()
-SWS_CANIF_00799 <User_CheckTrcvWakeFlagIndication>()

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00688 <User_ControllerModeIndication>()

=== ComM ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (ComM_Init), except the following ones:
ComM_Init, ComM_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the ComM module is initialized correctly.
-[SWS_ComM_00805] Caveats of ComM_Nm_NetworkStartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00806] Caveats of ComM_Nm_NetworkMode: The Communication

Manager Module is initialized correctly.()
-[SWS_ComM_00808] Caveats of ComM_Nm_PrepareBusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00810] Caveats of ComM_Nm_BusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00812] Caveats of ComM_Nm_RestartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00814] Caveats of ComM_EcuM_WakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00965] Caveats of ComM_EcuM_PNCWakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00816] Caveats of ComM_BusSM_ModeIndication(): The Communication Manager Module is initialized correctly.()

=== LdCom ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (LdCom_Init), except the following ones:
LdCom_Init, LdCom_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the LdCom module is initialized correctly.
–Last change on issue 75637 comment 23–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.41 Specification Item SWS_EthIf_00167

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZEDUNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of ’Runtime errors’

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification "belongs".

  *** BSW UML Model ***
  SWS_CanNm:
  ———-
  Chapter 8.6.1 Optional Interfaces:
  Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

  SWS_LinIf:
  ———-
  SWS_LinIf_00359: add Det_ReportRuntimeError

  SWS_UdpNm:
  ———-
  Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

  *** ECUC XML ***
  Not affected. No configuration of runtime error reporting required (see SWS BSW General).
  –Last change on issue 59085 comment 88–

  BW-C-Level:

Application	Specification	Bus
1	4	1

1.42 Specification Item SWS_EthIf_00171

Trace References:

none

Content:

Caveat: The function requires previous initialization (EthIf_Init).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #75637: [diverse] Remove requirements like initializiation requirements regarding the <Module>_Init API

  Problem description:

  Generally a Caveat can never start a requirement. For this all basic Software module shall be checked and cleaned up for this formulation.

  original finding was:

  According to the discussion of RfC# 67678 we detect requirements regarding the initializiation of the certain api's, e.g.:
  "
  ...
  [SWS_EthIf_00213] ?
  Caveat: The function requires previous initialization (EthIf_Init). ?()
  "
  Such requirment are not implementation relevant. Please remove this and add as note.
  –Last change on issue 75637 comment 2–

  Agreed solution:

  General:
  - Remove requirements which state that previous initialization of <Modul>_Init() is required, which are formulated like a note, e.g. : [SWS_EthIf_00213] ?
  Caveat: The function requires previous initialization (EthIf_Init). ?()
  - Add general requirement as note to ch. 8.3 Function definitions
  - Remove or reformulate all other requirements, which start with caveat. Most of them are simply a note or implementation hint.

  === EhtIf ===
  ~8.3 Function definitions

+Note: All functions in this chapter requires previous initialization (EthIf_Init), except the following ones:
EthIf_Init, EthIf_GetVersionInfo

~8.3.2 EthIf_SetControllerMode
-[SWS_EthIf_00038]

~8.3.3 EthIf_GetControllerMode
-[SWS_EthIf_00044]

~8.3.8 EthIf_ CheckWakeup
-[SWS_EthIf_00249]

~8.3.9 EthIf_GetPhysAddr
-[SWS_EthIf_00066]

~8.3.10 EthIf_SetPhysAddr
-[SWS_EthIf_00138]

~8.3.11 EthIf_UpdatePhysAddrFilter
-[SWS_EthIf_00144]

~8.3.12 EthIf_GetPortMacAddr
-[SWS_EthIf_00195]

~8.3.13 EthIf_GetArlTable
-[SWS_EthIf_00201]

8.3.14 EthIf_GetBufferLevel
-[SWS_EthIf_00207]

~8.3.15 EthIf_GetDropCount
-[SWS_EthIf_00213]

~8.3.16 EthIf_StoreConfiguration
-[SWS_EthIf_00218]

~8.3.17 EthIf_ResetConfiguration
-[SWS_EthIf_00223]

8.3.18 EthIf_GetCurrentTime
-[SWS_EthIf_00159]

~8.3.19 EthIf_EnableEgressTimeStamp
-[SWS_EthIf_00165]

~8.3.20 EthIf_GetEgressTimeStamp
-[SWS_EthIf_00171]

~8.3.21 EthIf_GetIngressTimeStamp
-[SWS_EthIf_00177]

~8.3.22 EthIf_SetCorrectionTime
-[SWS_EthIf_00183]

~8.3.23 EthIf_SetGlobalTime
-[SWS_EthIf_00189]

~8.3.24 EthIf_ProvideTxBuffer
-[SWS_EthIf_00074]

~8.3.25 EthIf_Transmit
-[SWS_EthIf_00081]

=== CanIf ===
General note for below items which contain phrases like "The CAN Interface module must be initialized after Power ON": this is not needed anymore cause already stated in SWS_CANIF_00661. So e.g. in cases where result from CanDrv is just passed through the whole requirement can be deleted.

Also it is assumed, that the mindset like defined in SWS_CANIF_00661 is there for all upper layer functions like <User_TxConfirmation> which might be called by CanIf.

Because if CanDrv would not be already initialised it would return E_NOT_OK which is passed through:
-SWS_CANIF_00312
-SWS_CANIF_00316

Because already guarded by SWS_CANIF_00661:
-SWS_CANIF_00334
-SWS_CANIF_00339

-SWS_CANIF_00344
-SWS_CANIF_00349

~SWS_CANIF_00661
All CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall not execute their normal operation and return E_NOT_OK unless the CanIf has been initialized with a preceding call of CanIf_Init().

+SWS_CANIF_????1
If the switch CANIF_PUBLIC_DEV_ERROR_DETECT is enabled, all CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall report to the DET (using CANIF_E_UNINIT) unless the CanIf has been initialized with a preceding call of CanIf_Init().

-SWS_CANIF_00323 change to
Note: During the call of CanIf_Transmit() the buffer of PduInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00329
CanIf_ReadRxPduData() shall not be used for CanIfRxSduId, which are defined to receive multiple CAN-Ids (range reception).

Add note below SWS_CANIF_00329
Note: During the call of of CanIf_ReadRxPduData() the buffer of CanIfRxInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00356
CanIf_SetDynamicTxId() shall not be interrupted by CanIf_Transmit(), if the same L-SDU ID is handled.

-SWS_CANIF_00407 change to
Note: The call context of CanIf_CheckValidation() is either on interrupt level (interrupt mode) or on task level (polling mode).

Caveat: The corresponding CAN controller and transceiver must be switched on via CanTrcv_SetOpMode(Transceiver, CANTRCV_TRCVMODE_NORMAL) and Can_SetControllerMode(Controller, CAN_CS_STARTED) and the corresponding mode indications must have been called.

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
-SWS_CANIF_00737 CanIf_GetTxConfirmationState()
-SWS_CANIF_00870 CanIf_SetBaudrate()

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
-SWS_CANIF_00401 CanIf_CheckWakeup()
-SWS_CANIF_00413 CanIf_TxConfirmation()
-SWS_CANIF_00422 CanIf_RxIndication()
-SWS_CANIF_00432 CanIf_ControllerBusOff()
-SWS_CANIF_00818 CanIf_ConfirmPnAvailability()
-SWS_CANIF_00807 CanIf_ClearTrcvWufFlagIndication()
-SWS_CANIF_00811 CanIf_CheckTrcvWakeFlagIndication()
-SWS_CANIF_00703 CanIf_ControllerModeIndication()
-SWS_CANIF_00709 CanIf_TrcvModeIndication()
-SWS_CANIF_00887 <User_TriggerTransmit>()
-SWS_CANIF_00437 <User_TxConfirmation>()

-SWS_CANIF_00440 change to
Note: Until <User_RxIndication>() returns, CanIf will not access PduInfoPtr. The PduInfoPtr is only valid and can be used by upper layers, until the indication returns. CanIf guarantees that the number of configured bytes for this PduInfoPtr is valid.
Note: The call context of <User_RxIndication>() is either on interrupt level (interrupt mode) or on task level (polling mode).

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN

Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00822 <User_ConfirmPnAvailability>()
-SWS_CANIF_00793 <User_ClearTrcvWufFlagIndication>()
-SWS_CANIF_00799 <User_CheckTrcvWakeFlagIndication>()

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00688 <User_ControllerModeIndication>()

=== ComM ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (ComM_Init), except the following ones:
ComM_Init, ComM_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the ComM module is initialized correctly.
-[SWS_ComM_00805] Caveats of ComM_Nm_NetworkStartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00806] Caveats of ComM_Nm_NetworkMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00808] Caveats of ComM_Nm_PrepareBusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00810] Caveats of ComM_Nm_BusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00812] Caveats of ComM_Nm_RestartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00814] Caveats of ComM_EcuM_WakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00965] Caveats of ComM_EcuM_PNCWakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00816] Caveats of ComM_BusSM_ModeIndication(): The Communication Manager Module is initialized correctly.()

=== LdCom ===

~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (LdCom_Init), except the following ones:
LdCom_Init, LdCom_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the LdCom module is initialized correctly.
–Last change on issue 75637 comment 23–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.43 Specification Item SWS_EthIf_00172

Trace References:

none

Content:

Service name:	EthIf_GetIngressTimeStampEthIf_GetIngressTimeStamp	
Syntax:	Std_ReturnType EthIf_GetIngressTimeStamp(uint8 CtrlIdx, const Eth_DataType* DataPtr, Eth_TimeStampQualType* timeQualPtr, Eth_TimeStampType* timeStampPtr)	
Service ID[hex]:	0x25	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	CtrlIdxEthIf_GetIngressTimeStamp.CtrlIdx	Index of the addresses ETH controller.
	DataPtrEthIf_GetIngressTimeStamp.DataPtr	Pointer to the message buffer, where Application expects ingress time stamping
Parameters (inout):	None	
Parameters (out):	timeQualPtrEthIf_GetIngressTimeStamp.timeQualPtr	quality of HW time stamp, e.g. based on current drift
	timeStampPtrEthIf_GetIngressTimeStamp.timeStampPtr	current time stamp
Return value:	Std_ReturnType	–
Description:	Reads back the ingress time stamp on a dedicated message object. It must be called within the RxIndication() function.	

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

1.44 Specification Item SWS_EthIf_00173

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZEDUNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of ’Runtime errors’

  Problem description:

Inconsistencies in SWS with semantics of Default errors
–Last change on issue 59085 comment 26–

Agreed solution:

solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

Notes:
- It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
- The review task of the ITs shall be done by the WP to which the specification "belongs".

*** BSW UML Model ***
SWS_CanNm:
———-
Chapter 8.6.1 Optional Interfaces:
Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

SWS_LinIf:
———-
SWS_LinIf_00359: add Det_ReportRuntimeError

SWS_UdpNm:
———-
Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.45 Specification Item SWS_EthIf_00177

Trace References:

none

Content:

Caveat: The function requires previous initialization (EthIf_Init).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #75637: [diverse] Remove requirements like initializiation requirements regarding the <Module>_Init API

  Problem description:

  Generally a Caveat can never start a requirement. For this all basic Software module shall be checked and cleaned up for this formulation.

  original finding was:

  According to the discussion of RfC# 67678 we detect requirements regarding the initializiation of the certain api's, e.g.:
  "
  ...
  [SWS_EthIf_00213] ?
  Caveat: The function requires previous initialization (EthIf_Init). ?()
  "
  Such requirment are not implementation relevant. Please remove this and add as note.
  –Last change on issue 75637 comment 2–

  Agreed solution:

  General:
  - Remove requirements which state that previous initialization of <Modul>_Init() is required, which are formulated like a note, e.g. : [SWS_EthIf_00213] ?
  Caveat: The function requires previous initialization (EthIf_Init). ?()
  - Add general requirement as note to ch. 8.3 Function definitions
  - Remove or reformulate all other requirements, which start with caveat. Most of them are simply a note or implementation hint.

  === EhtIf ===
  ~8.3 Function definitions

+Note: All functions in this chapter requires previous initialization (EthIf_Init), except the following ones:
EthIf_Init, EthIf_GetVersionInfo

~8.3.2 EthIf_SetControllerMode
-[SWS_EthIf_00038]

~8.3.3 EthIf_GetControllerMode
-[SWS_EthIf_00044]

~8.3.8 EthIf_ CheckWakeup
-[SWS_EthIf_00249]

~8.3.9 EthIf_GetPhysAddr
-[SWS_EthIf_00066]

~8.3.10 EthIf_SetPhysAddr
-[SWS_EthIf_00138]

~8.3.11 EthIf_UpdatePhysAddrFilter
-[SWS_EthIf_00144]

~8.3.12 EthIf_GetPortMacAddr
-[SWS_EthIf_00195]

~8.3.13 EthIf_GetArlTable
-[SWS_EthIf_00201]

8.3.14 EthIf_GetBufferLevel
-[SWS_EthIf_00207]

~8.3.15 EthIf_GetDropCount
-[SWS_EthIf_00213]

~8.3.16 EthIf_StoreConfiguration
-[SWS_EthIf_00218]

~8.3.17 EthIf_ResetConfiguration
-[SWS_EthIf_00223]

8.3.18 EthIf_GetCurrentTime
-[SWS_EthIf_00159]

~8.3.19 EthIf_EnableEgressTimeStamp
-[SWS_EthIf_00165]

~8.3.20 EthIf_GetEgressTimeStamp
-[SWS_EthIf_00171]

~8.3.21 EthIf_GetIngressTimeStamp
-[SWS_EthIf_00177]

~8.3.22 EthIf_SetCorrectionTime
-[SWS_EthIf_00183]

~8.3.23 EthIf_SetGlobalTime
-[SWS_EthIf_00189]

~8.3.24 EthIf_ProvideTxBuffer
-[SWS_EthIf_00074]

~8.3.25 EthIf_Transmit
-[SWS_EthIf_00081]

=== CanIf ===
General note for below items which contain phrases like "The CAN Interface module must be initialized after Power ON": this is not needed anymore cause already stated in SWS_CANIF_00661. So e.g. in cases where result from CanDrv is just passed through the whole requirement can be deleted.

Also it is assumed, that the mindset like defined in SWS_CANIF_00661 is there for all upper layer functions like <User_TxConfirmation> which might be called by CanIf.

Because if CanDrv would not be already initialised it would return E_NOT_OK which is passed through:
-SWS_CANIF_00312
-SWS_CANIF_00316

Because already guarded by SWS_CANIF_00661:
-SWS_CANIF_00334
-SWS_CANIF_00339

-SWS_CANIF_00344
-SWS_CANIF_00349

~SWS_CANIF_00661
All CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall not execute their normal operation and return E_NOT_OK unless the CanIf has been initialized with a preceding call of CanIf_Init().

+SWS_CANIF_????1
If the switch CANIF_PUBLIC_DEV_ERROR_DETECT is enabled, all CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall report to the DET (using CANIF_E_UNINIT) unless the CanIf has been initialized with a preceding call of CanIf_Init().

-SWS_CANIF_00323 change to
Note: During the call of CanIf_Transmit() the buffer of PduInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00329
CanIf_ReadRxPduData() shall not be used for CanIfRxSduId, which are defined to receive multiple CAN-Ids (range reception).

Add note below SWS_CANIF_00329
Note: During the call of of CanIf_ReadRxPduData() the buffer of CanIfRxInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00356
CanIf_SetDynamicTxId() shall not be interrupted by CanIf_Transmit(), if the same L-SDU ID is handled.

-SWS_CANIF_00407 change to
Note: The call context of CanIf_CheckValidation() is either on interrupt level (interrupt mode) or on task level (polling mode).

Caveat: The corresponding CAN controller and transceiver must be switched on via CanTrcv_SetOpMode(Transceiver, CANTRCV_TRCVMODE_NORMAL) and Can_SetControllerMode(Controller, CAN_CS_STARTED) and the corresponding mode indications must have been called.

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
-SWS_CANIF_00737 CanIf_GetTxConfirmationState()
-SWS_CANIF_00870 CanIf_SetBaudrate()

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
-SWS_CANIF_00401 CanIf_CheckWakeup()
-SWS_CANIF_00413 CanIf_TxConfirmation()
-SWS_CANIF_00422 CanIf_RxIndication()
-SWS_CANIF_00432 CanIf_ControllerBusOff()
-SWS_CANIF_00818 CanIf_ConfirmPnAvailability()
-SWS_CANIF_00807 CanIf_ClearTrcvWufFlagIndication()
-SWS_CANIF_00811 CanIf_CheckTrcvWakeFlagIndication()
-SWS_CANIF_00703 CanIf_ControllerModeIndication()
-SWS_CANIF_00709 CanIf_TrcvModeIndication()
-SWS_CANIF_00887 <User_TriggerTransmit>()
-SWS_CANIF_00437 <User_TxConfirmation>()

-SWS_CANIF_00440 change to
Note: Until <User_RxIndication>() returns, CanIf will not access PduInfoPtr. The PduInfoPtr is only valid and can be used by upper layers, until the indication returns. CanIf guarantees that the number of configured bytes for this PduInfoPtr is valid.
Note: The call context of <User_RxIndication>() is either on interrupt level (interrupt mode) or on task level (polling mode).

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN

Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00822 <User_ConfirmPnAvailability>()
-SWS_CANIF_00793 <User_ClearTrcvWufFlagIndication>()
-SWS_CANIF_00799 <User_CheckTrcvWakeFlagIndication>()

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00688 <User_ControllerModeIndication>()

=== ComM ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (ComM_Init), except the following ones:
ComM_Init, ComM_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the ComM module is initialized correctly.
-[SWS_ComM_00805] Caveats of ComM_Nm_NetworkStartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00806] Caveats of ComM_Nm_NetworkMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00808] Caveats of ComM_Nm_PrepareBusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00810] Caveats of ComM_Nm_BusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00812] Caveats of ComM_Nm_RestartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00814] Caveats of ComM_EcuM_WakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00965] Caveats of ComM_EcuM_PNCWakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00816] Caveats of ComM_BusSM_ModeIndication(): The Communication Manager Module is initialized correctly.()

=== LdCom ===

~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (LdCom_Init), except the following ones:
LdCom_Init, LdCom_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the LdCom module is initialized correctly.
–Last change on issue 75637 comment 23–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.46 Specification Item SWS_EthIf_00193

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZEDUNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification

"belongs".

*** BSW UML Model ***
SWS_CanNm:
———-
Chapter 8.6.1 Optional Interfaces:
Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

SWS_LinIf:
———-
SWS_LinIf_00359: add Det_ReportRuntimeError

SWS_UdpNm:
———-
Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.47 Specification Item SWS_EthIf_00195

Trace References:

none

Content:

Caveat: The function requires previous initialization (EthIf_Init).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #75637: [diverse] Remove requirements like initializiation requirements regarding the <Module>_Init API

Problem description:

Generally a Caveat can never start a requirement. For this all basic Software module shall be checked and cleaned up for this formulation.

original finding was:

According to the discussion of RfC# 67678 we detect requirements regarding the initializiation of the certain api's, e.g.:
"
...
[SWS_EthIf_00213] ?
Caveat: The function requires previous initialization (EthIf_Init). ?()
"
Such requirment are not implementation relevant. Please remove this and add as note.
–Last change on issue 75637 comment 2–

Agreed solution:

General:
- Remove requirements which state that previous initialization of <Modul>_Init() is required, which are formulated like a note, e.g. : [SWS_EthIf_00213] ?
Caveat: The function requires previous initialization (EthIf_Init). ?()
- Add general requirement as note to ch. 8.3 Function definitions
- Remove or reformulate all other requirements, which start with caveat. Most of them are simply a note or implementation hint.

=== EhtIf ===
~8.3 Function definitions
+Note: All functions in this chapter requires previous initialization (EthIf_Init), except the following ones:
EthIf_Init, EthIf_GetVersionInfo

~8.3.2 EthIf_SetControllerMode
-[SWS_EthIf_00038]

~8.3.3 EthIf_GetControllerMode
-[SWS_EthIf_00044]

~8.3.8 EthIf_ CheckWakeup
-[SWS_EthIf_00249]

~8.3.9 EthIf_GetPhysAddr
-[SWS_EthIf_00066]

~8.3.10 EthIf_SetPhysAddr
-[SWS_EthIf_00138]

~8.3.11 EthIf_UpdatePhysAddrFilter
-[SWS_EthIf_00144]

~8.3.12 EthIf_GetPortMacAddr
-[SWS_EthIf_00195]

~8.3.13 EthIf_GetArlTable
-[SWS_EthIf_00201]

8.3.14 EthIf_GetBufferLevel
-[SWS_EthIf_00207]

~8.3.15 EthIf_GetDropCount
-[SWS_EthIf_00213]

~8.3.16 EthIf_StoreConfiguration
-[SWS_EthIf_00218]

~8.3.17 EthIf_ResetConfiguration
-[SWS_EthIf_00223]

8.3.18 EthIf_GetCurrentTime
-[SWS_EthIf_00159]

~8.3.19 EthIf_EnableEgressTimeStamp
-[SWS_EthIf_00165]

~8.3.20 EthIf_GetEgressTimeStamp
-[SWS_EthIf_00171]

~8.3.21 EthIf_GetIngressTimeStamp
-[SWS_EthIf_00177]

~8.3.22 EthIf_SetCorrectionTime
-[SWS_EthIf_00183]

~8.3.23 EthIf_SetGlobalTime

-[SWS_EthIf_00189]

~8.3.24 EthIf_ProvideTxBuffer
-[SWS_EthIf_00074]

~8.3.25 EthIf_Transmit
-[SWS_EthIf_00081]

=== CanIf ===
General note for below items which contain phrases like "The CAN Interface module must be initialized after Power ON": this is not needed anymore cause already stated in SWS_CANIF_00661. So e.g. in cases where result from CanDrv is just passed through the whole requirement can be deleted.

Also it is assumed, that the mindset like defined in SWS_CANIF_00661 is there for all upper layer functions like <User_TxConfirmation> which might be called by CanIf.

Because if CanDrv would not be already initialised it would return E_NOT_OK which is passed through:
-SWS_CANIF_00312
-SWS_CANIF_00316

Because already guarded by SWS_CANIF_00661:
-SWS_CANIF_00334
-SWS_CANIF_00339
-SWS_CANIF_00344
-SWS_CANIF_00349

~SWS_CANIF_00661
All CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall not execute their normal operation and return E_NOT_OK unless the CanIf has been initialized with a preceding call of CanIf_Init().

+SWS_CANIF_????1
If the switch CANIF_PUBLIC_DEV_ERROR_DETECT is enabled, all CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall report to the DET

(using CANIF_E_UNINIT) unless the CanIf has been initialized with a preceding call of CanIf_Init().

-SWS_CANIF_00323 change to
Note: During the call of CanIf_Transmit() the buffer of PduInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00329
CanIf_ReadRxPduData() shall not be used for CanIfRxSduId, which are defined to receive multiple CAN-Ids (range reception).

Add note below SWS_CANIF_00329
Note: During the call of of CanIf_ReadRxPduData() the buffer of CanIfRxInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00356
CanIf_SetDynamicTxId() shall not be interrupted by CanIf_Transmit(), if the same L-SDU ID is handled.

-SWS_CANIF_00407 change to
Note: The call context of CanIf_CheckValidation() is either on interrupt level (interrupt mode) or on task level (polling mode).
Caveat: The corresponding CAN controller and transceiver must be switched on via CanTrcv_SetOpMode(Transceiver, CANTRCV_TRCVMODE_NORMAL) and Can_SetControllerMode(Controller, CAN_CS_STARTED) and the corresponding mode indications must have been called.

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
-SWS_CANIF_00737 CanIf_GetTxConfirmationState()
-SWS_CANIF_00870 CanIf_SetBaudrate()

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
-SWS_CANIF_00401 CanIf_CheckWakeup()
-SWS_CANIF_00413 CanIf_TxConfirmation()
-SWS_CANIF_00422 CanIf_RxIndication()
-SWS_CANIF_00432 CanIf_ControllerBusOff()
-SWS_CANIF_00818 CanIf_ConfirmPnAvailability()
-SWS_CANIF_00807 CanIf_ClearTrcvWufFlagIndication()
-SWS_CANIF_00811 CanIf_CheckTrcvWakeFlagIndication()
-SWS_CANIF_00703 CanIf_ControllerModeIndication()
-SWS_CANIF_00709 CanIf_TrcvModeIndication()
-SWS_CANIF_00887 <User_TriggerTransmit>()
-SWS_CANIF_00437 <User_TxConfirmation>()

-SWS_CANIF_00440 change to
Note: Until <User_RxIndication>() returns, CanIf will not access PduInfoPtr. The PduInfoPtr is only valid and can be used by upper layers, until the indication returns. CanIf guarantees that the number of configured bytes for this PduInfoPtr is valid.
Note: The call context of <User_RxIndication>() is either on interrupt level (interrupt mode) or on task level (polling mode).

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00822 <User_ConfirmPnAvailability>()
-SWS_CANIF_00793 <User_ClearTrcvWufFlagIndication>()
-SWS_CANIF_00799 <User_CheckTrcvWakeFlagIndication>()

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"

-SWS_CANIF_00688 <User_ControllerModeIndication>()

=== ComM ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (ComM_Init), except the following ones:
ComM_Init, ComM_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the ComM module is initialized correctly.
-[SWS_ComM_00805] Caveats of ComM_Nm_NetworkStartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00806] Caveats of ComM_Nm_NetworkMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00808] Caveats of ComM_Nm_PrepareBusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00810] Caveats of ComM_Nm_BusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00812] Caveats of ComM_Nm_RestartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00814] Caveats of ComM_EcuM_WakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00965] Caveats of ComM_EcuM_PNCWakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00816] Caveats of ComM_BusSM_ModeIndication(): The Communication Manager Module is initialized correctly.()

=== LdCom ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (LdCom_Init), except the following ones:
LdCom_Init, LdCom_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the LdCom module is initialized correctly.
–Last change on issue 75637 comment 23–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.48 Specification Item SWS_EthIf_00199

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZEDUNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of ’Runtime errors’

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification "belongs".

  *** BSW UML Model ***
  SWS_CanNm:
  ———-
  Chapter 8.6.1 Optional Interfaces:
  Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

  SWS_LinIf:
  ———-
  SWS_LinIf_00359: add Det_ReportRuntimeError

  SWS_UdpNm:
  ———-

Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.49 Specification Item SWS_EthIf_00201

Trace References:

none

Content:

Caveat: The function requires previous initialization (EthIf_Init).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #75637: [diverse] Remove requirements like initializiation requirements regarding the <Module>_Init API

  Problem description:

  Generally a Caveat can never start a requirement. For this all basic Software module shall be checked and cleaned up for this formulation.

  original finding was:

  According to the discussion of RfC# 67678 we detect requirements regarding the initializiation of the certain api's, e.g.:
  "
  ...
  [SWS_EthIf_00213] ?
  Caveat: The function requires previous initialization (EthIf_Init). ?()
  "
  Such requirment are not implementation relevant. Please remove this and add as

note.
–Last change on issue 75637 comment 2–

Agreed solution:

General:
- Remove requirements which state that previous initialization of <Modul>_Init() is required, which are formulated like a note, e.g. : [SWS_EthIf_00213] ?
Caveat: The function requires previous initialization (EthIf_Init). ?()
- Add general requirement as note to ch. 8.3 Function definitions
- Remove or reformulate all other requirements, which start with caveat. Most of them are simply a note or implementation hint.

=== EhtIf ===
~8.3 Function definitions
+Note: All functions in this chapter requires previous initialization (EthIf_Init), except the following ones:
EthIf_Init, EthIf_GetVersionInfo

~8.3.2 EthIf_SetControllerMode
-[SWS_EthIf_00038]

~8.3.3 EthIf_GetControllerMode
-[SWS_EthIf_00044]

~8.3.8 EthIf_ CheckWakeup
-[SWS_EthIf_00249]

~8.3.9 EthIf_GetPhysAddr
-[SWS_EthIf_00066]

~8.3.10 EthIf_SetPhysAddr
-[SWS_EthIf_00138]

~8.3.11 EthIf_UpdatePhysAddrFilter
-[SWS_EthIf_00144]

~8.3.12 EthIf_GetPortMacAddr
-[SWS_EthIf_00195]

~8.3.13 EthIf_GetArlTable
-[SWS_EthIf_00201]

8.3.14 EthIf_GetBufferLevel
-[SWS_EthIf_00207]

~8.3.15 EthIf_GetDropCount
-[SWS_EthIf_00213]

~8.3.16 EthIf_StoreConfiguration
-[SWS_EthIf_00218]

~8.3.17 EthIf_ResetConfiguration
-[SWS_EthIf_00223]

8.3.18 EthIf_GetCurrentTime
-[SWS_EthIf_00159]

~8.3.19 EthIf_EnableEgressTimeStamp
-[SWS_EthIf_00165]

~8.3.20 EthIf_GetEgressTimeStamp
-[SWS_EthIf_00171]

~8.3.21 EthIf_GetIngressTimeStamp
-[SWS_EthIf_00177]

~8.3.22 EthIf_SetCorrectionTime
-[SWS_EthIf_00183]

~8.3.23 EthIf_SetGlobalTime
-[SWS_EthIf_00189]

~8.3.24 EthIf_ProvideTxBuffer
-[SWS_EthIf_00074]

~8.3.25 EthIf_Transmit
-[SWS_EthIf_00081]

=== CanIf ===
General note for below items which contain phrases like "The CAN Interface module must be initialized after Power ON": this is not needed anymore cause already stated in SWS_CANIF_00661. So e.g. in cases where result from CanDrv is just passed through the whole requirement can be deleted.

Also it is assumed, that the mindset like defined in SWS_CANIF_00661 is there for all upper layer functions like <User_TxConfirmation> which might be called by CanIf.

Because if CanDrv would not be already initialised it would return E_NOT_OK which is passed through:
-SWS_CANIF_00312
-SWS_CANIF_00316

Because already guarded by SWS_CANIF_00661:
-SWS_CANIF_00334
-SWS_CANIF_00339
-SWS_CANIF_00344
-SWS_CANIF_00349

~SWS_CANIF_00661
All CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall not execute their normal operation and return E_NOT_OK unless the CanIf has been initialized with a preceding call of CanIf_Init().

+SWS_CANIF_????1
If the switch CANIF_PUBLIC_DEV_ERROR_DETECT is enabled, all CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall report to the DET (using CANIF_E_UNINIT) unless the CanIf has been initialized with a preceding call of CanIf_Init().

-SWS_CANIF_00323 change to
Note: During the call of CanIf_Transmit() the buffer of PduInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00329
CanIf_ReadRxPduData() shall not be used for CanIfRxSduId, which are defined to receive multiple CAN-Ids (range reception).

Add note below SWS_CANIF_00329
Note: During the call of of CanIf_ReadRxPduData() the buffer of CanIfRxInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00356
CanIf_SetDynamicTxId() shall not be interrupted by CanIf_Transmit(), if the same L-SDU ID is handled.

-SWS_CANIF_00407 change to
Note: The call context of CanIf_CheckValidation() is either on interrupt level (interrupt mode) or on task level (polling mode).
Caveat: The corresponding CAN controller and transceiver must be switched on via CanTrcv_SetOpMode(Transceiver, CANTRCV_TRCVMODE_NORMAL) and Can_SetControllerMode(Controller, CAN_CS_STARTED) and the corresponding mode indications must have been called.

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
-SWS_CANIF_00737 CanIf_GetTxConfirmationState()
-SWS_CANIF_00870 CanIf_SetBaudrate()

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
-SWS_CANIF_00401 CanIf_CheckWakeup()
-SWS_CANIF_00413 CanIf_TxConfirmation()
-SWS_CANIF_00422 CanIf_RxIndication()
-SWS_CANIF_00432 CanIf_ControllerBusOff()
-SWS_CANIF_00818 CanIf_ConfirmPnAvailability()
-SWS_CANIF_00807 CanIf_ClearTrcvWufFlagIndication()
-SWS_CANIF_00811 CanIf_CheckTrcvWakeFlagIndication()
-SWS_CANIF_00703 CanIf_ControllerModeIndication()
-SWS_CANIF_00709 CanIf_TrcvModeIndication()
-SWS_CANIF_00887 <User_TriggerTransmit>()
-SWS_CANIF_00437 <User_TxConfirmation>()

-SWS_CANIF_00440 change to
Note: Until <User_RxIndication>() returns, CanIf will not access PduInfoPtr. The PduInfoPtr is only valid and can be used by upper layers, until the indication returns. CanIf guarantees that the number of configured bytes for this PduInfoPtr is valid.
Note: The call context of <User_RxIndication>() is either on interrupt level (interrupt mode) or on task level (polling mode).

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00822 <User_ConfirmPnAvailability>()
-SWS_CANIF_00793 <User_ClearTrcvWufFlagIndication>()
-SWS_CANIF_00799 <User_CheckTrcvWakeFlagIndication>()

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00688 <User_ControllerModeIndication>()

=== ComM ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (ComM_Init), except the following ones:
ComM_Init, ComM_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the ComM module is initialized correctly.
-[SWS_ComM_00805] Caveats of ComM_Nm_NetworkStartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00806] Caveats of ComM_Nm_NetworkMode: The Communication

Manager Module is initialized correctly.()
-[SWS_ComM_00808] Caveats of ComM_Nm_PrepareBusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00810] Caveats of ComM_Nm_BusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00812] Caveats of ComM_Nm_RestartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00814] Caveats of ComM_EcuM_WakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00965] Caveats of ComM_EcuM_PNCWakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00816] Caveats of ComM_BusSM_ModeIndication(): The Communication Manager Module is initialized correctly.()

=== LdCom ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (LdCom_Init), except the following ones:
LdCom_Init, LdCom_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the LdCom module is initialized correctly.
–Last change on issue 75637 comment 23–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.50 Specification Item SWS_EthIf_00205

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZEDUNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification "belongs".

  *** BSW UML Model ***
  SWS_CanNm:
  ———-
  Chapter 8.6.1 Optional Interfaces:
  Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

  SWS_LinIf:
  ———-
  SWS_LinIf_00359: add Det_ReportRuntimeError

  SWS_UdpNm:
  ———-
  Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

  *** ECUC XML ***
  Not affected. No configuration of runtime error reporting required (see SWS BSW General).
  –Last change on issue 59085 comment 88–

  BW-C-Level:

Application	Specification	Bus
1	4	1

1.51 Specification Item SWS_EthIf_00207

Trace References:

none

Content:

Caveat: The function requires previous initialization (EthIf_Init).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #75637: [diverse] Remove requirements like initializiation requirements regarding the <Module>_Init API

  Problem description:

  Generally a Caveat can never start a requirement. For this all basic Software module shall be checked and cleaned up for this formulation.

  original finding was:

  According to the discussion of RfC# 67678 we detect requirements regarding the initializiation of the certain api's, e.g.:
  "
  ...
  [SWS_EthIf_00213] ?
  Caveat: The function requires previous initialization (EthIf_Init). ?()
  "
  Such requirment are not implementation relevant. Please remove this and add as note.
  –Last change on issue 75637 comment 2–

  Agreed solution:

  General:
  - Remove requirements which state that previous initialization of <Modul>_Init() is required, which are formulated like a note, e.g. : [SWS_EthIf_00213] ?
  Caveat: The function requires previous initialization (EthIf_Init). ?()
  - Add general requirement as note to ch. 8.3 Function definitions
  - Remove or reformulate all other requirements, which start with caveat. Most of them are simply a note or implementation hint.

  === EhtIf ===
  ~8.3 Function definitions

+Note: All functions in this chapter requires previous initialization (EthIf_Init), except the following ones:
EthIf_Init, EthIf_GetVersionInfo

~8.3.2 EthIf_SetControllerMode
-[SWS_EthIf_00038]

~8.3.3 EthIf_GetControllerMode
-[SWS_EthIf_00044]

~8.3.8 EthIf_ CheckWakeup
-[SWS_EthIf_00249]

~8.3.9 EthIf_GetPhysAddr
-[SWS_EthIf_00066]

~8.3.10 EthIf_SetPhysAddr
-[SWS_EthIf_00138]

~8.3.11 EthIf_UpdatePhysAddrFilter
-[SWS_EthIf_00144]

~8.3.12 EthIf_GetPortMacAddr
-[SWS_EthIf_00195]

~8.3.13 EthIf_GetArlTable
-[SWS_EthIf_00201]

8.3.14 EthIf_GetBufferLevel
-[SWS_EthIf_00207]

~8.3.15 EthIf_GetDropCount
-[SWS_EthIf_00213]

~8.3.16 EthIf_StoreConfiguration
-[SWS_EthIf_00218]

~8.3.17 EthIf_ResetConfiguration
-[SWS_EthIf_00223]

8.3.18 EthIf_GetCurrentTime
-[SWS_EthIf_00159]

~8.3.19 EthIf_EnableEgressTimeStamp
-[SWS_EthIf_00165]

~8.3.20 EthIf_GetEgressTimeStamp
-[SWS_EthIf_00171]

~8.3.21 EthIf_GetIngressTimeStamp
-[SWS_EthIf_00177]

~8.3.22 EthIf_SetCorrectionTime
-[SWS_EthIf_00183]

~8.3.23 EthIf_SetGlobalTime
-[SWS_EthIf_00189]

~8.3.24 EthIf_ProvideTxBuffer
-[SWS_EthIf_00074]

~8.3.25 EthIf_Transmit
-[SWS_EthIf_00081]

=== CanIf ===
General note for below items which contain phrases like "The CAN Interface module must be initialized after Power ON": this is not needed anymore cause already stated in SWS_CANIF_00661. So e.g. in cases where result from CanDrv is just passed through the whole requirement can be deleted.

Also it is assumed, that the mindset like defined in SWS_CANIF_00661 is there for all upper layer functions like <User_TxConfirmation> which might be called by CanIf.

Because if CanDrv would not be already initialised it would return E_NOT_OK which is passed through:
-SWS_CANIF_00312
-SWS_CANIF_00316

Because already guarded by SWS_CANIF_00661:
-SWS_CANIF_00334
-SWS_CANIF_00339

-SWS_CANIF_00344
-SWS_CANIF_00349

~SWS_CANIF_00661
All CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall not execute their normal operation and return E_NOT_OK unless the CanIf has been initialized with a preceding call of CanIf_Init().

+SWS_CANIF_????1
If the switch CANIF_PUBLIC_DEV_ERROR_DETECT is enabled, all CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall report to the DET (using CANIF_E_UNINIT) unless the CanIf has been initialized with a preceding call of CanIf_Init().

-SWS_CANIF_00323 change to
Note: During the call of CanIf_Transmit() the buffer of PduInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00329
CanIf_ReadRxPduData() shall not be used for CanIfRxSduId, which are defined to receive multiple CAN-Ids (range reception).

Add note below SWS_CANIF_00329
Note: During the call of of CanIf_ReadRxPduData() the buffer of CanIfRxInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00356
CanIf_SetDynamicTxId() shall not be interrupted by CanIf_Transmit(), if the same L-SDU ID is handled.

-SWS_CANIF_00407 change to
Note: The call context of CanIf_CheckValidation() is either on interrupt level (interrupt mode) or on task level (polling mode).

Caveat: The corresponding CAN controller and transceiver must be switched on via CanTrcv_SetOpMode(Transceiver, CANTRCV_TRCVMODE_NORMAL) and Can_SetControllerMode(Controller, CAN_CS_STARTED) and the corresponding mode indications must have been called.

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
-SWS_CANIF_00737 CanIf_GetTxConfirmationState()
-SWS_CANIF_00870 CanIf_SetBaudrate()

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
-SWS_CANIF_00401 CanIf_CheckWakeup()
-SWS_CANIF_00413 CanIf_TxConfirmation()
-SWS_CANIF_00422 CanIf_RxIndication()
-SWS_CANIF_00432 CanIf_ControllerBusOff()
-SWS_CANIF_00818 CanIf_ConfirmPnAvailability()
-SWS_CANIF_00807 CanIf_ClearTrcvWufFlagIndication()
-SWS_CANIF_00811 CanIf_CheckTrcvWakeFlagIndication()
-SWS_CANIF_00703 CanIf_ControllerModeIndication()
-SWS_CANIF_00709 CanIf_TrcvModeIndication()
-SWS_CANIF_00887 <User_TriggerTransmit>()
-SWS_CANIF_00437 <User_TxConfirmation>()

-SWS_CANIF_00440 change to
Note: Until <User_RxIndication>() returns, CanIf will not access PduInfoPtr. The PduInfoPtr is only valid and can be used by upper layers, until the indication returns. CanIf guarantees that the number of configured bytes for this PduInfoPtr is valid.
Note: The call context of <User_RxIndication>() is either on interrupt level (interrupt mode) or on task level (polling mode).

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN

Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00822 <User_ConfirmPnAvailability>()
-SWS_CANIF_00793 <User_ClearTrcvWufFlagIndication>()
-SWS_CANIF_00799 <User_CheckTrcvWakeFlagIndication>()

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00688 <User_ControllerModeIndication>()

=== ComM ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (ComM_Init), except the following ones:
ComM_Init, ComM_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the ComM module is initialized correctly.
-[SWS_ComM_00805] Caveats of ComM_Nm_NetworkStartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00806] Caveats of ComM_Nm_NetworkMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00808] Caveats of ComM_Nm_PrepareBusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00810] Caveats of ComM_Nm_BusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00812] Caveats of ComM_Nm_RestartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00814] Caveats of ComM_EcuM_WakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00965] Caveats of ComM_EcuM_PNCWakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00816] Caveats of ComM_BusSM_ModeIndication(): The Communication Manager Module is initialized correctly.()

=== LdCom ===

~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (LdCom_Init), except the following ones:
LdCom_Init, LdCom_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the LdCom module is initialized correctly.
–Last change on issue 75637 comment 23–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.52 Specification Item SWS_EthIf_00217

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZEDUNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification

"belongs".

*** BSW UML Model ***
SWS_CanNm:
———-
Chapter 8.6.1 Optional Interfaces:
Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

SWS_LinIf:
———-
SWS_LinIf_00359: add Det_ReportRuntimeError

SWS_UdpNm:
———-
Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.53 Specification Item SWS_EthIf_00218

Trace References:

none

Content:

Caveat: The function requires previous initialization (EthIf_Init).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #75637: [diverse] Remove requirements like initializiation requirements regarding the <Module>_Init API

Problem description:

Generally a Caveat can never start a requirement. For this all basic Software module shall be checked and cleaned up for this formulation.

original finding was:

According to the discussion of RfC# 67678 we detect requirements regarding the initializiation of the certain api's, e.g.:
"
...
[SWS_EthIf_00213] ?
Caveat: The function requires previous initialization (EthIf_Init). ?()
"
Such requirment are not implementation relevant. Please remove this and add as note.
–Last change on issue 75637 comment 2–

Agreed solution:

General:
- Remove requirements which state that previous initialization of <Modul>_Init() is required, which are formulated like a note, e.g. : [SWS_EthIf_00213] ?
Caveat: The function requires previous initialization (EthIf_Init). ?()
- Add general requirement as note to ch. 8.3 Function definitions
- Remove or reformulate all other requirements, which start with caveat. Most of them are simply a note or implementation hint.

=== EhtIf ===
~8.3 Function definitions
+Note: All functions in this chapter requires previous initialization (EthIf_Init), except the following ones:
EthIf_Init, EthIf_GetVersionInfo

~8.3.2 EthIf_SetControllerMode
-[SWS_EthIf_00038]

~8.3.3 EthIf_GetControllerMode
-[SWS_EthIf_00044]

~8.3.8 EthIf_ CheckWakeup
-[SWS_EthIf_00249]

~8.3.9 EthIf_GetPhysAddr
-[SWS_EthIf_00066]

~8.3.10 EthIf_SetPhysAddr
-[SWS_EthIf_00138]

~8.3.11 EthIf_UpdatePhysAddrFilter
-[SWS_EthIf_00144]

~8.3.12 EthIf_GetPortMacAddr
-[SWS_EthIf_00195]

~8.3.13 EthIf_GetArlTable
-[SWS_EthIf_00201]

8.3.14 EthIf_GetBufferLevel
-[SWS_EthIf_00207]

~8.3.15 EthIf_GetDropCount
-[SWS_EthIf_00213]

~8.3.16 EthIf_StoreConfiguration
-[SWS_EthIf_00218]

~8.3.17 EthIf_ResetConfiguration
-[SWS_EthIf_00223]

8.3.18 EthIf_GetCurrentTime
-[SWS_EthIf_00159]

~8.3.19 EthIf_EnableEgressTimeStamp
-[SWS_EthIf_00165]

~8.3.20 EthIf_GetEgressTimeStamp
-[SWS_EthIf_00171]

~8.3.21 EthIf_GetIngressTimeStamp
-[SWS_EthIf_00177]

~8.3.22 EthIf_SetCorrectionTime
-[SWS_EthIf_00183]

~8.3.23 EthIf_SetGlobalTime

-[SWS_EthIf_00189]

~8.3.24 EthIf_ProvideTxBuffer
-[SWS_EthIf_00074]

~8.3.25 EthIf_Transmit
-[SWS_EthIf_00081]

=== CanIf ===
General note for below items which contain phrases like "The CAN Interface module must be initialized after Power ON": this is not needed anymore cause already stated in SWS_CANIF_00661. So e.g. in cases where result from CanDrv is just passed through the whole requirement can be deleted.

Also it is assumed, that the mindset like defined in SWS_CANIF_00661 is there for all upper layer functions like <User_TxConfirmation> which might be called by CanIf.

Because if CanDrv would not be already initialised it would return E_NOT_OK which is passed through:
-SWS_CANIF_00312
-SWS_CANIF_00316

Because already guarded by SWS_CANIF_00661:
-SWS_CANIF_00334
-SWS_CANIF_00339
-SWS_CANIF_00344
-SWS_CANIF_00349

~SWS_CANIF_00661
All CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall not execute their normal operation and return E_NOT_OK unless the CanIf has been initialized with a preceding call of CanIf_Init().

+SWS_CANIF_????1
If the switch CANIF_PUBLIC_DEV_ERROR_DETECT is enabled, all CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall report to the DET

(using CANIF_E_UNINIT) unless the CanIf has been initialized with a preceding call of CanIf_Init().

-SWS_CANIF_00323 change to
Note: During the call of CanIf_Transmit() the buffer of PduInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00329
CanIf_ReadRxPduData() shall not be used for CanIfRxSduId, which are defined to receive multiple CAN-Ids (range reception).

Add note below SWS_CANIF_00329
Note: During the call of of CanIf_ReadRxPduData() the buffer of CanIfRxInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00356
CanIf_SetDynamicTxId() shall not be interrupted by CanIf_Transmit(), if the same L-SDU ID is handled.

-SWS_CANIF_00407 change to
Note: The call context of CanIf_CheckValidation() is either on interrupt level (interrupt mode) or on task level (polling mode).
Caveat: The corresponding CAN controller and transceiver must be switched on via CanTrcv_SetOpMode(Transceiver, CANTRCV_TRCVMODE_NORMAL) and Can_SetControllerMode(Controller, CAN_CS_STARTED) and the corresponding mode indications must have been called.

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
-SWS_CANIF_00737 CanIf_GetTxConfirmationState()
-SWS_CANIF_00870 CanIf_SetBaudrate()

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
-SWS_CANIF_00401 CanIf_CheckWakeup()
-SWS_CANIF_00413 CanIf_TxConfirmation()
-SWS_CANIF_00422 CanIf_RxIndication()
-SWS_CANIF_00432 CanIf_ControllerBusOff()
-SWS_CANIF_00818 CanIf_ConfirmPnAvailability()
-SWS_CANIF_00807 CanIf_ClearTrcvWufFlagIndication()
-SWS_CANIF_00811 CanIf_CheckTrcvWakeFlagIndication()
-SWS_CANIF_00703 CanIf_ControllerModeIndication()
-SWS_CANIF_00709 CanIf_TrcvModeIndication()
-SWS_CANIF_00887 <User_TriggerTransmit>()
-SWS_CANIF_00437 <User_TxConfirmation>()

-SWS_CANIF_00440 change to
Note: Until <User_RxIndication>() returns, CanIf will not access PduInfoPtr. The PduInfoPtr is only valid and can be used by upper layers, until the indication returns. CanIf guarantees that the number of configured bytes for this PduInfoPtr is valid.
Note: The call context of <User_RxIndication>() is either on interrupt level (interrupt mode) or on task level (polling mode).

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00822 <User_ConfirmPnAvailability>()
-SWS_CANIF_00793 <User_ClearTrcvWufFlagIndication>()
-SWS_CANIF_00799 <User_CheckTrcvWakeFlagIndication>()

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"

-SWS_CANIF_00688 <User_ControllerModeIndication>()

=== ComM ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (ComM_Init), except the following ones:
ComM_Init, ComM_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the ComM module is initialized correctly.
-[SWS_ComM_00805] Caveats of ComM_Nm_NetworkStartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00806] Caveats of ComM_Nm_NetworkMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00808] Caveats of ComM_Nm_PrepareBusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00810] Caveats of ComM_Nm_BusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00812] Caveats of ComM_Nm_RestartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00814] Caveats of ComM_EcuM_WakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00965] Caveats of ComM_EcuM_PNCWakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00816] Caveats of ComM_BusSM_ModeIndication(): The Communication Manager Module is initialized correctly.()

=== LdCom ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (LdCom_Init), except the following ones:
LdCom_Init, LdCom_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the LdCom module is initialized correctly.
–Last change on issue 75637 comment 23–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.54 Specification Item SWS_EthIf_00222

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZEDUNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification "belongs".

  *** BSW UML Model ***
  SWS_CanNm:
  ———-
  Chapter 8.6.1 Optional Interfaces:
  Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

  SWS_LinIf:
  ———-
  SWS_LinIf_00359: add Det_ReportRuntimeError

  SWS_UdpNm:
  ———-

Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.55 Specification Item SWS_EthIf_00223

Trace References:

none

Content:

Caveat: The function requires previous initialization (EthIf_Init).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #75637: [diverse] Remove requirements like initializiation requirements regarding the <Module>_Init API

  Problem description:

  Generally a Caveat can never start a requirement. For this all basic Software module shall be checked and cleaned up for this formulation.

  original finding was:

  According to the discussion of RfC# 67678 we detect requirements regarding the initializiation of the certain api's, e.g.:
  "
  ...
  [SWS_EthIf_00213] ?
  Caveat: The function requires previous initialization (EthIf_Init). ?()
  "
  Such requirment are not implementation relevant. Please remove this and add as

note.
–Last change on issue 75637 comment 2–

Agreed solution:

General:
- Remove requirements which state that previous initialization of <Modul>_Init() is required, which are formulated like a note, e.g. : [SWS_EthIf_00213] ?
Caveat: The function requires previous initialization (EthIf_Init). ?()
- Add general requirement as note to ch. 8.3 Function definitions
- Remove or reformulate all other requirements, which start with caveat. Most of them are simply a note or implementation hint.

=== EhtIf ===
~8.3 Function definitions
+Note: All functions in this chapter requires previous initialization (EthIf_Init), except the following ones:
EthIf_Init, EthIf_GetVersionInfo

~8.3.2 EthIf_SetControllerMode
-[SWS_EthIf_00038]

~8.3.3 EthIf_GetControllerMode
-[SWS_EthIf_00044]

~8.3.8 EthIf_ CheckWakeup
-[SWS_EthIf_00249]

~8.3.9 EthIf_GetPhysAddr
-[SWS_EthIf_00066]

~8.3.10 EthIf_SetPhysAddr
-[SWS_EthIf_00138]

~8.3.11 EthIf_UpdatePhysAddrFilter
-[SWS_EthIf_00144]

~8.3.12 EthIf_GetPortMacAddr
-[SWS_EthIf_00195]

~8.3.13 EthIf_GetArlTable
-[SWS_EthIf_00201]

8.3.14 EthIf_GetBufferLevel
-[SWS_EthIf_00207]

~8.3.15 EthIf_GetDropCount
-[SWS_EthIf_00213]

~8.3.16 EthIf_StoreConfiguration
-[SWS_EthIf_00218]

~8.3.17 EthIf_ResetConfiguration
-[SWS_EthIf_00223]

8.3.18 EthIf_GetCurrentTime
-[SWS_EthIf_00159]

~8.3.19 EthIf_EnableEgressTimeStamp
-[SWS_EthIf_00165]

~8.3.20 EthIf_GetEgressTimeStamp
-[SWS_EthIf_00171]

~8.3.21 EthIf_GetIngressTimeStamp
-[SWS_EthIf_00177]

~8.3.22 EthIf_SetCorrectionTime
-[SWS_EthIf_00183]

~8.3.23 EthIf_SetGlobalTime
-[SWS_EthIf_00189]

~8.3.24 EthIf_ProvideTxBuffer
-[SWS_EthIf_00074]

~8.3.25 EthIf_Transmit
-[SWS_EthIf_00081]

=== CanIf ===
General note for below items which contain phrases like "The CAN Interface module must be initialized after Power ON": this is not needed anymore cause already stated in SWS_CANIF_00661. So e.g. in cases where result from CanDrv is just passed through the whole requirement can be deleted.

Also it is assumed, that the mindset like defined in SWS_CANIF_00661 is there for all upper layer functions like <User_TxConfirmation> which might be called by CanIf.

Because if CanDrv would not be already initialised it would return E_NOT_OK which is passed through:
-SWS_CANIF_00312
-SWS_CANIF_00316

Because already guarded by SWS_CANIF_00661:
-SWS_CANIF_00334
-SWS_CANIF_00339
-SWS_CANIF_00344
-SWS_CANIF_00349

~SWS_CANIF_00661
All CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall not execute their normal operation and return E_NOT_OK unless the CanIf has been initialized with a preceding call of CanIf_Init().

+SWS_CANIF_????1
If the switch CANIF_PUBLIC_DEV_ERROR_DETECT is enabled, all CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall report to the DET (using CANIF_E_UNINIT) unless the CanIf has been initialized with a preceding call of CanIf_Init().

-SWS_CANIF_00323 change to
Note: During the call of CanIf_Transmit() the buffer of PduInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00329
CanIf_ReadRxPduData() shall not be used for CanIfRxSduId, which are defined to receive multiple CAN-Ids (range reception).

Add note below SWS_CANIF_00329
Note: During the call of of CanIf_ReadRxPduData() the buffer of CanIfRxInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00356
CanIf_SetDynamicTxId() shall not be interrupted by CanIf_Transmit(), if the same L-SDU ID is handled.

-SWS_CANIF_00407 change to
Note: The call context of CanIf_CheckValidation() is either on interrupt level (interrupt mode) or on task level (polling mode).
Caveat: The corresponding CAN controller and transceiver must be switched on via CanTrcv_SetOpMode(Transceiver, CANTRCV_TRCVMODE_NORMAL) and Can_SetControllerMode(Controller, CAN_CS_STARTED) and the corresponding mode indications must have been called.

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
-SWS_CANIF_00737 CanIf_GetTxConfirmationState()
-SWS_CANIF_00870 CanIf_SetBaudrate()

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
-SWS_CANIF_00401 CanIf_CheckWakeup()
-SWS_CANIF_00413 CanIf_TxConfirmation()
-SWS_CANIF_00422 CanIf_RxIndication()
-SWS_CANIF_00432 CanIf_ControllerBusOff()
-SWS_CANIF_00818 CanIf_ConfirmPnAvailability()
-SWS_CANIF_00807 CanIf_ClearTrcvWufFlagIndication()
-SWS_CANIF_00811 CanIf_CheckTrcvWakeFlagIndication()
-SWS_CANIF_00703 CanIf_ControllerModeIndication()
-SWS_CANIF_00709 CanIf_TrcvModeIndication()
-SWS_CANIF_00887 <User_TriggerTransmit>()
-SWS_CANIF_00437 <User_TxConfirmation>()

-SWS_CANIF_00440 change to
Note: Until <User_RxIndication>() returns, CanIf will not access PduInfoPtr. The PduInfoPtr is only valid and can be used by upper layers, until the indication returns. CanIf guarantees that the number of configured bytes for this PduInfoPtr is valid.
Note: The call context of <User_RxIndication>() is either on interrupt level (interrupt mode) or on task level (polling mode).

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00822 <User_ConfirmPnAvailability>()
-SWS_CANIF_00793 <User_ClearTrcvWufFlagIndication>()
-SWS_CANIF_00799 <User_CheckTrcvWakeFlagIndication>()

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00688 <User_ControllerModeIndication>()

=== ComM ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (ComM_Init), except the following ones:
ComM_Init, ComM_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the ComM module is initialized correctly.
-[SWS_ComM_00805] Caveats of ComM_Nm_NetworkStartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00806] Caveats of ComM_Nm_NetworkMode: The Communication

Manager Module is initialized correctly.()
-[SWS_ComM_00808] Caveats of ComM_Nm_PrepareBusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00810] Caveats of ComM_Nm_BusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00812] Caveats of ComM_Nm_RestartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00814] Caveats of ComM_EcuM_WakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00965] Caveats of ComM_EcuM_PNCWakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00816] Caveats of ComM_BusSM_ModeIndication(): The Communication Manager Module is initialized correctly.()

=== LdCom ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (LdCom_Init), except the following ones:
LdCom_Init, LdCom_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the LdCom module is initialized correctly.
–Last change on issue 75637 comment 23–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.56 Specification Item SWS_EthIf_00232

Trace References:

none

Content:

Service name:	EthIf_TrcvModeIndicationEthIf_TrcvModeIndication
Syntax:	void EthIf_TrcvModeIndication(uint8 TrcvIdx, EthTrcv_ModeType TrcvMode)
Service ID[hex]:	0x0f
Sync/Async:	Synchronous

Reentrancy:	Non Reentrant for the same CtrlIdx, reentrant for different	
Parameters (in):	TrcvIdxEthIf_TrcvModeIndication.TrcvIdx	Index of the Ethernet transceiver within the context of the Ethernet Interface
	TrcvModeEthIf_TrcvModeIndication.TrcvMode	Notified Ethernet transceiver mode
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	Called asynchronously when mode a mode change has been read out. Triggered by previous EthIf the function is triggered by previous call of EthTrcv_SetTransceiverMode call. Can it can directly be called within the trigger functionsfunction.	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76528: [EthTrcv] Description of mandatory API "EthIf_TrcvModeIndicaton" seem to be partly wrong

  Problem description:

  The description for the mandatory API "EthIf_TrcvModeIndication" says:
  "
  ...
  Triggered by Eth_SetTranceiverMode ...
  ...
  "
  I think this should changed to "Triggered by EthTrcv_SetTranceiverMode"

  Agreed solution:

  Change API name in the description of EthIf_TrcvModeIndication in requirement SWS_EthTrcv_00085
  from ... Eth_SetTransceiverMode ...
  to ... EthTrcv_SetTransceiverMode ...

  Also change the description in the EthIf
  ~SWS_EthIf_00232
  –Last change on issue 76528 comment 2–

  BW-C-Level:

Application	Specification	Bus
1	1	1

- RfC #77250: [EthIf][EthSwt][EthTrcv] EthSwt_SetSwitchPortMode() broken for ports without Ethernet transceiver

  Problem description:

SWS_EthSwt_00019 states that EthSwt_SetSwitchPortMode() shall (unconditionally) invoke EthTrcv_SetTransceiverMode() of the Ethernet Transceiver Driver.

This is bogus for the following two reasons:
1) a particular switch port does not necessarily have an associated Ethernet transceiver. - In that case EthSwtPortTrcvRef won't be present. This is probably the case for MAC2MAC links if EthSwtPortRole is present an either set to ETHSWT_HOST_PORT or to ETHSWT_UP_LINK_PORT.
2) Even if the particular switch port has an associated Ethernet transceiver, SWS_EthTrcv_00043 states that the EthTrcv shall invoke EthIf_TrcvModeIndication() as a response to EthTrcv_SetTransceiverMode().

IMHO 1) should be solved by extending SWS_EthSwt_00019 distinguishing between ports which are associated with a transceiver (i.e., EthSwtPortTrcvRef present) - in that case EthTrcv_SetTransceiverMode() should be called - and ports which are not associated with a transceiver (i.e., EthSwtPortTrcvRef is not present) - in that case something else has to happen and this needs to be specified (probably the switch port shall be internally just disabled - maybe this needs to happen in the first case as well?)

2) leads to the following call chain: EthIf_SwitchPortGroupRequestMode() -> EthSwt_SetSwitchPortMode() -> EthTrcv_SetTransceiverMode() -> EthIf_TrcvModeIndication(). Thus the indication actually bypasses the EthSwt which IMHO is extremely ugly. Additionally for switch ports without associated transceiver, the EthIf will never get a EthIf_TrcvModeIndication() ...

Agreed solution:

=== System Template ===
+ Add upstream mapping of EthSwtPortTrcvRef to CouplingPort.physicalLayerType

=== EthSwt ===

~ch. 7.2.4 Production Errors
move SWS_EthSwt_00113 to 7.2.5 Extended production Errors
~SWS_EthSwt_00113 change detection criteria to
Fail:When access to the Ethernet Switch fails the module shall report the extended production error with event status DEM_EVENT_STATUS_PREFAILED to DEM.
Pass:When access to the Ethernet Switch succeeds the module shall report the extended production error with event status DEM_EVENT_STATUS_PREPASSED to DEM.

+ SWS_EthSwt_xxxxx new ext. production Error
Error Name: ETHSWT_E_SYNCPORT2PHY

Short Description: Ethernet switch port and the referenced Ethernet transceiver are in contradicting modes.
Long description: While getting the Ethernet switch port mode, the Ethernet switch driver detected an inconsistent state between Ethernet switch port and the referenced Ethernet transceiver Mode.
Detection Criteria: Fail: When getting the Ethernet switch port mode together with the Ethernet transceiver mode and the mode of the two referenced modules was found
inconsistent the module shall report the extended production error with event status DEM_EVENT_STATUS_PREFAILED to DEM.
Pass:When getting the Ethernet switch port mode together with the Ethernet transceiver mode and the mode of the two referenced modules was found
consistent the module shall report the extended production error with event status DEM_EVENT_STATUS_PREPASSED to DEM.
all other attributer: N/A

~ ch. 8.3.2 EthSwt_SwitchInit
~[SWS_EthSwt_00016] change "production error" to "extended production error"

~ch. 8.3.3 EthSwt_SetSwitchPortMode
~[SWS_EthSwt_00019]The function EthSwt_SetSwitchPortMode() shall put the indexed port of the switch into the specified mode. If EthSwtPort references an EthTrcv then the function EthTrcv_SetTransceiverMode() of the Ethernet Transceiver Driver shall additionally be called with the corresponding transceiver mode.

+[SWS_EthSwt_xxxxx] When calling the function EthSwt_SetSwitchPortMode() with mode ETHTRCV_MODE_DOWN, the EthSwt shall disable the Ethernet switch port directly for reduction of power consumption, if it is possible.

+[SWS_EthSwt_xxxxx] When calling the function EthSwt_SetSwitchPortMode(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT_E_ACCESS and return E_NOT_OK, otherwise pass the extended production error ETHSWT_E_ACCESS and return E_OK.

+[SWS_EthSwt_xxxxxx] If EthSwtPort does not references an EthTrcv, EthSwt shall indicate a mode of the port by the API EthIf_SwitchPortModeIndication latest during the next EthSwt_MainFunction.(SRS_ETH_00118)

~ch. 10.1.3 EthSwtDemEventParameterRefs
+ SWS Item ECUC_EthSwt_xxxxx1 : Name ETHSWT_E_SYNCPORT2PHY

Parent Container EthSwtDemEventParameterRefs
Description Reference to the DemEventParameter which shall be issued when the error "Ethernet switch port and the referenced Ethernet transceiver are in contradicting modes" has occurred.
Multiplicity 0..1
Type Symbolic name reference to [DemEventParameter]
Post-Build Variant Multiplicity true
Post-Build Variant Value true
Multiplicity Configuration
Class Pre-compile time X VARIANT-PRE-COMPILE
Link time X VARIANT-LINK-TIME
Post-build time X VARIANT-POST-BUILD
Value Configuration Class
Pre-compile time X VARIANT-PRE-COMPILE
Link time X VARIANT-LINK-TIME
Post-build time X VARIANT-POST-BUILD
Scope / Dependency
scope: local

~ch. 10.1.5 EthSwtPort
~ECUC_EthSwt_00041 EthSwtPortTrcvRef change following attributes:
Multiplicity Configuration Class | Link time | VARIANT-LINK-TIME, VARIANT-POST-BUILD
| Post build | –
add to dependency: If EthSwtPortPhysicalLayerType is defined, then EthSwtPortTrcvRef holds the reference to the corresponding EthTrcv.

~ECUC_EthSwt_00054 add dependency to EthSwtPortPhysicalLayerType: If a EthSwtPort has an EthSwtPortPhysicalLayerType then EthSwtPort shall reference an EthTrcv.

=== EthIf ===
Add to chapter 8.4 (callback notifications):
+ EthIf_SwitchPortModeIndication
Syntax: void EthIf_SwitchPortModeIndication(uint8 SwitchIdx, uint8 SwitchPortIdx, EthSwt_ModeType PortMode)
Sevice ID: pick a free one
Sync/Async: Synchronous
Reentrancy: Non Reentrant
Parmaters in: SwitchIdx: Index of the switch within the context of the Ethernet Switch Driver
SwitchPortIdx: Index of the port at the addressed switch
PortMode: notified Ethernet Switch port mode.

Description: The EthIf shall determine the expected notifications based on the EthSwtPort configuration. In case the EthSwtPort references an EthTrcv the EthIf expects a notification from the EthTrcv via API EthIf_TrcvModeIndication(). Otherwise the EthIf expects a notification from the EthSwt via API EthIf_SwitchPortModeIndication().

~SWS_EthIf_00232
change from
Called asynchronously when mode has been read out. Triggered by previous
EthTrcv_SetTransceiverMode call. Can directly be called within the trigger
functions.
To
Called asynchronously when a mode change has been read out. If the function is
triggered by previous call of
EthTrcv_SetTransceiverMode it can directly be called within the trigger function.
–Last change on issue 77250 comment 50–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.57 Specification Item SWS_EthIf_00235

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZED UNINIT otherwise (if DET is disabled) return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

Notes:
- It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
- The review task of the ITs shall be done by the WP to which the specification "belongs".

*** BSW UML Model ***
SWS_CanNm:
———-
Chapter 8.6.1 Optional Interfaces:
Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

SWS_LinIf:
———-
SWS_LinIf_00359: add Det_ReportRuntimeError

SWS_UdpNm:
———-
Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.58 Specification Item SWS_EthIf_00240

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZED UNINIT otherwise (if DET is disabled) return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification "belongs".

  *** BSW UML Model ***
  SWS_CanNm:
  ———-
  Chapter 8.6.1 Optional Interfaces:
  Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

  SWS_LinIf:
  ———-
  SWS_LinIf_00359: add Det_ReportRuntimeError

  SWS_UdpNm:
  ———-
  Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.59 Specification Item SWS_EthIf_00242

Trace References:

none

Content:

If development error detection is enabled: the function shall check the parameter TrcvWakeupModePtr for being valid. If the check fails, the function shall raise the development error ETHIF_E_INVPARAM_POINTER otherwise (if DET is disabled) return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76524: [EthIf][EthTrcv] missing definition for ETHIF_E_INV_POINTER / ETHTRCV_E_INV_POINTER

  Problem description:

  ETHIF_E_INV_POINTER appears in [SWS_EthIf_00242] [SWS_EthIf_00323] [SWS_EthIf_00333] [SWS_EthIf_00337] [SWS_EthIf_00338] [SWS_EthIf_00339].

  However, there's no definition for ETHIF_E_INV_POINTER.

  As default error list ([SWS_EthIf_00017]) already have ETHIF_E_PARAM_POINTER 0x05 (I believe it fits to the purpose of ETHIF_E_INV_POINTER), could you consider to replace ETHIF_E_INV_POINTER with ETHIF_E_PARAM_POINTER, please?

  Also, same issue exists at EthTrcv (ETHTRCV_E_INV_POINTER not defined at [SWS_EthTrcv_00131], ETHTRCV_E_PARAM_POINTER exists).

  Note:

- EthSwt has ETHSWT_E_INV_POINTER, but no ETHSWT_E_PARAM_POINTER (different default error identifier style has been applied, is handled by RfC # 76195).
- WdgIf has both WDGIF_E_INV_POINTER (this is used) and WDGIF_E_PARAM_POINTER (added by RfC # 59818 and RfC # 62544), but WDGIF_E_PARAM_POINTER is not used in the SWS (as mentioned in 62544).
- Xcp @ R4.2.2 has both XCP_E_INV_POINTER (this is used) and XCP_E_PARAM_POINTER, but XCP_E_PARAM_POINTER is not used in the SWS.
- Xcp @ R4.3.0 has both XCP_E_PARAM_POINTER (this is used) and XCP_E_INV_POINTER, but XCP_E_INV_POINTER is not used in the SWS. (changed from R4.2.2)

Remarks:
Even though there're two similar errors, <MA>_E_INV_POINTER and <MA>_E_PARAM_POINTER, I think there's no strong need to replace one of them with other. It's sufficient if this "undefined" issue was resolved, I think.
–Last change on issue 76524 comment 6–

Agreed solution:

SWS EthIf
==
replace ETHIF_E_INV_POINTER by ETHIF_E_PARAM_POINTER

SWS EthTrcv
==
replace ETHTRCV_E_INV_POINTER by ETHTRCV_E_PARAM_POINTER

SWS Xcp
==
remove XCP_E_INV_POINTER from the table [SWS_Xcp_00857]
–Last change on issue 76524 comment 8–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.60 Specification Item SWS_EthIf_00246

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZED UNINIT otherwise (if DET is disabled) return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification "belongs".

  *** BSW UML Model ***
  SWS_CanNm:
  ———-
  Chapter 8.6.1 Optional Interfaces:
  Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

  SWS_LinIf:
  ———-
  SWS_LinIf_00359: add Det_ReportRuntimeError

  SWS_UdpNm:
  ———-
  Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.61 Specification Item SWS_EthIf_00249

Trace References:

SRS_Eth_00106

Content:

Caveat: The function EthIf_CheckWakeup() requires previous transceiver initialization (EthIf_Init).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #75637: [diverse] Remove requirements like initializiation requirements regarding the <Module>_Init API

  Problem description:

  Generally a Caveat can never start a requirement. For this all basic Software module shall be checked and cleaned up for this formulation.

  original finding was:

  According to the discussion of RfC# 67678 we detect requirements regarding the initializiation of the certain api's, e.g.:
  "
  ...
  [SWS_EthIf_00213] ?
  Caveat: The function requires previous initialization (EthIf_Init). ?()
  "
  Such requirment are not implementation relevant. Please remove this and add as note.
  –Last change on issue 75637 comment 2–

  Agreed solution:

General:
- Remove requirements which state that previous initialization of <Modul>_Init() is required, which are formulated like a note, e.g. : [SWS_EthIf_00213] ?
Caveat: The function requires previous initialization (EthIf_Init). ?()
- Add general requirement as note to ch. 8.3 Function definitions
- Remove or reformulate all other requirements, which start with caveat. Most of them are simply a note or implementation hint.

=== EhtIf ===
~8.3 Function definitions
+Note: All functions in this chapter requires previous initialization (EthIf_Init), except the following ones:
EthIf_Init, EthIf_GetVersionInfo

~8.3.2 EthIf_SetControllerMode
-[SWS_EthIf_00038]

~8.3.3 EthIf_GetControllerMode
-[SWS_EthIf_00044]

~8.3.8 EthIf_ CheckWakeup
-[SWS_EthIf_00249]

~8.3.9 EthIf_GetPhysAddr
-[SWS_EthIf_00066]

~8.3.10 EthIf_SetPhysAddr
-[SWS_EthIf_00138]

~8.3.11 EthIf_UpdatePhysAddrFilter
-[SWS_EthIf_00144]

~8.3.12 EthIf_GetPortMacAddr
-[SWS_EthIf_00195]

~8.3.13 EthIf_GetArlTable
-[SWS_EthIf_00201]

8.3.14 EthIf_GetBufferLevel
-[SWS_EthIf_00207]

~8.3.15 EthIf_GetDropCount

-[SWS_EthIf_00213]

~8.3.16 EthIf_StoreConfiguration
-[SWS_EthIf_00218]

~8.3.17 EthIf_ResetConfiguration
-[SWS_EthIf_00223]

8.3.18 EthIf_GetCurrentTime
-[SWS_EthIf_00159]

~8.3.19 EthIf_EnableEgressTimeStamp
-[SWS_EthIf_00165]

~8.3.20 EthIf_GetEgressTimeStamp
-[SWS_EthIf_00171]

~8.3.21 EthIf_GetIngressTimeStamp
-[SWS_EthIf_00177]

~8.3.22 EthIf_SetCorrectionTime
-[SWS_EthIf_00183]

~8.3.23 EthIf_SetGlobalTime
-[SWS_EthIf_00189]

~8.3.24 EthIf_ProvideTxBuffer
-[SWS_EthIf_00074]

~8.3.25 EthIf_Transmit
-[SWS_EthIf_00081]

=== CanIf ===
General note for below items which contain phrases like "The CAN Interface module must be initialized after Power ON": this is not needed anymore cause already stated in SWS_CANIF_00661. So e.g. in cases where result from CanDrv is just passed through the whole requirement can be deleted.

Also it is assumed, that the mindset like defined in SWS_CANIF_00661 is there for all upper layer functions like <User_TxConfirmation> which might be called by CanIf.

Because if CanDrv would not be already initialised it would return E_NOT_OK which is passed through:
-SWS_CANIF_00312
-SWS_CANIF_00316

Because already guarded by SWS_CANIF_00661:
-SWS_CANIF_00334
-SWS_CANIF_00339
-SWS_CANIF_00344
-SWS_CANIF_00349

~SWS_CANIF_00661
All CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall not execute their normal operation and return E_NOT_OK unless the CanIf has been initialized with a preceding call of CanIf_Init().

+SWS_CANIF_????1
If the switch CANIF_PUBLIC_DEV_ERROR_DETECT is enabled, all CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall report to the DET (using CANIF_E_UNINIT) unless the CanIf has been initialized with a preceding call of CanIf_Init().

-SWS_CANIF_00323 change to
Note: During the call of CanIf_Transmit() the buffer of PduInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00329
CanIf_ReadRxPduData() shall not be used for CanIfRxSduId, which are defined to receive multiple CAN-Ids (range reception).

Add note below SWS_CANIF_00329
Note: During the call of of CanIf_ReadRxPduData() the buffer of CanIfRxInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper

layer.

~SWS_CANIF_00356
CanIf_SetDynamicTxId() shall not be interrupted by CanIf_Transmit(), if the same L-SDU ID is handled.

-SWS_CANIF_00407 change to
Note: The call context of CanIf_CheckValidation() is either on interrupt level (interrupt mode) or on task level (polling mode).
Caveat: The corresponding CAN controller and transceiver must be switched on via CanTrcv_SetOpMode(Transceiver, CANTRCV_TRCVMODE_NORMAL) and Can_SetControllerMode(Controller, CAN_CS_STARTED) and the corresponding mode indications must have been called.

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
-SWS_CANIF_00737 CanIf_GetTxConfirmationState()
-SWS_CANIF_00870 CanIf_SetBaudrate()

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
-SWS_CANIF_00401 CanIf_CheckWakeup()
-SWS_CANIF_00413 CanIf_TxConfirmation()
-SWS_CANIF_00422 CanIf_RxIndication()
-SWS_CANIF_00432 CanIf_ControllerBusOff()
-SWS_CANIF_00818 CanIf_ConfirmPnAvailability()
-SWS_CANIF_00807 CanIf_ClearTrcvWufFlagIndication()
-SWS_CANIF_00811 CanIf_CheckTrcvWakeFlagIndication()
-SWS_CANIF_00703 CanIf_ControllerModeIndication()
-SWS_CANIF_00709 CanIf_TrcvModeIndication()
-SWS_CANIF_00887 <User_TriggerTransmit>()
-SWS_CANIF_00437 <User_TxConfirmation>()

-SWS_CANIF_00440 change to
Note: Until <User_RxIndication>() returns, CanIf will not access PduInfoPtr. The

PduInfoPtr is only valid and can be used by upper layers, until the indication returns.
CanIf guarantees that the number of configured bytes for this PduInfoPtr is valid.
Note: The call context of <User_RxIndication>() is either on interrupt level (interrupt mode) or on task level (polling mode).

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00822 <User_ConfirmPnAvailability>()
-SWS_CANIF_00793 <User_ClearTrcvWufFlagIndication>()
-SWS_CANIF_00799 <User_CheckTrcvWakeFlagIndication>()

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00688 <User_ControllerModeIndication>()

=== ComM ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (ComM_Init), except the following ones:
ComM_Init, ComM_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the ComM module is initialized correctly.
-[SWS_ComM_00805] Caveats of ComM_Nm_NetworkStartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00806] Caveats of ComM_Nm_NetworkMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00808] Caveats of ComM_Nm_PrepareBusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00810] Caveats of ComM_Nm_BusSleepMode: The Communica-

tion Manager Module is initialized correctly.()
-[SWS_ComM_00812] Caveats of ComM_Nm_RestartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00814] Caveats of ComM_EcuM_WakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00965] Caveats of ComM_EcuM_PNCWakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00816] Caveats of ComM_BusSM_ModeIndication(): The Communication Manager Module is initialized correctly.()

=== LdCom ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (LdCom_Init), except the following ones:
LdCom_Init, LdCom_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the LdCom module is initialized correctly.
–Last change on issue 75637 comment 23–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.62 Specification Item SWS_EthIf_00257

Trace References:

none

Content:

If no EthIfSwitchPortGroup is configured, all EthSwtPorts belonging to a switch shall be switched on if a least one EthIfController referencing this switch is requested with ETH_MODE_ACTIVE.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76490: [EthSwt] Clarification for activating / deactivating ports when no port groups are defined

  Problem description:

Within WP-A2 Meeting today we found the following item:

There are two requirements talking about activation and deactivation of ports in case of no switch port groups are defined:

[SWS_EthIf_00257] If no EthIfSwitchPortGroup is configured, all EthSwtPorts shall be switched on if a least one EthIfController is requested with ETH_MODE_ACTIVE.

[SWS_EthIf_00258] If no EthIfSwitchPortGroup is configured, all EthSwtPorts shall be switched off if all EthIfController are requested with ETH_MODE_DOWN.

What is not explicitly mentioned is that this makes only sense for EthIfController referencing a switch.

Agreed solution:

Change [SWS_EthIf_00257] to
"If no EthIfSwitchPortGroup is configured, all EthSwtPorts belonging to a switch shall be switched on if a least one EthIfController referencing this switch is requested with ETH_MODE_ACTIVE.

Change [SWS_EthIf_00258] to
"If no EthIfSwitchPortGroup is configured, all EthSwtPorts belonging to a switch shall be switched off if all EthIfController referencing this switch are requested with ETH_MODE_DOWN."

BW-C-Level:

Application	Specification	Bus
1	1	1

1.63 Specification Item SWS_EthIf_00258

Trace References:

none

Content:

If no EthIfSwitchPortGroup is configured, all EthSwtPorts belonging to a switch shall be switched off if all EthIfController referencing this switch are requested with ETH_MODE_DOWN.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76490: [EthSwt] Clarification for activating / deactivating ports when no port groups are defined

  Problem description:

  Within WP-A2 Meeting today we found the following item:

  There are two requirements talking about activation and deactivation of ports in case of no switch port groups are defined:

  [SWS_EthIf_00257] If no EthIfSwitchPortGroup is configured, all EthSwtPorts shall be switched on if a least one EthIfController is requested with ETH_MODE_ACTIVE.

  [SWS_EthIf_00258] If no EthIfSwitchPortGroup is configured, all EthSwtPorts shall be switched off if all EthIfController are requested with ETH_MODE_DOWN.

  What is not explicitly mentioned is that this makes only sense for EthIfController referencing a switch.

  Agreed solution:

  Change [SWS_EthIf_00257] to
  "If no EthIfSwitchPortGroup is configured, all EthSwtPorts belonging to a switch shall be switched on if a least one EthIfController referencing this switch is requested with ETH_MODE_ACTIVE.

  Change [SWS_EthIf_00258] to
  "If no EthIfSwitchPortGroup is configured, all EthSwtPorts belonging to a switch shall be switched off if all EthIfController referencing this switch are requested with ETH_MODE_DOWN."

  BW-C-Level:

Application	Specification	Bus
1	1	1

1.64 Specification Item SWS_EthIf_00261

Trace References:

none

Content:

In case a EthIfSwitchPortGroup is not connected to any EthIfController, the EthIf shall indicate the accumulated link state of the EthIfSwitchPortGroup to the BswM by calling BswM_EthIf_PortGroupLinkStateChg for the EthIfSwitchPortGroup when the link state changes (refer to SWS_EthIf_xxx21 00259 for link state accumulation).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76496: [EthIf] incorrect reference @ SWS_EthIf_00261

  Problem description:

  Incorrect reference "SWS_EthIf_xxx21" exists at [SWS_EthIf_00261].
  –Last change on issue 76496 comment 1–

  Agreed solution:

  proposed solution at [SWS_EthIf_00261]
  changed from
  < (refer to SWS_EthIf_xxx21 for link state accumulation)
  to
  > (refer to SWS_EthIf_00259 for link state accumulation)
  –Last change on issue 76496 comment 1–

  BW-C-Level:

Application	Specification	Bus
1	1	1

1.65 Specification Item SWS_EthIf_00273

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZED UNINIT and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

Inconsistencies in SWS with semantics of Default errors
–Last change on issue 59085 comment 26–

Agreed solution:

solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

Notes:
- It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
- The review task of the ITs shall be done by the WP to which the specification "belongs".

*** BSW UML Model ***
SWS_CanNm:
———-
Chapter 8.6.1 Optional Interfaces:
Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

SWS_LinIf:
———-
SWS_LinIf_00359: add Det_ReportRuntimeError

SWS_UdpNm:
———-
Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.66 Specification Item SWS_EthIf_00277

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZED UNINIT and return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification "belongs".

  *** BSW UML Model ***
  SWS_CanNm:
  ———-
  Chapter 8.6.1 Optional Interfaces:
  Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

  SWS_LinIf:
  ———-
  SWS_LinIf_00359: add Det_ReportRuntimeError

  SWS_UdpNm:
  ———-

Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.67 Specification Item SWS_EthIf_00278

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZEDUNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification "belongs".

*** BSW UML Model ***
SWS_CanNm:
———-
Chapter 8.6.1 Optional Interfaces:
Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

SWS_LinIf:
———-
SWS_LinIf_00359: add Det_ReportRuntimeError

SWS_UdpNm:
———-
Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.68 Specification Item SWS_EthIf_00280

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init() was previously called.

If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZEDUNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification "belongs".

  *** BSW UML Model ***
  SWS_CanNm:
  ———-
  Chapter 8.6.1 Optional Interfaces:
  Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

  SWS_LinIf:
  ———-
  SWS_LinIf_00359: add Det_ReportRuntimeError

  SWS_UdpNm:
  ———-
  Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

  *** ECUC XML ***
  Not affected. No configuration of runtime error reporting required (see SWS BSW General).
  –Last change on issue 59085 comment 88–

  BW-C-Level:

Application	Specification	Bus
1	4	1

1.69 Specification Item SWS_EthIf_00284

Trace References:

none

Content:

Caveat: The function requires previous Switch initialization (EthSwt_Init()).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #75637: [diverse] Remove requirements like initializiation requirements regarding the <Module>_Init API

  Problem description:

  Generally a Caveat can never start a requirement. For this all basic Software module shall be checked and cleaned up for this formulation.

  original finding was:

  According to the discussion of RfC# 67678 we detect requirements regarding the initializiation of the certain api's, e.g.:
  "
  ...
  [SWS_EthIf_00213] ?
  Caveat: The function requires previous initialization (EthIf_Init). ?()
  "
  Such requirment are not implementation relevant. Please remove this and add as note.
  –Last change on issue 75637 comment 2–

  Agreed solution:

  General:
  - Remove requirements which state that previous initialization of <Modul>_Init() is required, which are formulated like a note, e.g. : [SWS_EthIf_00213] ?
  Caveat: The function requires previous initialization (EthIf_Init). ?()
  - Add general requirement as note to ch. 8.3 Function definitions
  - Remove or reformulate all other requirements, which start with caveat. Most of them are simply a note or implementation hint.

  === EhtIf ===
  ~8.3 Function definitions

+Note: All functions in this chapter requires previous initialization (EthIf_Init), except the following ones:
EthIf_Init, EthIf_GetVersionInfo

~8.3.2 EthIf_SetControllerMode
-[SWS_EthIf_00038]

~8.3.3 EthIf_GetControllerMode
-[SWS_EthIf_00044]

~8.3.8 EthIf_ CheckWakeup
-[SWS_EthIf_00249]

~8.3.9 EthIf_GetPhysAddr
-[SWS_EthIf_00066]

~8.3.10 EthIf_SetPhysAddr
-[SWS_EthIf_00138]

~8.3.11 EthIf_UpdatePhysAddrFilter
-[SWS_EthIf_00144]

~8.3.12 EthIf_GetPortMacAddr
-[SWS_EthIf_00195]

~8.3.13 EthIf_GetArlTable
-[SWS_EthIf_00201]

8.3.14 EthIf_GetBufferLevel
-[SWS_EthIf_00207]

~8.3.15 EthIf_GetDropCount
-[SWS_EthIf_00213]

~8.3.16 EthIf_StoreConfiguration
-[SWS_EthIf_00218]

~8.3.17 EthIf_ResetConfiguration
-[SWS_EthIf_00223]

8.3.18 EthIf_GetCurrentTime
-[SWS_EthIf_00159]

~8.3.19 EthIf_EnableEgressTimeStamp
-[SWS_EthIf_00165]

~8.3.20 EthIf_GetEgressTimeStamp
-[SWS_EthIf_00171]

~8.3.21 EthIf_GetIngressTimeStamp
-[SWS_EthIf_00177]

~8.3.22 EthIf_SetCorrectionTime
-[SWS_EthIf_00183]

~8.3.23 EthIf_SetGlobalTime
-[SWS_EthIf_00189]

~8.3.24 EthIf_ProvideTxBuffer
-[SWS_EthIf_00074]

~8.3.25 EthIf_Transmit
-[SWS_EthIf_00081]

=== CanIf ===
General note for below items which contain phrases like "The CAN Interface module must be initialized after Power ON": this is not needed anymore cause already stated in SWS_CANIF_00661. So e.g. in cases where result from CanDrv is just passed through the whole requirement can be deleted.

Also it is assumed, that the mindset like defined in SWS_CANIF_00661 is there for all upper layer functions like <User_TxConfirmation> which might be called by CanIf.

Because if CanDrv would not be already initialised it would return E_NOT_OK which is passed through:
-SWS_CANIF_00312
-SWS_CANIF_00316

Because already guarded by SWS_CANIF_00661:
-SWS_CANIF_00334
-SWS_CANIF_00339

-SWS_CANIF_00344
-SWS_CANIF_00349

~SWS_CANIF_00661
All CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall not execute their normal operation and return E_NOT_OK unless the CanIf has been initialized with a preceding call of CanIf_Init().

+SWS_CANIF_????1
If the switch CANIF_PUBLIC_DEV_ERROR_DETECT is enabled, all CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall report to the DET (using CANIF_E_UNINIT) unless the CanIf has been initialized with a preceding call of CanIf_Init().

-SWS_CANIF_00323 change to
Note: During the call of CanIf_Transmit() the buffer of PduInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00329
CanIf_ReadRxPduData() shall not be used for CanIfRxSduId, which are defined to receive multiple CAN-Ids (range reception).

Add note below SWS_CANIF_00329
Note: During the call of of CanIf_ReadRxPduData() the buffer of CanIfRxInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00356
CanIf_SetDynamicTxId() shall not be interrupted by CanIf_Transmit(), if the same L-SDU ID is handled.

-SWS_CANIF_00407 change to
Note: The call context of CanIf_CheckValidation() is either on interrupt level (interrupt mode) or on task level (polling mode).

Caveat: The corresponding CAN controller and transceiver must be switched on via CanTrcv_SetOpMode(Transceiver, CANTRCV_TRCVMODE_NORMAL) and Can_SetControllerMode(Controller, CAN_CS_STARTED) and the corresponding mode indications must have been called.

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
-SWS_CANIF_00737 CanIf_GetTxConfirmationState()
-SWS_CANIF_00870 CanIf_SetBaudrate()

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
-SWS_CANIF_00401 CanIf_CheckWakeup()
-SWS_CANIF_00413 CanIf_TxConfirmation()
-SWS_CANIF_00422 CanIf_RxIndication()
-SWS_CANIF_00432 CanIf_ControllerBusOff()
-SWS_CANIF_00818 CanIf_ConfirmPnAvailability()
-SWS_CANIF_00807 CanIf_ClearTrcvWufFlagIndication()
-SWS_CANIF_00811 CanIf_CheckTrcvWakeFlagIndication()
-SWS_CANIF_00703 CanIf_ControllerModeIndication()
-SWS_CANIF_00709 CanIf_TrcvModeIndication()
-SWS_CANIF_00887 <User_TriggerTransmit>()
-SWS_CANIF_00437 <User_TxConfirmation>()

-SWS_CANIF_00440 change to
Note: Until <User_RxIndication>() returns, CanIf will not access PduInfoPtr. The PduInfoPtr is only valid and can be used by upper layers, until the indication returns. CanIf guarantees that the number of configured bytes for this PduInfoPtr is valid.
Note: The call context of <User_RxIndication>() is either on interrupt level (interrupt mode) or on task level (polling mode).

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN

Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00822 <User_ConfirmPnAvailability>()
-SWS_CANIF_00793 <User_ClearTrcvWufFlagIndication>()
-SWS_CANIF_00799 <User_CheckTrcvWakeFlagIndication>()

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00688 <User_ControllerModeIndication>()

=== ComM ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (ComM_Init), except the following ones:
ComM_Init, ComM_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the ComM module is initialized correctly.
-[SWS_ComM_00805] Caveats of ComM_Nm_NetworkStartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00806] Caveats of ComM_Nm_NetworkMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00808] Caveats of ComM_Nm_PrepareBusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00810] Caveats of ComM_Nm_BusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00812] Caveats of ComM_Nm_RestartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00814] Caveats of ComM_EcuM_WakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00965] Caveats of ComM_EcuM_PNCWakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00816] Caveats of ComM_BusSM_ModeIndication(): The Communication Manager Module is initialized correctly.()

=== LdCom ===

~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (LdCom_Init), except the following ones:
LdCom_Init, LdCom_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the LdCom module is initialized correctly.
–Last change on issue 75637 comment 23–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.70 Specification Item SWS_EthIf_00286

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service Eth_Init() was previously called.

If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZEDUNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of ’Runtime errors’

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is

referring to that error and adapt them if needed.
- The review task of the ITs shall be done by the WP to which the specification "belongs".

*** BSW UML Model ***
SWS_CanNm:
———-
Chapter 8.6.1 Optional Interfaces:
Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

SWS_LinIf:
———-
SWS_LinIf_00359: add Det_ReportRuntimeError

SWS_UdpNm:
———-
Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.71 Specification Item SWS_EthIf_00292

Trace References:

none

Content:

Caveat: The function requires previous Switch initialization (EthSwt_Init()).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #75637: [diverse] Remove requirements like initializiation requirements regarding the <Module>_Init API

  Problem description:

  Generally a Caveat can never start a requirement. For this all basic Software module shall be checked and cleaned up for this formulation.

  original finding was:

  According to the discussion of RfC# 67678 we detect requirements regarding the initializiation of the certain api's, e.g.:
  "
  ...
  [SWS_EthIf_00213] ?
  Caveat: The function requires previous initialization (EthIf_Init). ?()
  "
  Such requirment are not implementation relevant. Please remove this and add as note.
  –Last change on issue 75637 comment 2–

  Agreed solution:

  General:
  - Remove requirements which state that previous initialization of <Modul>_Init() is required, which are formulated like a note, e.g. : [SWS_EthIf_00213] ?
  Caveat: The function requires previous initialization (EthIf_Init). ?()
  - Add general requirement as note to ch. 8.3 Function definitions
  - Remove or reformulate all other requirements, which start with caveat. Most of them are simply a note or implementation hint.

  === EhtIf ===
  ~8.3 Function definitions
  +Note: All functions in this chapter requires previous initialization (EthIf_Init), except the following ones:
  EthIf_Init, EthIf_GetVersionInfo

  ~8.3.2 EthIf_SetControllerMode
  -[SWS_EthIf_00038]

  ~8.3.3 EthIf_GetControllerMode
  -[SWS_EthIf_00044]

  ~8.3.8 EthIf_ CheckWakeup

-[SWS_EthIf_00249]

~8.3.9 EthIf_GetPhysAddr
-[SWS_EthIf_00066]

~8.3.10 EthIf_SetPhysAddr
-[SWS_EthIf_00138]

~8.3.11 EthIf_UpdatePhysAddrFilter
-[SWS_EthIf_00144]

~8.3.12 EthIf_GetPortMacAddr
-[SWS_EthIf_00195]

~8.3.13 EthIf_GetArlTable
-[SWS_EthIf_00201]

8.3.14 EthIf_GetBufferLevel
-[SWS_EthIf_00207]

~8.3.15 EthIf_GetDropCount
-[SWS_EthIf_00213]

~8.3.16 EthIf_StoreConfiguration
-[SWS_EthIf_00218]

~8.3.17 EthIf_ResetConfiguration
-[SWS_EthIf_00223]

8.3.18 EthIf_GetCurrentTime
-[SWS_EthIf_00159]

~8.3.19 EthIf_EnableEgressTimeStamp
-[SWS_EthIf_00165]

~8.3.20 EthIf_GetEgressTimeStamp
-[SWS_EthIf_00171]

~8.3.21 EthIf_GetIngressTimeStamp
-[SWS_EthIf_00177]

~8.3.22 EthIf_SetCorrectionTime
-[SWS_EthIf_00183]

~8.3.23 EthIf_SetGlobalTime
-[SWS_EthIf_00189]

~8.3.24 EthIf_ProvideTxBuffer
-[SWS_EthIf_00074]

~8.3.25 EthIf_Transmit
-[SWS_EthIf_00081]

=== CanIf ===
General note for below items which contain phrases like "The CAN Interface module must be initialized after Power ON": this is not needed anymore cause already stated in SWS_CANIF_00661. So e.g. in cases where result from CanDrv is just passed through the whole requirement can be deleted.

Also it is assumed, that the mindset like defined in SWS_CANIF_00661 is there for all upper layer functions like <User_TxConfirmation> which might be called by CanIf.

Because if CanDrv would not be already initialised it would return E_NOT_OK which is passed through:
-SWS_CANIF_00312
-SWS_CANIF_00316

Because already guarded by SWS_CANIF_00661:
-SWS_CANIF_00334
-SWS_CANIF_00339
-SWS_CANIF_00344
-SWS_CANIF_00349

~SWS_CANIF_00661
All CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall not execute their normal operation and return E_NOT_OK unless the CanIf has been initialized with a preceding call of CanIf_Init().

+SWS_CANIF_????1

If the switch CANIF_PUBLIC_DEV_ERROR_DETECT is enabled, all CanIf API services other than CanIf_Init() and CanIf_GetVersionInfo() shall report to the DET (using CANIF_E_UNINIT) unless the CanIf has been initialized with a preceding call of CanIf_Init().

-SWS_CANIF_00323 change to
Note: During the call of CanIf_Transmit() the buffer of PduInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00329
CanIf_ReadRxPduData() shall not be used for CanIfRxSduId, which are defined to receive multiple CAN-Ids (range reception).

Add note below SWS_CANIF_00329
Note: During the call of of CanIf_ReadRxPduData() the buffer of CanIfRxInfoPtr is controlled by CanIf and this buffer should not be accessed for read/write from another call context. After return of this call the ownership changes to the upper layer.

~SWS_CANIF_00356
CanIf_SetDynamicTxId() shall not be interrupted by CanIf_Transmit(), if the same L-SDU ID is handled.

-SWS_CANIF_00407 change to
Note: The call context of CanIf_CheckValidation() is either on interrupt level (interrupt mode) or on task level (polling mode).
Caveat: The corresponding CAN controller and transceiver must be switched on via CanTrcv_SetOpMode(Transceiver, CANTRCV_TRCVMODE_NORMAL) and Can_SetControllerMode(Controller, CAN_CS_STARTED) and the corresponding mode indications must have been called.

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."
-SWS_CANIF_00737 CanIf_GetTxConfirmationState()
-SWS_CANIF_00870 CanIf_SetBaudrate()

Remove all following requirements and instead add there a Note with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
-SWS_CANIF_00401 CanIf_CheckWakeup()
-SWS_CANIF_00413 CanIf_TxConfirmation()
-SWS_CANIF_00422 CanIf_RxIndication()
-SWS_CANIF_00432 CanIf_ControllerBusOff()
-SWS_CANIF_00818 CanIf_ConfirmPnAvailability()
-SWS_CANIF_00807 CanIf_ClearTrcvWufFlagIndication()
-SWS_CANIF_00811 CanIf_CheckTrcvWakeFlagIndication()
-SWS_CANIF_00703 CanIf_ControllerModeIndication()
-SWS_CANIF_00709 CanIf_TrcvModeIndication()
-SWS_CANIF_00887 <User_TriggerTransmit>()
-SWS_CANIF_00437 <User_TxConfirmation>()

-SWS_CANIF_00440 change to
Note: Until <User_RxIndication>() returns, CanIf will not access PduInfoPtr. The PduInfoPtr is only valid and can be used by upper layers, until the indication returns. CanIf guarantees that the number of configured bytes for this PduInfoPtr is valid.
Note: The call context of <User_RxIndication>() is either on interrupt level (interrupt mode) or on task level (polling mode).

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is either on interrupt level (interrupt mode) or on task level (polling mode)."
"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00455 <User_ValidateWakeupEvent>()
-SWS_CANIF_00822 <User_ConfirmPnAvailability>()
-SWS_CANIF_00793 <User_ClearTrcvWufFlagIndication>()
-SWS_CANIF_00799 <User_CheckTrcvWakeFlagIndication>()

Remove all following requirements and instead add there two Notes with template <FUNC>
"Note: The call context of <FUNC> is on task level (polling mode)."

"Note: The callback service <FUNC> is in general re-entrant for multiple CAN Controller usage, but not for the same CAN Controller"
-SWS_CANIF_00688 <User_ControllerModeIndication>()

=== ComM ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (ComM_Init), except the following ones:
ComM_Init, ComM_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the ComM module is initialized correctly.
-[SWS_ComM_00805] Caveats of ComM_Nm_NetworkStartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00806] Caveats of ComM_Nm_NetworkMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00808] Caveats of ComM_Nm_PrepareBusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00810] Caveats of ComM_Nm_BusSleepMode: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00812] Caveats of ComM_Nm_RestartIndication: The ComM module is initialized correctly.()
-[SWS_ComM_00814] Caveats of ComM_EcuM_WakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00965] Caveats of ComM_EcuM_PNCWakeUpIndication: The Communication Manager Module is initialized correctly.()
-[SWS_ComM_00816] Caveats of ComM_BusSM_ModeIndication(): The Communication Manager Module is initialized correctly.()

=== LdCom ===
~ ch. 8.3 Function definition
+Note: All functions in this chapter requires previous initialization (LdCom_Init), except the following ones:
LdCom_Init, LdCom_GetVersionInfo

~ ch. 8.4 Callback definition
+Note: All functions in this chapter requires that the LdCom module is initialized correctly.
–Last change on issue 75637 comment 23–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.72 Specification Item SWS_EthIf_00299

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZEDUNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of ’Runtime errors’

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification "belongs".

  *** BSW UML Model ***
  SWS_CanNm:
  ———-
  Chapter 8.6.1 Optional Interfaces:
  Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

  SWS_LinIf:
  ———-

SWS_LinIf_00359: add Det_ReportRuntimeError

SWS_UdpNm:
———-
Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.73 Specification Item SWS_EthIf_00302

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service Eth If_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZEDUNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another.

Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
- The review task of the ITs shall be done by the WP to which the specification "belongs".

*** BSW UML Model ***
SWS_CanNm:
———-
Chapter 8.6.1 Optional Interfaces:
Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

SWS_LinIf:
———-
SWS_LinIf_00359: add Det_ReportRuntimeError

SWS_UdpNm:
———-
Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.74 Specification Item SWS_EthIf_00304

Trace References:

SRS_BSW_00101

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZEDUNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification "belongs".

  *** BSW UML Model ***
  SWS_CanNm:
  ———-
  Chapter 8.6.1 Optional Interfaces:
  Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

  SWS_LinIf:
  ———-
  SWS_LinIf_00359: add Det_ReportRuntimeError

  SWS_UdpNm:
  ———-
  Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

  *** ECUC XML ***
  Not affected. No configuration of runtime error reporting required (see SWS BSW General).
  –Last change on issue 59085 comment 88–

  BW-C-Level:

Application	**Specification**	**Bus**
1	4	1

1.75 Specification Item SWS_EthIf_00306

Trace References:

SRS_BSW_00101

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZEDUNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification "belongs".

  *** BSW UML Model ***
  SWS_CanNm:
  ——–-
  Chapter 8.6.1 Optional Interfaces:
  Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

  SWS_LinIf:
  ——–-

SWS_LinIf_00359: add Det_ReportRuntimeError

SWS_UdpNm:
———-
Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.76 Specification Item SWS_EthIf_00321

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZED UNINIT otherwise (if DET is disabled) return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:

- It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
- The review task of the ITs shall be done by the WP to which the specification "belongs".

*** BSW UML Model ***
SWS_CanNm:
———-
Chapter 8.6.1 Optional Interfaces:
Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

SWS_LinIf:
———-
SWS_LinIf_00359: add Det_ReportRuntimeError

SWS_UdpNm:
———-
Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.77 Specification Item SWS_EthIf_00323

Trace References:

none

Content:

If development error detection is enabled: the function shall check the parameter Signal QualityPtr for being valid. If the check fails, the function shall raise the development error ETHIF_E_INVPARAM_POINTER otherwise (if DET is disabled) return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76524: [EthIf][EthTrcv] missing definition for ETHIF_E_INV_POINTER / ETHTRCV_E_INV_POINTER

  Problem description:

  ETHIF_E_INV_POINTER appears in [SWS_EthIf_00242] [SWS_EthIf_00323] [SWS_EthIf_00333] [SWS_EthIf_00337] [SWS_EthIf_00338] [SWS_EthIf_00339].

  However, there's no definition for ETHIF_E_INV_POINTER.

  As default error list ([SWS_EthIf_00017]) already have ETHIF_E_PARAM_POINTER 0x05 (I believe it fits to the purpose of ETHIF_E_INV_POINTER), could you consider to replace ETHIF_E_INV_POINTER with ETHIF_E_PARAM_POINTER, please?

  Also, same issue exists at EthTrcv (ETHTRCV_E_INV_POINTER not defined at [SWS_EthTrcv_00131], ETHTRCV_E_PARAM_POINTER exists).

  Note:
  - EthSwt has ETHSWT_E_INV_POINTER, but no ETHSWT_E_PARAM_POINTER (different default error identifier style has been applied, is handled by RfC # 76195).
  - WdgIf has both WDGIF_E_INV_POINTER (this is used) and WDGIF_E_PARAM_POINTER (added by RfC # 59818 and RfC # 62544), but WDGIF_E_PARAM_POINTER is not used in the SWS (as mentioned in 62544).
  - Xcp @ R4.2.2 has both XCP_E_INV_POINTER (this is used) and XCP_E_PARAM_POINTER, but XCP_E_PARAM_POINTER is not used in the SWS.
  - Xcp @ R4.3.0 has both XCP_E_PARAM_POINTER (this is used) and XCP_E_INV_POINTER, but XCP_E_INV_POINTER is not used in the SWS. (changed from R4.2.2)

  Remarks:
  Even though there're two similar errors, <MA>_E_INV_POINTER and <MA>_E_PARAM_POINTER, I think there's no strong need to replace one of them with other. It's sufficient if this "undefined" issue was resolved, I think.
  –Last change on issue 76524 comment 6–

  Agreed solution:

SWS EthIf
===
replace ETHIF_E_INV_POINTER by ETHIF_E_PARAM_POINTER

SWS EthTrcv
===
replace ETHTRCV_E_INV_POINTER by ETHTRCV_E_PARAM_POINTER

SWS Xcp
===
remove XCP_E_INV_POINTER from the table [SWS_Xcp_00857]
–Last change on issue 76524 comment 8–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.78 Specification Item SWS_EthIf_00325

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZED UNINIT otherwise (if DET is disabled) return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:

- It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
- The review task of the ITs shall be done by the WP to which the specification "belongs".

*** BSW UML Model ***
SWS_CanNm:
———-
Chapter 8.6.1 Optional Interfaces:
Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

SWS_LinIf:
———-
SWS_LinIf_00359: add Det_ReportRuntimeError

SWS_UdpNm:
———-
Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.79 Specification Item SWS_EthIf_00327

Trace References:

none

Content:

The function EthIf_SetPhyTxLoopbackMode shall forward the call to function EthTrcv_SetPhyTxLoopbackMode of the corresponding Ethernet Transceiver Driver (EthIfTransceiverIdx).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77007: [EthIf] [SWS_EthIf_00327][SWS_EthIf_00328][SWS_EthIf_00329] assigned to two different requirements

  Problem description:

  [SWS_EthIf_00327][SWS_EthIf_00328][SWS_EthIf_00329] were assigned to two different APIs, EthIf_SetPhyLoopbackMode() and EthIf_SetPhyTxMode().

  Note that, at SVN rev. 257404 (comes from the IT # 75454), [SWS_EthIf_00327] was added with the text: "The function EthIf_SetPhyTxMode shall forward the call to function EthTrcv_SetPhyTxMode of the corresponding Ethernet Transceiver Driver (EthIfTransceiverIdx).". But corresponding change history for EthIf_SetPhyTxMode() was not found.

  ———

  sec. 8.3.30 EthIf_SetPhyLoopbackMode

  ———

  [SWS_EthIf_00327]
  The function EthIf_SetPhyLoopbackMode shall forward the call to function
  EthTrcv_SetPhyLoopbackMode of the corresponding Ethernet Transceiver Driver
  (EthIfTransceiverIdx). ()

  [SWS_EthIf_00328]
  If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZED otherwise (if DET is disabled) return E_NOT_OK.()

  [SWS_EthIf_00329]
  If development error detection is enabled: the function shall check the parameter TrcvIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_TRCV_IDX otherwise (if DET is disabled) return E_NOT_OK.()

  ———

  sec. 8.3.31 EthIf_SetPhyTxMode

  ———

  [SWS_EthIf_00327]
  The function EthIf_SetPhyTxMode shall forward the call to function
  EthTrcv_SetPhyTxMode of the corresponding Ethernet Transceiver Driver
  (EthIfTransceiverIdx). ()

[SWS_EthIf_00328]
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZED otherwise (if DET is disabled) return E_NOT_OK. ()

[SWS_EthIf_00329]
If development error detection is enabled: the function shall check the parameter TrcvIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_TRCV_IDX otherwise (if DET is disabled) return E_NOT_OK.()

Agreed solution:

In sec. 8.3.31 EthIf_SetPhyTxMode
* [SWS_EthIf_00327]: assign new requirement ID
* [SWS_EthIf_00328]: assign new requirement ID
* [SWS_EthIf_00329]: assign new requirement ID

BW-C-Level:

Application	Specification	Bus
1	1	1

1.80 Specification Item SWS_EthIf_00328

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZED UNINIT otherwise (if DET is disabled) return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

Agreed solution:

solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

Notes:
- It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
- The review task of the ITs shall be done by the WP to which the specification "belongs".

*** BSW UML Model ***
SWS_CanNm:
———-
Chapter 8.6.1 Optional Interfaces:
Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

SWS_LinIf:
———-
SWS_LinIf_00359: add Det_ReportRuntimeError

SWS_UdpNm:
———-
Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.81 Specification Item SWS_EthIf_00331

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZED UNINIT otherwise (if DET is disabled) return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of ’Runtime errors’

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification "belongs".

  *** BSW UML Model ***
  SWS_CanNm:
  ———-
  Chapter 8.6.1 Optional Interfaces:
  Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

  SWS_LinIf:
  ———-
  SWS_LinIf_00359: add Det_ReportRuntimeError

  SWS_UdpNm:
  ———-
  Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.82 Specification Item SWS_EthIf_00333

Trace References:

none

Content:

If development error detection is enabled: the function shall check the parameter ResultPtr for being valid. If the check fails, the function shall raise the development error ETHIF_E_INVPARAM_POINTER otherwise (if DET is disabled) return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76524: [EthIf][EthTrcv] missing definition for ETHIF_E_INV_POINTER / ETHTRCV_E_INV_POINTER

  Problem description:

  ETHIF_E_INV_POINTER appears in [SWS_EthIf_00242] [SWS_EthIf_00323] [SWS_EthIf_00333] [SWS_EthIf_00337] [SWS_EthIf_00338] [SWS_EthIf_00339].

  However, there's no definition for ETHIF_E_INV_POINTER.

  As default error list ([SWS_EthIf_00017]) already have ETHIF_E_PARAM_POINTER 0x05 (I believe it fits to the purpose of ETHIF_E_INV_POINTER), could you consider to replace ETHIF_E_INV_POINTER with ETHIF_E_PARAM_POINTER, please?

  Also, same issue exists at EthTrcv (ETHTRCV_E_INV_POINTER not defined at [SWS_EthTrcv_00131], ETHTRCV_E_PARAM_POINTER exists).

Note:
- EthSwt has ETHSWT_E_INV_POINTER, but no ETHSWT_E_PARAM_POINTER (different default error identifier style has been applied, is handled by RfC # 76195).
- WdgIf has both WDGIF_E_INV_POINTER (this is used) and WDGIF_E_PARAM_POINTER (added by RfC # 59818 and RfC # 62544), but WDGIF_E_PARAM_POINTER is not used in the SWS (as mentioned in 62544).
- Xcp @ R4.2.2 has both XCP_E_INV_POINTER (this is used) and XCP_E_PARAM_POINTER, but XCP_E_PARAM_POINTER is not used in the SWS.
- Xcp @ R4.3.0 has both XCP_E_PARAM_POINTER (this is used) and XCP_E_INV_POINTER, but XCP_E_INV_POINTER is not used in the SWS. (changed from R4.2.2)

Remarks:
Even though there're two similar errors, <MA>_E_INV_POINTER and <MA>_E_PARAM_POINTER, I think there's no strong need to replace one of them with other. It's sufficient if this "undefined" issue was resolved, I think.
–Last change on issue 76524 comment 6–

Agreed solution:

SWS EthIf
==
replace ETHIF_E_INV_POINTER by ETHIF_E_PARAM_POINTER

SWS EthTrcv
==
replace ETHTRCV_E_INV_POINTER by ETHTRCV_E_PARAM_POINTER

SWS Xcp
==
remove XCP_E_INV_POINTER from the table [SWS_Xcp_00857]
–Last change on issue 76524 comment 8–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.83 Specification Item SWS_EthIf_00335

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZED UNINIT otherwise (if DET is disabled) return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification "belongs".

  *** BSW UML Model ***
  SWS_CanNm:
  ———-
  Chapter 8.6.1 Optional Interfaces:
  Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

  SWS_LinIf:
  ———-
  SWS_LinIf_00359: add Det_ReportRuntimeError

  SWS_UdpNm:
  ———-
  Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.84 Specification Item SWS_EthIf_00337

Trace References:

none

Content:

If development error detection is enabled: the function shall check the parameter Org UniqueIdPtr for being valid. If the check fails, the function shall raise the development error ETHIF_E_INVPARAM_POINTER otherwise (if DET is disabled) return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76524: [EthIf][EthTrcv] missing definition for ETHIF_E_INV_POINTER / ETHTRCV_E_INV_POINTER

  Problem description:

  ETHIF_E_INV_POINTER appears in [SWS_EthIf_00242] [SWS_EthIf_00323] [SWS_EthIf_00333] [SWS_EthIf_00337] [SWS_EthIf_00338] [SWS_EthIf_00339].

  However, there's no definition for ETHIF_E_INV_POINTER.

  As default error list ([SWS_EthIf_00017]) already have ETHIF_E_PARAM_POINTER 0x05 (I believe it fits to the purpose of ETHIF_E_INV_POINTER), could you consider to replace ETHIF_E_INV_POINTER with ETHIF_E_PARAM_POINTER, please?

  Also, same issue exists at EthTrcv (ETHTRCV_E_INV_POINTER not defined at [SWS_EthTrcv_00131], ETHTRCV_E_PARAM_POINTER exists).

Note:
- EthSwt has ETHSWT_E_INV_POINTER, but no ETHSWT_E_PARAM_POINTER (different default error identifier style has been applied, is handled by RfC # 76195).
- WdgIf has both WDGIF_E_INV_POINTER (this is used) and WDGIF_E_PARAM_POINTER (added by RfC # 59818 and RfC # 62544), but WDGIF_E_PARAM_POINTER is not used in the SWS (as mentioned in 62544).
- Xcp @ R4.2.2 has both XCP_E_INV_POINTER (this is used) and XCP_E_PARAM_POINTER, but XCP_E_PARAM_POINTER is not used in the SWS.
- Xcp @ R4.3.0 has both XCP_E_PARAM_POINTER (this is used) and XCP_E_INV_POINTER, but XCP_E_INV_POINTER is not used in the SWS. (changed from R4.2.2)

Remarks:
Even though there're two similar errors, <MA>_E_INV_POINTER and <MA>_E_PARAM_POINTER, I think there's no strong need to replace one of them with other. It's sufficient if this "undefined" issue was resolved, I think.
–Last change on issue 76524 comment 6–

Agreed solution:

SWS EthIf
===
replace ETHIF_E_INV_POINTER by ETHIF_E_PARAM_POINTER

SWS EthTrcv
===
replace ETHTRCV_E_INV_POINTER by ETHTRCV_E_PARAM_POINTER

SWS Xcp
===
remove XCP_E_INV_POINTER from the table [SWS_Xcp_00857]
–Last change on issue 76524 comment 8–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.85 Specification Item SWS_EthIf_00338

Trace References:

none

Content:

If development error detection is enabled: the function shall check the parameter ModelNrPtr for being valid. If the check fails, the function shall raise the development error ETHIF_E_INVPARAM_POINTER otherwise (if DET is disabled) return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76524: [EthIf][EthTrcv] missing definition for ETHIF_E_INV_POINTER / ETHTRCV_E_INV_POINTER

  Problem description:

  ETHIF_E_INV_POINTER appears in [SWS_EthIf_00242] [SWS_EthIf_00323] [SWS_EthIf_00333] [SWS_EthIf_00337] [SWS_EthIf_00338] [SWS_EthIf_00339].

  However, there's no definition for ETHIF_E_INV_POINTER.

  As default error list ([SWS_EthIf_00017]) already have ETHIF_E_PARAM_POINTER 0x05 (I believe it fits to the purpose of ETHIF_E_INV_POINTER), could you consider to replace ETHIF_E_INV_POINTER with ETHIF_E_PARAM_POINTER, please?

  Also, same issue exists at EthTrcv (ETHTRCV_E_INV_POINTER not defined at [SWS_EthTrcv_00131], ETHTRCV_E_PARAM_POINTER exists).

  Note:
  - EthSwt has ETHSWT_E_INV_POINTER, but no ETHSWT_E_PARAM_POINTER (different default error identifier style has been applied, is handled by RfC # 76195).
  - WdgIf has both WDGIF_E_INV_POINTER (this is used) and WDGIF_E_PARAM_POINTER (added by RfC # 59818 and RfC # 62544), but WDGIF_E_PARAM_POINTER is not used in the SWS (as mentioned in 62544).
  - Xcp @ R4.2.2 has both XCP_E_INV_POINTER (this is used) and XCP_E_PARAM_POINTER, but XCP_E_PARAM_POINTER is not used in the SWS.
  - Xcp @ R4.3.0 has both XCP_E_PARAM_POINTER (this is used) and XCP_E_INV_POINTER, but XCP_E_INV_POINTER is not used in the SWS. (changed from R4.2.2)

  Remarks:

Even though there're two similar errors, <MA>_E_INV_POINTER and <MA>_E_PARAM_POINTER, I think there's no strong need to replace one of them with other. It's sufficient if this "undefined" issue was resolved, I think.
–Last change on issue 76524 comment 6–

Agreed solution:

SWS EthIf
===
replace ETHIF_E_INV_POINTER by ETHIF_E_PARAM_POINTER

SWS EthTrcv
===
replace ETHTRCV_E_INV_POINTER by ETHTRCV_E_PARAM_POINTER

SWS Xcp
===
remove XCP_E_INV_POINTER from the table [SWS_Xcp_00857]
–Last change on issue 76524 comment 8–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.86 Specification Item SWS_EthIf_00339

Trace References:

none

Content:

If development error detection is enabled: the function shall check the parameter Revision NrPtr for being valid. If the check fails, the function shall raise the development error ETHIF_E_INVPARAM_POINTER otherwise (if DET is disabled) return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76524: [EthIf][EthTrcv] missing definition for ETHIF_E_INV_POINTER / ETHTRCV_E_INV_POINTER

  Problem description:

  ETHIF_E_INV_POINTER appears in [SWS_EthIf_00242] [SWS_EthIf_00323] [SWS_EthIf_00333] [SWS_EthIf_00337] [SWS_EthIf_00338] [SWS_EthIf_00339].

However, there's no definition for ETHIF_E_INV_POINTER.

As default error list ([SWS_EthIf_00017]) already have ETHIF_E_PARAM_POINTER 0x05 (I believe it fits to the purpose of ETHIF_E_INV_POINTER), could you consider to replace ETHIF_E_INV_POINTER with ETHIF_E_PARAM_POINTER, please?

Also, same issue exists at EthTrcv (ETHTRCV_E_INV_POINTER not defined at [SWS_EthTrcv_00131], ETHTRCV_E_PARAM_POINTER exists).

Note:
- EthSwt has ETHSWT_E_INV_POINTER, but no ETHSWT_E_PARAM_POINTER (different default error identifier style has been applied, is handled by RfC # 76195).
- WdgIf has both WDGIF_E_INV_POINTER (this is used) and WDGIF_E_PARAM_POINTER (added by RfC # 59818 and RfC # 62544), but WDGIF_E_PARAM_POINTER is not used in the SWS (as mentioned in 62544).
- Xcp @ R4.2.2 has both XCP_E_INV_POINTER (this is used) and XCP_E_PARAM_POINTER, but XCP_E_PARAM_POINTER is not used in the SWS.
- Xcp @ R4.3.0 has both XCP_E_PARAM_POINTER (this is used) and XCP_E_INV_POINTER, but XCP_E_INV_POINTER is not used in the SWS. (changed from R4.2.2)

Remarks:
Even though there're two similar errors, <MA>_E_INV_POINTER and <MA>_E_PARAM_POINTER, I think there's no strong need to replace one of them with other. It's sufficient if this "undefined" issue was resolved, I think.
–Last change on issue 76524 comment 6–

Agreed solution:

SWS EthIf
==
replace ETHIF_E_INV_POINTER by ETHIF_E_PARAM_POINTER

SWS EthTrcv
==
replace ETHTRCV_E_INV_POINTER by ETHTRCV_E_PARAM_POINTER

SWS Xcp

==
remove XCP_E_INV_POINTER from the table [SWS_Xcp_00857]
–Last change on issue 76524 comment 8–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.87 Specification Item SWS_EthIf_00343

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service Eth If_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZEDUNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification "belongs".

  *** BSW UML Model ***
  SWS_CanNm:
  ———-

Chapter 8.6.1 Optional Interfaces:
Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

SWS_LinIf:
———-
SWS_LinIf_00359: add Det_ReportRuntimeError

SWS_UdpNm:
———-
Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.88 Specification Item SWS_EthIf_00349

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZEDUNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of ’Runtime errors’

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

Notes:
- It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
- The review task of the ITs shall be done by the WP to which the specification "belongs".

*** BSW UML Model ***
SWS_CanNm:
———-
Chapter 8.6.1 Optional Interfaces:
Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

SWS_LinIf:
———-
SWS_LinIf_00359: add Det_ReportRuntimeError

SWS_UdpNm:
———-
Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.89 Specification Item SWS_EthIf_00355

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZEDUNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification "belongs".

  *** BSW UML Model ***
  SWS_CanNm:
  ———-
  Chapter 8.6.1 Optional Interfaces:
  Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

  SWS_LinIf:
  ———-
  SWS_LinIf_00359: add Det_ReportRuntimeError

  SWS_UdpNm:
  ———-
  Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

  *** ECUC XML ***

Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.90 Specification Item SWS_EthIf_00362

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZEDUNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification "belongs".

  *** BSW UML Model ***
  SWS_CanNm:
  ———-

Chapter 8.6.1 Optional Interfaces:
Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

SWS_LinIf:
———-
SWS_LinIf_00359: add Det_ReportRuntimeError

SWS_UdpNm:
———-
Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.91 Specification Item SWS_EthIf_00368

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZEDUNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

Notes:
- It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
- The review task of the ITs shall be done by the WP to which the specification "belongs".

*** BSW UML Model ***
SWS_CanNm:
———-
Chapter 8.6.1 Optional Interfaces:
Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

SWS_LinIf:
———-
SWS_LinIf_00359: add Det_ReportRuntimeError

SWS_UdpNm:
———-
Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.92 Specification Item SWS_EthIf_00375

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZEDUNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification "belongs".

  *** BSW UML Model ***
  SWS_CanNm:
  ———-
  Chapter 8.6.1 Optional Interfaces:
  Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

  SWS_LinIf:
  ———-
  SWS_LinIf_00359: add Det_ReportRuntimeError

  SWS_UdpNm:
  ———-
  Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

  *** ECUC XML ***

Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.93 Specification Item SWS_EthIf_00382

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZEDUNINIT.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification "belongs".

  *** BSW UML Model ***
  SWS_CanNm:
  ———-

Chapter 8.6.1 Optional Interfaces:
Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

SWS_LinIf:
———-
SWS_LinIf_00359: add Det_ReportRuntimeError

SWS_UdpNm:
———-
Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.94 Specification Item SWS_EthIf_00388

Trace References:

none

Content:

The function EthIf_SetPhyTxMode shall forward the call to function EthTrcv_SetPhyTxMode of the corresponding Ethernet Transceiver Driver (EthIfTransceiverIdx).

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77007: [EthIf] [SWS_EthIf_00327][SWS_EthIf_00328][SWS_EthIf_00329] assigned to two different requirements

  Problem description:

  [SWS_EthIf_00327][SWS_EthIf_00328][SWS_EthIf_00329] were assigned to two different APIs, EthIf_SetPhyLoopbackMode() and EthIf_SetPhyTxMode().

  Note that, at SVN rev. 257404 (comes from the IT # 75454), [SWS_EthIf_00327] was added with the text: "The function EthIf_SetPhyTxMode shall forward the call to function EthTrcv_SetPhyTxMode of the corresponding Ethernet Transceiver Driver (EthIfTransceiverIdx).". But corresponding change history for EthIf_SetPhyTxMode() was not found.

---

sec. 8.3.30 EthIf_SetPhyLoopbackMode

[SWS_EthIf_00327]
The function EthIf_SetPhyLoopbackMode shall forward the call to function EthTrcv_SetPhyLoopbackMode of the corresponding Ethernet Transceiver Driver (EthIfTransceiverIdx). ()

[SWS_EthIf_00328]
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZED otherwise (if DET is disabled) return E_NOT_OK.()

[SWS_EthIf_00329]
If development error detection is enabled: the function shall check the parameter TrcvIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_TRCV_IDX otherwise (if DET is disabled) return E_NOT_OK.()

---

sec. 8.3.31 EthIf_SetPhyTxMode

[SWS_EthIf_00327]
The function EthIf_SetPhyTxMode shall forward the call to function EthTrcv_SetPhyTxMode of the corresponding Ethernet Transceiver Driver (EthIfTransceiverIdx). ()

[SWS_EthIf_00328]
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZED otherwise (if DET is disabled) return E_NOT_OK. ()

[SWS_EthIf_00329]
If development error detection is enabled: the function shall check the parameter

TrcvIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_TRCV_IDX otherwise (if DET is disabled) return E_NOT_OK.()

Agreed solution:

In sec. 8.3.31 EthIf_SetPhyTxMode
* [SWS_EthIf_00327]: assign new requirement ID
* [SWS_EthIf_00328]: assign new requirement ID
* [SWS_EthIf_00329]: assign new requirement ID

BW-C-Level:

Application	Specification	Bus
1	1	1

1.95 Specification Item SWS_EthIf_00389

Trace References:

none

Content:

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT otherwise (if DET is disabled) return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #59085: Rollout of 'Runtime errors'

  Problem description:

  Inconsistencies in SWS with semantics of Default errors
  –Last change on issue 59085 comment 26–

  Agreed solution:

  solution in Column "G" of the new attachment https://www.autosar.org/bugzilla/attachment.cgi?id=4604

  Notes:
  - It is not enough just to migrate the error from one classification table to another. Please also check the related requirements (and background information) which is referring to that error and adapt them if needed.
  - The review task of the ITs shall be done by the WP to which the specification "belongs".

*** BSW UML Model ***
SWS_CanNm:
———-
Chapter 8.6.1 Optional Interfaces:
Add within SWS_CanNm_00325 the API function Det_ReportRunTimeError

SWS_LinIf:
———-
SWS_LinIf_00359: add Det_ReportRuntimeError

SWS_UdpNm:
———-
Replace UDPNM_E_NO_INIT with UDPNM_E_UNINIT in description of API UdpNm_MainFunction_<Instance Id> (SWS_UdpNm_00234)

*** ECUC XML ***
Not affected. No configuration of runtime error reporting required (see SWS BSW General).
–Last change on issue 59085 comment 88–

BW-C-Level:

Application	Specification	Bus
1	4	1

- RfC #77007: [EthIf] [SWS_EthIf_00327][SWS_EthIf_00328][SWS_EthIf_00329] assigned to two different requirements

  Problem description:

  [SWS_EthIf_00327][SWS_EthIf_00328][SWS_EthIf_00329] were assigned to two different APIs, EthIf_SetPhyLoopbackMode() and EthIf_SetPhyTxMode().

  Note that, at SVN rev. 257404 (comes from the IT # 75454), [SWS_EthIf_00327] was added with the text: "The function EthIf_SetPhyTxMode shall forward the call to function EthTrcv_SetPhyTxMode of the corresponding Ethernet Transceiver Driver (EthIfTransceiverIdx).". But corresponding change history for EthIf_SetPhyTxMode() was not found.

  ——

  sec. 8.3.30 EthIf_SetPhyLoopbackMode

  ——

[SWS_EthIf_00327]
The function EthIf_SetPhyLoopbackMode shall forward the call to function
EthTrcv_SetPhyLoopbackMode of the corresponding Ethernet Transceiver Driver
(EthIfTransceiverIdx). ()

[SWS_EthIf_00328]
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZED otherwise (if DET is disabled) return E_NOT_OK.()

[SWS_EthIf_00329]
If development error detection is enabled: the function shall check the parameter TrcvIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_TRCV_IDX otherwise (if DET is disabled) return E_NOT_OK.()

---

sec. 8.3.31 EthIf_SetPhyTxMode

[SWS_EthIf_00327]
The function EthIf_SetPhyTxMode shall forward the call to function
EthTrcv_SetPhyTxMode of the corresponding Ethernet Transceiver Driver
(EthIfTransceiverIdx). ()

[SWS_EthIf_00328]
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZED otherwise (if DET is disabled) return E_NOT_OK. ()

[SWS_EthIf_00329]
If development error detection is enabled: the function shall check the parameter TrcvIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_TRCV_IDX otherwise (if DET is disabled) return E_NOT_OK.()

Agreed solution:

In sec. 8.3.31 EthIf_SetPhyTxMode
* [SWS_EthIf_00327]: assign new requirement ID
* [SWS_EthIf_00328]: assign new requirement ID
* [SWS_EthIf_00329]: assign new requirement ID

BW-C-Level:

Application	Specification	Bus
1	1	1

1.96 Specification Item SWS_EthIf_00390

Trace References:

none

Content:

If development error detection is enabled: the function shall check the parameter TrcvIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_TRCV_IDX otherwise (if DET is disabled) return E_NOT_OK.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77007: [EthIf] [SWS_EthIf_00327][SWS_EthIf_00328][SWS_EthIf_00329] assigned to two different requirements

  Problem description:

  [SWS_EthIf_00327][SWS_EthIf_00328][SWS_EthIf_00329] were assigned to two different APIs, EthIf_SetPhyLoopbackMode() and EthIf_SetPhyTxMode().

  Note that, at SVN rev. 257404 (comes from the IT # 75454), [SWS_EthIf_00327] was added with the text: "The function EthIf_SetPhyTxMode shall forward the call to function EthTrcv_SetPhyTxMode of the corresponding Ethernet Transceiver Driver (EthIfTransceiverIdx).". But corresponding change history for EthIf_SetPhyTxMode() was not found.

  ——

  sec. 8.3.30 EthIf_SetPhyLoopbackMode

  ——

  [SWS_EthIf_00327]
  The function EthIf_SetPhyLoopbackMode shall forward the call to function
  EthTrcv_SetPhyLoopbackMode of the corresponding Ethernet Transceiver Driver
  (EthIfTransceiverIdx). ()

  [SWS_EthIf_00328]
  If development error detection is enabled: the function shall check that the service

EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZED otherwise (if DET is disabled) return E_NOT_OK.()

[SWS_EthIf_00329]
If development error detection is enabled: the function shall check the parameter TrcvIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_TRCV_IDX otherwise (if DET is disabled) return E_NOT_OK.()

---

sec. 8.3.31 EthIf_SetPhyTxMode

[SWS_EthIf_00327]
The function EthIf_SetPhyTxMode shall forward the call to function
EthTrcv_SetPhyTxMode of the corresponding Ethernet Transceiver Driver
(EthIfTransceiverIdx). ()

[SWS_EthIf_00328]
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOT_INITIALIZED otherwise (if DET is disabled) return E_NOT_OK. ()

[SWS_EthIf_00329]
If development error detection is enabled: the function shall check the parameter TrcvIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_TRCV_IDX otherwise (if DET is disabled) return E_NOT_OK.()

Agreed solution:

In sec. 8.3.31 EthIf_SetPhyTxMode
* [SWS_EthIf_00327]: assign new requirement ID
* [SWS_EthIf_00328]: assign new requirement ID
* [SWS_EthIf_00329]: assign new requirement ID

BW-C-Level:

Application	Specification	Bus
1	1	1

1.97 Specification Item SWS_EthIf_91019

Trace References:

SRS_Eth_00117

Content:

Service name:	EthIf_GetPhySignalQuality (obsolete)EthIf_GetPhySignalQuality	
Syntax:	Std_ReturnType EthIf_GetPhySignalQuality(uint8 TrcvIdx, uint8* SignalQualityPtr)	
Service ID[hex]:	0x16	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.	
Parameters (in):	TrcvIdxEthIf_GetPhySignalQuality.TrcvIdx	Index of the transceiver within the context of the Ethernet Interface
Parameters (inout):	None	
Parameters (out):	SignalQualityPtrEthIf_GetPhySignalQuality.SignalQualityPtr	Pointer to the memory where the signal quality in percent shall be stored.
Return value:	Std_ReturnType	E_OK: The request has been accepted E_NOT_OK: The request has not been accepted.
Description:	Obtains the current signal quality of the link of the indexed transceiver Tags: atp.Status=obsolete	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77676: [EthIf][EthSwt][EthTrcv] Complete handling to determine signal quality

  Problem description:

  AR4.3.0 introduce APIs EthSwt_GetPortSignalQuality and EthTrcv_GetPhySignalQuality. This API's return the result of the signal quality meassured by a dedicated Ethernet transceiver. But there is a lack of how the meassurement for the signal quality shall be triggert respectively handled.

  Agreed solution:

  === EthTrcv ===
  ~[SWS_EthTrcv_91001] EthTrcv_GetPhySignalQuality
  ~ uint8* SignalQualityPtr change to uint32* SignalQualityPtr
  ~ change the description of parameter SignalQualityPtr to "Pointer to the memory where the signal quality shall be stored."

  === EthSwt ===

~[SWS_EthSwt_91014] EthSwt_GetPortSignalQuality
~ uint8* SignalQualityPtr change to uint32* SignalQualityPtr
~ change the description of parameter SignalQualityPtr to "Pointer to the memory where the signal quality shall be stored."
~ change the description: "...If no transceiver is referenced the signal quality shall be set to 0xFFFFFFFF."
~[SWS_EthSwt_00293] change "0xFF" to "0xFFFFFFFF"

=== EthIf ===
~ch.8.2
+[EthIf_xxxxx1] EthIf_SignalQualityResultType
Type Structure
Element uint32 HighestSignalQuality the highest signal quality of a link since last clear
Element uint32 LowestSignalQuality the lowest link signal quality of a link since last clear
Element uint32 ActualSignalQuality the actual signal quality

~ch. 8.3
~ [SWS_EthIf_91019] set EthIf_GetPhySignalQuality to "deprecated"

+[EthIf_xxxxx2] EthIf_GetTrcvSignalQuality
Syntax: Std_ReturnType EthIf_GetTrcvSignalQuality(
uint8 TrcvIdx,
EthIf_SignalQualityResultType* ResultPtr)
Sync/Async: Synchronous
Reentrancy: Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.
Parameters (in): TrcvIdx Index of the transceiver within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): ResultPtr Pointer to the memory where the signal quality in percent shall be stored.
Return value: Std_ReturnType
E_OK: The signal quality retrieved successfully
E_NOT_OK: The signal quality not retrieved successfully
Description: Retrieves the signal quality of the link of the given Ethernet transceiver

+[EthIf_xxxxx3] EthIf_GetSwitchPortSignalQuality
Syntax: Std_ReturnType EthIf_GetSwitchPortSignalQuality(
uint8 SwitchIdx,
uint8 SwitchPortIdx,
EthIf_SignalQualityResultType* ResultPtr)

Sync/Async: Synchronous
Reentrancy: Reentrant for different Ethernet switch indexes and Ethernet Switch port indexes. Non reentrant for the same SwitchPortIdx.
Parameters (in): SwitchIdx Index of the Ethernet switch within the context of the Ethernet Interface
SwitchPortIdx Index of the Ethernet switch port within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): ResultPtr Pointer to the memory where the signal quality in percent shall be stored.
Return value: Std_ReturnType
E_OK: The signal quality retrieved successfully
E_NOT_OK: The signal quality not retrieved successfully
Description: Retrieves the signal quality of the link of the given Ethernet switch port

+[EthIf_xxxxx4] EthIf_ClearTrcvSignalQuality
Syntax: Std_ReturnType EthIf_ClearTrcvSignalQuality(
uint8 TrcvIdx)
Sync/Async: Synchronous
Reentrancy: Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.
Parameters (in): TrcvIdx Index of the transceiver within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): None
Return value: Std_ReturnType
E_OK: The signal quality cleared successfully
E_NOT_OK: The signal quality cleared not successfully
Description: Clear the stored signal quality of the link of the given Ethernet transceiver

+[EthIf_xxxxx5] EthIf_ClearSwitchPortSignalQuality
Syntax: Std_ReturnType EthIf_ClearSwitchPortSignalQuality(
uint8 SwitchIdx,
uint8 SwitchPortIdx)
Sync/Async: Synchronous
Reentrancy: Reentrant for different Ethernet switch indexes and Ethernet Switch port indexes. Non reentrant for the same SwitchPortIdx.
Parameters (in): SwitchIdx Index of the Ethernet switch within the context of the Ethernet Interface
SwitchPortIdx Index of the Ethernet switch port within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): None

Return value: Std_ReturnType
E_OK: The signal quality cleared successfully
E_NOT_OK: The signal quality cleared not successfully
Description: Clear the stored signal quality of the link of the given Ethernet switch port

~ch. 8.5 Scheduled functions
~ SWS_EthIf_91104 EthIf_MainFunctionState
~Description: The function is polling different communication hardware (Ethernet transceiver, Ethernet switch ports) related information, e.g. link state, signal quality.

+[EthIf_xxxx12] The EthIf_MainFunctionState shall poll Ethernet communication hardware related information with the period of EthIfMainFunctionStatePeriod.

+[EthIf_xxxxx6] For each Ethernet switch port where a link state of ETHSWT_LINK_STATE_ACTIVE is yielded and references an Ethernet Transceiver the function shall poll the signal quality by calling EthSwt_GetPortSignalQuality().

+[EthIf_xxxxx7] For each Ethernet transceiver where a link state of ETHTRCV_LINK_STATE_ACTIVE is yielded the function shall poll the signal quality by calling EthTrcv_GetPhySignalQuality().

+[EthIf_xxxxx9] The obtained signal quality value shall be stored as type of EthIf_SignalQualityResultType. The value shall always be stored as ActualSignalQuality. If the obtained signal quality is higher than the stored highest signal quality (HighestSignalQuality), then HighestSignalQuality shall be updated with the obtained signal quality. If the obtained signal quality is lower than the lowest signal quality (LowestSignalQuality), then LowestSignalQuality shall be updated with the obtained signal quality.

~ch. 10 Configuration specification
add the following parameters to section 10.1.2 EthIfGeneral
+ [ECUC_EthIf_xxxxx1] EthIfEnableSignalQualityApi
+ Description: enable/disable the APIs read and clear the signal quality
+ Multiplicity : 1
+ Type : EcucBooleanParamDef
+ Scope / Dependency : scope: local

+ [ECUC_EthIf_xxxxx2] EthIfSignalQualityCheckPeriod
+ Description : Specifies the period in units of seconds in which the signal quality it polled in the context of EthIf_MainfunctionState. The value shall be an integral multiple of EthIfMainFunctionStatePeriod.

+ Multiplicity : 0..1
+ Type : EcucFloatParamDef
+ Scope / Dependency : scope: local
dependency: If this parameter is defined, the EthIf_MainFunctionState shall be generated and parameter EthIfEnableSignalQualityApi shall be set to TRUE.
–Last change on issue 77676 comment 38–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.98 Specification Item SWS_EthIf_91052

Trace References:

none

Content:

Service name:	EthIf_GetVlanIdEthIf_GetVlanId	
Syntax:	Std_ReturnType EthIf_GetVlanId(uint8 CtrlIdx, uint16* VlanIdPtr)	
Service ID[hex]:	0x43	
Sync/Async:	Asynchronous Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	CtrlIdxEthIf_GetVlanId.CtrlIdx	Index of the Ethernet controller within the context of the Ethernet Interface
Parameters (inout):	None	
Parameters (out):	VlanIdPtrEthIf_GetVlanId.VlanIdPtr	Pointer to store the VLAN identifier (VID) of the Ethernet controller. 0 if the the Ethernet controller represents no virtual network (VLAN).
Return value:	Std_ReturnType	E_OK: success E_NOT_OK: failure
Description:	Returns the VLAN identifier of the requested Ethernet controller.	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77982: [EthIf] ClassicPlatform: The API functions EthIf_GetVlanId() shall be synchronous

  Problem description:

Name: Helmut Gepp
Phone: +43 1 59983-5032
Role: Developer

Description/Motivation:
The API function EthIf_GetVlanId() reads only a configuration parameter (VLAN identifier) and returns it. There is no waiting or blocking state. Thus there is no reason which this function is defined as asynchronous.

Was there already a decision? No

Agreed solution:

~SWS_EthIf_91052
old:
Sync/Async: Asynchronous
new:;
Sync/Async: Synchronous
–Last change on issue 77982 comment 2–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.99 Specification Item SWS_EthIf_91054

Trace References:

none

Content:

Service name:	EthIf_GetBufWTxParamsEthIf_GetBufWTxParams
Syntax:	Std_ReturnType EthIf_GetBufWTxParams(uint8 CtrlIdx, const WEth_BufWTxParamIdType* TxParamIds, uint32* ParamValues, uint8 NumParams)
Service ID[hex]:	0x31

Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	CtrlIdxEthIf_GetBufWTxParams.CtrlIdx	Index of the Ethernet controller within the context of the Ethernet Interface
	TxParamIdsEthIf_GetBufWTxParams.TxParamIds	IDs of the Parameter that are requested
	NumParamsEthIf_GetBufWTxParams.NumParams	Number of Parameters that are requested
Parameters (inout):	None	
Parameters (out):	ParamValuesEthIf_GetBufWTxParams.ParamValues	Values of the Parameters requested
Return value:	Std_ReturnType	E_OK: success E_NOT_OK: failed reading parameters
Description:	Read out values related to the transmit direction of the transceiver for a transmitted packet. For example, this could be transaction ID belonging to one single packet.	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76647: [EthIf] Findings in EthIf

  Problem description:

  Name: Stellwag
  Phone:
  Role: WP-A review (as part of V2x review)

  Figure 1/Introduction: should be enhanced with the wireless modules. Also the V2x should be mentioned as upper layer.

  Chapter 4.1: What does "The Ethernet Interface module is only able to handle a single thread of execution. The execution must not be pre-empted by itself." mean? Only one "caller" at a time? If the EthIf is using a callback/callout the called function is not allowed to use the EthIf? Please detail/explain.

  Chapter 4.1: "The Ethernet Interface module is only able to handle a single thread of execution. The execution must not be pre-empted by itself." is no limitation - please reword or remove

  Chapter 5/page 14: add V2xGn to the list of users of EthIf; remove the "empty" list

SWS_EthIf_00226: Why is EthIf unconditionally using the EthSwt..Cfg.h file? What is really required? Btw: This dependency is not shown in the diagram in chapter 5.1.1 ...

SWS_EthIf_00261: please replace SWS_EthIf_xxx21 with the real number

Chapter 7.2.1: There is no "Default error" in AUTOSAR. Please replace with "development error"

SWS_EthIf_00149: If we follow SWS_EthIf_00153 then this type is missing in SWS_EthIf_00023 .... but maybe SWS_EthIf_00023 only means those types starting with Eth_ ...

SWS_EthIf_00150: This type is availibe but seems not be used in any API ... please clarify or remove

SWS_EthIf_00039: This function is marked as non-reentrant. If the EthIf has access to more drivers (e.g. Eth and WEth): Who coordinates the users that no parallel calls are done? This problem applies to many APIs.

SWS_EthIf_00190: Does this function only works when a swicth is used? What about cases without a switch but some dedicated controllers? What about wirless access? Please either update or change name to EthIf_GetPortMacAddrFromSwitch()

SWS_EthIf_00214: This API stores "values" "somewhere" - is there any service to "restore" this values? SWS_EthIf_00219 just "resets" the values ...

SWS_EthIf_00095: "Caveat: The function requires previous initialization (EthIf_Init)" is no requirement for me, please remove. There are more requirements starting with "Caveat: ..." ...

SWS_EthIf_00098: This contradicts SWS_BSW_00037, please remove (same applies to SWS_EthIf_00278)

SWS_EthIf_00099: ETHIF_ENABLE_RX_INTERRUPT is not defined in the SWS ...

SWS_EthIf_91051: What is the difference to SWS_EthIf_00097? Why do we need N receiever main functions? Where is the period configured? (it’s not in ECUC_EthIf_00023...)

SWS_EthIf_00100: ETHIF_ENABLE_TX_INTERRUPT is not defined in the SWS ...

Chapter 8.5: Why are different tables are used for scheduled functions? This just confuses readers, I suggest to use the form from the SWS template.

SWS_EthIf_00102: This table is empty. Either add functions required by the EthIf or remove table.

SWS_EthIf_91009: I think this is not an "Asynchronous" function but a synchronouse one.

Agreed solution:

In chapter 1:
- change "This indicates the main task of the Ethernet Interface:
Provide to upper layers a hardware independent interface to the Ethernet Communication System comprising multiple different Ethernet controllers and transceivers. This interface shall be uniform for all Ethernet controllers and transceivers. Thus, the upper layers (TCP/IP, EthSM, CDD) may access the underlying bus system in a uniform manner."
to
"This indicates the main task of the Ethernet Interface:
Provide to upper layers a hardware independent interface to the Ethernet Communication System comprising multiple different wired or wireless Ethernet controllers and transceivers. This interface shall be uniform for all Ethernet controllers and transceivers. Thus, the upper layers (TCP/IP, EthSM, CDD, V2x modules) may access the underlying bus system in a uniform manner."
- Add a upper layer "V2x modules" to figure 1
- Add WEth and WEthTrcv as lower layer modules to figure 1

replace
SWS_EthIf_00114
The function shall change the state of the component from ETHIF_STATE_UNINIT to ETHIF_STATE_INIT
by
SWS_EthIf_00114
The function shall change the state of the component from uninitialized to initialized.

Chapter 4.1:
- remove "The Ethernet Interface module is only able to handle a single thread of execution. The execution must not be pre-empted by itself."

Chapter 5:
- add V2xGn to the list of users of EthIf; remove the "empty" list of "Modules used

by the Ethernet Interface module:"

SWS_EthIf_00150:
- remove SWS_EthIf_00150

~SWS_EthIf_00153 from:
The types specified in SWS_EthernetInterface shall be declared in Eth_GeneralTypes.h.()
to
General Eth types specified in SWS_EthernetInterface shall be declared in Eth_GeneralTypes.h.()

8.3.35 EthIf_GetBufWTxParams
replace SWS-ID by new ID: SWS_EthIf_91009 -> SWS_EthIf_xxxxx

~ SWS_EthIf_00097
remove from Description: " It checks also for transceiver state changes."
replace from Description: "transmission confirmations" by "reception indications"

~SWS_EthIf_00099
replace ETHIF_ENABLE_RX_INTERRUPT by EthIfEnableRxInterrupt

~SWS_EthIf_00100
replace ETHIF_ENABLE_TX_INTERRUPT by EthIfEnableTxInterrupt

remove SWS_EthIf_00102

Please remove the requirement id and convert it to a note:
SWS_EthIf_00027, SWS_EthIf_00038, SWS_EthIf_00044, SWS_EthIf_00066, SWS_EthIf_00138, SWS_EthIf_00144, SWS_EthIf_00195, SWS_EthIf_00201, SWS_EthIf_00207, SWS_EthIf_00218, SWS_EthIf_00223, SWS_EthIf_00159, SWS_EthIf_00165, SWS_EthIf_00171, SWS_EthIf_00177, SWS_EthIf_00074, SWS_EthIf_00089, SWS_EthIf_00095

Please remove the requirement id and maybe keep it as a note:
SWS_EthIf_00249, SWS_EthIf_00284, SWS_EthIf_00081, SWS_EthIf_00090, SWS_EthIf_00096, SWS_EthIf_00292.
–Last change on issue 76647 comment 37–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.100 Specification Item SWS_EthIf_91056

Trace References:

none

Content:

Service name:	EthIf_GetTrcvSignalQualityEthIf_GetTrcvSignalQuality	
Syntax:	Std_ReturnType EthIf_GetTrcvSignalQuality(uint8 TrcvIdx, EthIf_SignalQualityResultType* ResultPtr)	
Service ID[hex]:	0	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.	
Parameters (in):	TrcvIdxEthIf_GetTrcvSignalQuality.TrcvIdx	Index of the transceiver within the context of the Ethernet Interface
Parameters (inout):	None	
Parameters (out):	ResultPtrEthIf_GetTrcvSignalQuality.ResultPtr	Pointer to the memory where the signal quality in percent shall be stored.
Return value:	Std_ReturnType	E_OK: The signal quality retrieved successfully E_NOT_OK: The signal quality not retrieved successfully
Description:	Retrieves the signal quality of the link of the given Ethernet transceiver	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77676: [EthIf][EthSwt][EthTrcv] Complete handling to determine signal quality

  Problem description:

  AR4.3.0 introduce APIs EthSwt_GetPortSignalQuality and EthTrcv_GetPhySignalQuality. This API's return the result of the signal quality meassured by a dedicated Ethernet transceiver. But there is a lack of how the meassurement for the signal quality shall be triggert respectively handled.

  Agreed solution:

=== EthTrcv ===
~[SWS_EthTrcv_91001] EthTrcv_GetPhySignalQuality
~ uint8* SignalQualityPtr change to uint32* SignalQualityPtr
~ change the description of parameter SignalQualityPtr to "Pointer to the memory where the signal quality shall be stored."

=== EthSwt ===
~[SWS_EthSwt_91014] EthSwt_GetPortSignalQuality
~ uint8* SignalQualityPtr change to uint32* SignalQualityPtr
~ change the description of parameter SignalQualityPtr to "Pointer to the memory where the signal quality shall be stored."
~ change the description: "...If no transceiver is referenced the signal quality shall be set to 0xFFFFFFFF."
~[SWS_EthSwt_00293] change "0xFF" to "0xFFFFFFFF"

=== EthIf ===
~ch.8.2
+[EthIf_xxxxx1] EthIf_SignalQualityResultType
Type Structure
Element uint32 HighestSignalQuality the highest signal quality of a link since last clear
Element uint32 LowestSignalQuality the lowest link signal quality of a link since last clear
Element uint32 ActualSignalQuality the actual signal quality

~ch. 8.3
~ [SWS_EthIf_91019] set EthIf_GetPhySignalQuality to "deprecated"

+[EthIf_xxxxx2] EthIf_GetTrcvSignalQuality
Syntax: Std_ReturnType EthIf_GetTrcvSignalQuality(
uint8 TrcvIdx,
EthIf_SignalQualityResultType* ResultPtr)
Sync/Async: Synchronous
Reentrancy: Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.
Parameters (in): TrcvIdx Index of the transceiver within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): ResultPtr Pointer to the memory where the signal quality in percent shall be stored.
Return value: Std_ReturnType
E_OK: The signal quality retrieved successfully
E_NOT_OK: The signal quality not retrieved successfully

Description: Retrieves the signal quality of the link of the given Ethernet transceiver

+[EthIf_xxxxx3] EthIf_GetSwitchPortSignalQuality
Syntax: Std_ReturnType EthIf_GetSwitchPortSignalQuality(
uint8 SwitchIdx,
uint8 SwitchPortIdx,
EthIf_SignalQualityResultType* ResultPtr)
Sync/Async: Synchronous
Reentrancy: Reentrant for different Ethernet switch indexes and Ethernet Switch port indexes. Non reentrant for the same SwitchPortIdx.
Parameters (in): SwitchIdx Index of the Ethernet switch within the context of the Ethernet Interface
SwitchPortIdx Index of the Ethernet switch port within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): ResultPtr Pointer to the memory where the signal quality in percent shall be stored.
Return value: Std_ReturnType
E_OK: The signal quality retrieved successfully
E_NOT_OK: The signal quality not retrieved successfully
Description: Retrieves the signal quality of the link of the given Ethernet switch port

+[EthIf_xxxxx4] EthIf_ClearTrcvSignalQuality
Syntax: Std_ReturnType EthIf_ClearTrcvSignalQuality(
uint8 TrcvIdx)
Sync/Async: Synchronous
Reentrancy: Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.
Parameters (in): TrcvIdx Index of the transceiver within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): None
Return value: Std_ReturnType
E_OK: The signal quality cleared successfully
E_NOT_OK: The signal quality cleared not successfully
Description: Clear the stored signal quality of the link of the given Ethernet transceiver

+[EthIf_xxxxx5] EthIf_ClearSwitchPortSignalQuality
Syntax: Std_ReturnType EthIf_ClearSwitchPortSignalQuality(
uint8 SwitchIdx,
uint8 SwitchPortIdx)
Sync/Async: Synchronous
Reentrancy: Reentrant for different Ethernet switch indexes and Ethernet Switch

port indexes. Non reentrant for the same SwitchPortIdx.
Parameters (in): SwitchIdx Index of the Ethernet switch within the context of the Ethernet Interface
SwitchPortIdx Index of the Ethernet switch port within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): None
Return value: Std_ReturnType
E_OK: The signal quality cleared successfully
E_NOT_OK: The signal quality cleared not successfully
Description: Clear the stored signal quality of the link of the given Ethernet switch port

~ch. 8.5 Scheduled functions
~ SWS_EthIf_91104 EthIf_MainFunctionState
~Description: The function is polling different communication hardware (Ethernet transceiver, Ethernet switch ports) related information, e.g. link state, signal quality.

+[EthIf_xxxx12] The EthIf_MainFunctionState shall poll Ethernet communication hardware related information with the period of EthIfMainFunctionStatePeriod.

+[EthIf_xxxxx6] For each Ethernet switch port where a link state of ETHSWT_LINK_STATE_ACTIVE is yielded and references an Ethernet Transceiver the function shall poll the signal quality by calling EthSwt_GetPortSignalQuality().

+[EthIf_xxxxx7] For each Ethernet transceiver where a link state of ETHTRCV_LINK_STATE_ACTIVE is yielded the function shall poll the signal quality by calling EthTrcv_GetPhySignalQuality().

+[EthIf_xxxxx9] The obtained signal quality value shall be stored as type of EthIf_SignalQualityResultType. The value shall always be stored as ActualSignalQuality. If the obtained signal quality is higher than the stored highest signal quality (HighestSignalQuality), then HighestSignalQuality shall be updated with the obtained signal quality. If the obtained signal quality is lower than the lowest signal quality (LowestSignalQuality), then LowestSignalQuality shall be updated with the obtained signal quality.

~ch. 10 Configuration specification
add the following parameters to section 10.1.2 EthIfGeneral
+ [ECUC_EthIf_xxxxx1] EthIfEnableSignalQualityApi
+ Description: enable/disable the APIs read and clear the signal quality
+ Multiplicity : 1

+ Type : EcucBooleanParamDef
+ Scope / Dependency : scope: local

+ [ECUC_EthIf_xxxxx2] EthIfSignalQualityCheckPeriod
+ Description : Specifies the period in units of seconds in which the signal quality it polled in the context of EthIf_MainfunctionState. The value shall be an integral multiple of EthIfMainFunctionStatePeriod.
+ Multiplicity : 0..1
+ Type : EcucFloatParamDef
+ Scope / Dependency : scope: local
dependency: If this parameter is defined, the EthIf_MainFunctionState shall be generated and parameter EthIfEnableSignalQualityApi shall be set to TRUE.
–Last change on issue 77676 comment 38–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.101 Specification Item SWS_EthIf_91057

Trace References:

none

Content:

Name:	EthIf_SignalQualityResultTypeEthIf_SignalQualityResultType		
Type:	Structure		
Element:	uint32	HighestSignalQualityEthIf_SignalQualityResultType.HighestSignalQuality	the highest signal quality of a link since last clear
	uint32	LowestSignalQualityEthIf_SignalQualityResultType.LowestSignalQuality	the lowest link signal quality of a link since last clear
	uint32	ActualSignalQualityEthIf_SignalQualityResultType.ActualSignalQuality	the actual signal quality
Description:	–		

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77676: [EthIf][EthSwt][EthTrcv] Complete handling to determine signal quality

Problem description:

AR4.3.0 introduce APIs EthSwt_GetPortSignalQuality and EthTrcv_GetPhySignalQuality. This API's return the result of the signal quality meassured by a dedicated Ethernet transceiver. But there is a lack of how the meassurement for the signal quality shall be triggert respectively handled.

Agreed solution:

=== EthTrcv ===
~[SWS_EthTrcv_91001] EthTrcv_GetPhySignalQuality
~ uint8* SignalQualityPtr change to uint32* SignalQualityPtr
~ change the description of parameter SignalQualityPtr to "Pointer to the memory where the signal quality shall be stored."

=== EthSwt ===
~[SWS_EthSwt_91014] EthSwt_GetPortSignalQuality
~ uint8* SignalQualityPtr change to uint32* SignalQualityPtr
~ change the description of parameter SignalQualityPtr to "Pointer to the memory where the signal quality shall be stored."
~ change the description: "...If no transceiver is referenced the signal quality shall be set to 0xFFFFFFFF."
~[SWS_EthSwt_00293] change "0xFF" to "0xFFFFFFFF"

=== EthIf ===
~ch.8.2
+[EthIf_xxxxx1] EthIf_SignalQualityResultType
Type Structure
Element uint32 HighestSignalQuality the highest signal quality of a link since last clear
Element uint32 LowestSignalQuality the lowest link signal quality of a link since last clear
Element uint32 ActualSignalQuality the actual signal quality

~ch. 8.3
~ [SWS_EthIf_91019] set EthIf_GetPhySignalQuality to "deprecated"

+[EthIf_xxxxx2] EthIf_GetTrcvSignalQuality
Syntax: Std_ReturnType EthIf_GetTrcvSignalQuality(
uint8 TrcvIdx,
EthIf_SignalQualityResultType* ResultPtr)
Sync/Async: Synchronous
Reentrancy: Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.

Parameters (in): TrcvIdx Index of the transceiver within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): ResultPtr Pointer to the memory where the signal quality in percent shall be stored.
Return value: Std_ReturnType
E_OK: The signal quality retrieved successfully
E_NOT_OK: The signal quality not retrieved successfully
Description: Retrieves the signal quality of the link of the given Ethernet transceiver

+[EthIf_xxxxx3] EthIf_GetSwitchPortSignalQuality
Syntax: Std_ReturnType EthIf_GetSwitchPortSignalQuality(
uint8 SwitchIdx,
uint8 SwitchPortIdx,
EthIf_SignalQualityResultType* ResultPtr)
Sync/Async: Synchronous
Reentrancy: Reentrant for different Ethernet switch indexes and Ethernet Switch port indexes. Non reentrant for the same SwitchPortIdx.
Parameters (in): SwitchIdx Index of the Ethernet switch within the context of the Ethernet Interface
SwitchPortIdx Index of the Ethernet switch port within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): ResultPtr Pointer to the memory where the signal quality in percent shall be stored.
Return value: Std_ReturnType
E_OK: The signal quality retrieved successfully
E_NOT_OK: The signal quality not retrieved successfully
Description: Retrieves the signal quality of the link of the given Ethernet switch port

+[EthIf_xxxxx4] EthIf_ClearTrcvSignalQuality
Syntax: Std_ReturnType EthIf_ClearTrcvSignalQuality(
uint8 TrcvIdx)
Sync/Async: Synchronous
Reentrancy: Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.
Parameters (in): TrcvIdx Index of the transceiver within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): None
Return value: Std_ReturnType
E_OK: The signal quality cleared successfully
E_NOT_OK: The signal quality cleared not successfully
Description: Clear the stored signal quality of the link of the given Ethernet

transceiver

+[EthIf_xxxxx5] EthIf_ClearSwitchPortSignalQuality
Syntax: Std_ReturnType EthIf_ClearSwitchPortSignalQuality(
uint8 SwitchIdx,
uint8 SwitchPortIdx)
Sync/Async: Synchronous
Reentrancy: Reentrant for different Ethernet switch indexes and Ethernet Switch port indexes. Non reentrant for the same SwitchPortIdx.
Parameters (in): SwitchIdx Index of the Ethernet switch within the context of the Ethernet Interface
SwitchPortIdx Index of the Ethernet switch port within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): None
Return value: Std_ReturnType
E_OK: The signal quality cleared successfully
E_NOT_OK: The signal quality cleared not successfully
Description: Clear the stored signal quality of the link of the given Ethernet switch port

~ch. 8.5 Scheduled functions
~ SWS_EthIf_91104 EthIf_MainFunctionState
~Description: The function is polling different communication hardware (Ethernet transceiver, Ethernet switch ports) related information, e.g. link state, signal quality.

+[EthIf_xxxx12] The EthIf_MainFunctionState shall poll Ethernet communication hardware related information with the period of EthIfMainFunctionStatePeriod.

+[EthIf_xxxxx6] For each Ethernet switch port where a link state of ETHSWT_LINK_STATE_ACTIVE is yielded and references an Ethernet Transceiver the function shall poll the signal quality by calling EthSwt_GetPortSignalQuality().

+[EthIf_xxxxx7] For each Ethernet transceiver where a link state of ETHTRCV_LINK_STATE_ACTIVE is yielded the function shall poll the signal quality by calling EthTrcv_GetPhySignalQuality().

+[EthIf_xxxxx9] The obtained signal quality value shall be stored as type of EthIf_SignalQualityResultType. The value shall always be stored as ActualSignalQuality. If the obtained signal quality is higher than the stored highest signal quality (HighestSignalQuality), then HighestSignalQuality shall be updated with the obtained signal quality. If the obtained signal quality is lower than the lowest signal quality (LowestSignalQuality), then LowestSignalQuality shall be updated with the obtained signal quality.

~ch. 10 Configuration specification
add the following parameters to section 10.1.2 EthIfGeneral
+ [ECUC_EthIf_xxxxx1] EthIfEnableSignalQualityApi
+ Description: enable/disable the APIs read and clear the signal quality
+ Multiplicity : 1
+ Type : EcucBooleanParamDef
+ Scope / Dependency : scope: local

+ [ECUC_EthIf_xxxxx2] EthIfSignalQualityCheckPeriod
+ Description : Specifies the period in units of seconds in which the signal quality it polled in the context of EthIf_MainfunctionState. The value shall be an integral multiple of EthIfMainFunctionStatePeriod.
+ Multiplicity : 0..1
+ Type : EcucFloatParamDef
+ Scope / Dependency : scope: local
dependency: If this parameter is defined, the EthIf_MainFunctionState shall be generated and parameter EthIfEnableSignalQualityApi shall be set to TRUE.
–Last change on issue 77676 comment 38–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.102 Specification Item SWS_EthIf_91058

Trace References:

none

Content:

Service name:	EthIf_GetSwitchPortSignalQualityEthIf_GetSwitchPortSignalQuality
Syntax:	Std_ReturnType EthIf_GetSwitchPortSignalQuality(uint8 SwitchIdx, uint8 SwitchPortIdx, EthIf_SignalQualityResultType* ResultPtr)
Service ID[hex]:	0
Sync/Async:	Synchronous

Reentrancy:	Reentrant for different Ethernet switch indexes and Ethernet Switch port indexes. Non reentrant for the same SwitchPortIdx.	
Parameters (in):	SwitchIdxEthIf_GetSwitchPortSignalQuality.SwitchIdx	Index of the Ethernet switch within the context of the Ethernet Interface
	SwitchPortIdxEthIf_GetSwitchPortSignalQuality.SwitchPortIdx	Index of the Ethernet switch port within the context of the Ethernet Interface
Parameters (inout):	None	
Parameters (out):	ResultPtrEthIf_GetSwitchPortSignalQuality.ResultPtr	Pointer to the memory where the signal quality in percent shall be stored.
Return value:	Std_ReturnType	E_OK: The signal quality retrieved successfully E_NOT_OK: The signal quality not retrieved successfully
Description:	Retrieves the signal quality of the link of the given Ethernet switch port	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77676: [EthIf][EthSwt][EthTrcv] Complete handling to determine signal quality

  Problem description:

  AR4.3.0 introduce APIs EthSwt_GetPortSignalQuality and EthTrcv_GetPhySignalQuality. This API's return the result of the signal quality meassured by a dedicated Ethernet transceiver. But there is a lack of how the meassurement for the signal quality shall be triggert respectively handled.

  Agreed solution:

  === EthTrcv ===
  ~[SWS_EthTrcv_91001] EthTrcv_GetPhySignalQuality
  ~ uint8* SignalQualityPtr change to uint32* SignalQualityPtr
  ~ change the description of parameter SignalQualityPtr to "Pointer to the memory where the signal quality shall be stored."

  === EthSwt ===
  ~[SWS_EthSwt_91014] EthSwt_GetPortSignalQuality
  ~ uint8* SignalQualityPtr change to uint32* SignalQualityPtr
  ~ change the description of parameter SignalQualityPtr to "Pointer to the memory where the signal quality shall be stored."
  ~ change the description: "...If no transceiver is referenced the signal quality shall be set to 0xFFFFFFFF."
  ~[SWS_EthSwt_00293] change "0xFF" to "0xFFFFFFFF"

=== EthIf ===
~ch.8.2
+[EthIf_xxxxx1] EthIf_SignalQualityResultType
Type Structure
Element uint32 HighestSignalQuality the highest signal quality of a link since last clear
Element uint32 LowestSignalQuality the lowest link signal quality of a link since last clear
Element uint32 ActualSignalQuality the actual signal quality

~ch. 8.3
~ [SWS_EthIf_91019] set EthIf_GetPhySignalQuality to "deprecated"

+[EthIf_xxxxx2] EthIf_GetTrcvSignalQuality
Syntax: Std_ReturnType EthIf_GetTrcvSignalQuality(
uint8 TrcvIdx,
EthIf_SignalQualityResultType* ResultPtr)
Sync/Async: Synchronous
Reentrancy: Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.
Parameters (in): TrcvIdx Index of the transceiver within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): ResultPtr Pointer to the memory where the signal quality in percent shall be stored.
Return value: Std_ReturnType
E_OK: The signal quality retrieved successfully
E_NOT_OK: The signal quality not retrieved successfully
Description: Retrieves the signal quality of the link of the given Ethernet transceiver

+[EthIf_xxxxx3] EthIf_GetSwitchPortSignalQuality
Syntax: Std_ReturnType EthIf_GetSwitchPortSignalQuality(
uint8 SwitchIdx,
uint8 SwitchPortIdx,
EthIf_SignalQualityResultType* ResultPtr)
Sync/Async: Synchronous
Reentrancy: Reentrant for different Ethernet switch indexes and Ethernet Switch port indexes. Non reentrant for the same SwitchPortIdx.
Parameters (in): SwitchIdx Index of the Ethernet switch within the context of the Ethernet Interface
SwitchPortIdx Index of the Ethernet switch port within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): ResultPtr Pointer to the memory where the signal quality in

percent shall be stored.
Return value: Std_ReturnType
E_OK: The signal quality retrieved successfully
E_NOT_OK: The signal quality not retrieved successfully
Description: Retrieves the signal quality of the link of the given Ethernet switch port

+[EthIf_xxxxx4] EthIf_ClearTrcvSignalQuality
Syntax: Std_ReturnType EthIf_ClearTrcvSignalQuality(
uint8 TrcvIdx)
Sync/Async: Synchronous
Reentrancy: Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.
Parameters (in): TrcvIdx Index of the transceiver within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): None
Return value: Std_ReturnType
E_OK: The signal quality cleared successfully
E_NOT_OK: The signal quality cleared not successfully
Description: Clear the stored signal quality of the link of the given Ethernet transceiver

+[EthIf_xxxxx5] EthIf_ClearSwitchPortSignalQuality
Syntax: Std_ReturnType EthIf_ClearSwitchPortSignalQuality(
uint8 SwitchIdx,
uint8 SwitchPortIdx)
Sync/Async: Synchronous
Reentrancy: Reentrant for different Ethernet switch indexes and Ethernet Switch port indexes. Non reentrant for the same SwitchPortIdx.
Parameters (in): SwitchIdx Index of the Ethernet switch within the context of the Ethernet Interface
SwitchPortIdx Index of the Ethernet switch port within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): None
Return value: Std_ReturnType
E_OK: The signal quality cleared successfully
E_NOT_OK: The signal quality cleared not successfully
Description: Clear the stored signal quality of the link of the given Ethernet switch port

~ch. 8.5 Scheduled functions
~ SWS_EthIf_91104 EthIf_MainFunctionState

~Description: The function is polling different communication hardware (Ethernet transceiver, Ethernet switch ports) related information, e.g. link state, signal quality.

+[EthIf_xxxx12] The EthIf_MainFunctionState shall poll Ethernet communication hardware related information with the period of EthIfMainFunctionStatePeriod.

+[EthIf_xxxxx6] For each Ethernet switch port where a link state of ETHSWT_LINK_STATE_ACTIVE is yielded and references an Ethernet Transceiver the function shall poll the signal quality by calling EthSwt_GetPortSignalQuality().

+[EthIf_xxxxx7] For each Ethernet transceiver where a link state of ETHTRCV_LINK_STATE_ACTIVE is yielded the function shall poll the signal quality by calling EthTrcv_GetPhySignalQuality().

+[EthIf_xxxxx9] The obtained signal quality value shall be stored as type of EthIf_SignalQualityResultType. The value shall always be stored as ActualSignalQuality. If the obtained signal quality is higher than the stored highest signal quality (HighestSignalQuality), then HighestSignalQuality shall be updated with the obtained signal quality. If the obtained signal quality is lower than the lowest signal quality (LowestSignalQuality), then LowestSignalQuality shall be updated with the obtained signal quality.

~ch. 10 Configuration specification
add the following parameters to section 10.1.2 EthIfGeneral
+ [ECUC_EthIf_xxxxx1] EthIfEnableSignalQualityApi
+ Description: enable/disable the APIs read and clear the signal quality
+ Multiplicity : 1
+ Type : EcucBooleanParamDef
+ Scope / Dependency : scope: local

+ [ECUC_EthIf_xxxxx2] EthIfSignalQualityCheckPeriod
+ Description : Specifies the period in units of seconds in which the signal quality it polled in the context of EthIf_MainfunctionState. The value shall be an integral multiple of EthIfMainFunctionStatePeriod.
+ Multiplicity : 0..1
+ Type : EcucFloatParamDef
+ Scope / Dependency : scope: local
dependency: If this parameter is defined, the EthIf_MainFunctionState shall be generated and parameter EthIfEnableSignalQualityApi shall be set to TRUE.
–Last change on issue 77676 comment 38–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.103 Specification Item SWS_EthIf_91059

Trace References:

none

Content:

Service name:	EthIf_ClearTrcvSignalQualityEthIf_ClearTrcvSignalQuality	
Syntax:	Std_ReturnType EthIf_ClearTrcvSignalQuality(uint8 TrcvIdx)	
Service ID[hex]:	0	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.	
Parameters (in):	TrcvIdxEthIf_ClearTrcvSignalQuality.TrcvIdx	Index of the transceiver within the context of the Ethernet Interface
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: The signal quality cleared successfully E_NOT_OK: The signal quality cleared not successfully
Description:	Clear the stored signal quality of the link of the given Ethernet transceiver	

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77676: [EthIf][EthSwt][EthTrcv] Complete handling to determine signal quality

  Problem description:

  AR4.3.0 introduce APIs EthSwt_GetPortSignalQuality and EthTrcv_GetPhySignalQuality. This API's return the result of the signal quality meassured by a dedicated Ethernet transceiver. But there is a lack of how the meassurement for the signal quality shall be triggert respectively handled.

  Agreed solution:

  === EthTrcv ===
  ~[SWS_EthTrcv_91001] EthTrcv_GetPhySignalQuality

~ uint8* SignalQualityPtr change to uint32* SignalQualityPtr
~ change the description of parameter SignalQualityPtr to "Pointer to the memory where the signal quality shall be stored."

=== EthSwt ===
~[SWS_EthSwt_91014] EthSwt_GetPortSignalQuality
~ uint8* SignalQualityPtr change to uint32* SignalQualityPtr
~ change the description of parameter SignalQualityPtr to "Pointer to the memory where the signal quality shall be stored."
~ change the description: "...If no transceiver is referenced the signal quality shall be set to 0xFFFFFFFF."
~[SWS_EthSwt_00293] change "0xFF" to "0xFFFFFFFF"

=== EthIf ===
~ch.8.2
+[EthIf_xxxxx1] EthIf_SignalQualityResultType
Type Structure
Element uint32 HighestSignalQuality the highest signal quality of a link since last clear
Element uint32 LowestSignalQuality the lowest link signal quality of a link since last clear
Element uint32 ActualSignalQuality the actual signal quality

~ch. 8.3
~ [SWS_EthIf_91019] set EthIf_GetPhySignalQuality to "deprecated"

+[EthIf_xxxxx2] EthIf_GetTrcvSignalQuality
Syntax: Std_ReturnType EthIf_GetTrcvSignalQuality(
uint8 TrcvIdx,
EthIf_SignalQualityResultType* ResultPtr)
Sync/Async: Synchronous
Reentrancy: Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.
Parameters (in): TrcvIdx Index of the transceiver within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): ResultPtr Pointer to the memory where the signal quality in percent shall be stored.
Return value: Std_ReturnType
E_OK: The signal quality retrieved successfully
E_NOT_OK: The signal quality not retrieved successfully
Description: Retrieves the signal quality of the link of the given Ethernet transceiver

+[EthIf_xxxxx3] EthIf_GetSwitchPortSignalQuality
Syntax: Std_ReturnType EthIf_GetSwitchPortSignalQuality(
uint8 SwitchIdx,
uint8 SwitchPortIdx,
EthIf_SignalQualityResultType* ResultPtr)
Sync/Async: Synchronous
Reentrancy: Reentrant for different Ethernet switch indexes and Ethernet Switch port indexes. Non reentrant for the same SwitchPortIdx.
Parameters (in): SwitchIdx Index of the Ethernet switch within the context of the Ethernet Interface
SwitchPortIdx Index of the Ethernet switch port within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): ResultPtr Pointer to the memory where the signal quality in percent shall be stored.
Return value: Std_ReturnType
E_OK: The signal quality retrieved successfully
E_NOT_OK: The signal quality not retrieved successfully
Description: Retrieves the signal quality of the link of the given Ethernet switch port

+[EthIf_xxxxx4] EthIf_ClearTrcvSignalQuality
Syntax: Std_ReturnType EthIf_ClearTrcvSignalQuality(
uint8 TrcvIdx)
Sync/Async: Synchronous
Reentrancy: Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.
Parameters (in): TrcvIdx Index of the transceiver within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): None
Return value: Std_ReturnType
E_OK: The signal quality cleared successfully
E_NOT_OK: The signal quality cleared not successfully
Description: Clear the stored signal quality of the link of the given Ethernet transceiver

+[EthIf_xxxxx5] EthIf_ClearSwitchPortSignalQuality
Syntax: Std_ReturnType EthIf_ClearSwitchPortSignalQuality(
uint8 SwitchIdx,
uint8 SwitchPortIdx)
Sync/Async: Synchronous
Reentrancy: Reentrant for different Ethernet switch indexes and Ethernet Switch port indexes. Non reentrant for the same SwitchPortIdx.
Parameters (in): SwitchIdx Index of the Ethernet switch within the context of the

Ethernet Interface
SwitchPortIdx Index of the Ethernet switch port within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): None
Return value: Std_ReturnType
E_OK: The signal quality cleared successfully
E_NOT_OK: The signal quality cleared not successfully
Description: Clear the stored signal quality of the link of the given Ethernet switch port

~ch. 8.5 Scheduled functions
~ SWS_EthIf_91104 EthIf_MainFunctionState
~Description: The function is polling different communication hardware (Ethernet transceiver, Ethernet switch ports) related information, e.g. link state, signal quality.

+[EthIf_xxxx12] The EthIf_MainFunctionState shall poll Ethernet communication hardware related information with the period of EthIfMainFunctionStatePeriod.

+[EthIf_xxxxx6] For each Ethernet switch port where a link state of ETHSWT_LINK_STATE_ACTIVE is yielded and references an Ethernet Transceiver the function shall poll the signal quality by calling EthSwt_GetPortSignalQuality().

+[EthIf_xxxxx7] For each Ethernet transceiver where a link state of ETHTRCV_LINK_STATE_ACTIVE is yielded the function shall poll the signal quality by calling EthTrcv_GetPhySignalQuality().

+[EthIf_xxxxx9] The obtained signal quality value shall be stored as type of EthIf_SignalQualityResultType. The value shall always be stored as ActualSignalQuality. If the obtained signal quality is higher than the stored highest signal quality (HighestSignalQuality), then HighestSignalQuality shall be updated with the obtained signal quality. If the obtained signal quality is lower than the lowest signal quality (LowestSignalQuality), then LowestSignalQuality shall be updated with the obtained signal quality.

~ch. 10 Configuration specification
add the following parameters to section 10.1.2 EthIfGeneral
+ [ECUC_EthIf_xxxxx1] EthIfEnableSignalQualityApi
+ Description: enable/disable the APIs read and clear the signal quality
+ Multiplicity : 1
+ Type : EcucBooleanParamDef
+ Scope / Dependency : scope: local

+ [ECUC_EthIf_xxxxx2] EthIfSignalQualityCheckPeriod
+ Description : Specifies the period in units of seconds in which the signal quality it polled in the context of EthIf_MainfunctionState. The value shall be an integral multiple of EthIfMainFunctionStatePeriod.
+ Multiplicity : 0..1
+ Type : EcucFloatParamDef
+ Scope / Dependency : scope: local
dependency: If this parameter is defined, the EthIf_MainFunctionState shall be generated and parameter EthIfEnableSignalQualityApi shall be set to TRUE.
–Last change on issue 77676 comment 38–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.104 Specification Item SWS_EthIf_91060

Trace References:

none

Content:

Service name:	EthIf_ClearSwitchPortSignalQualityEthIf_ClearSwitchPortSignalQuality	
Syntax:	Std_ReturnType EthIf_ClearSwitchPortSignalQuality(uint8 SwitchIdx, uint8 SwitchPortIdx)	
Service ID[hex]:	0	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant for different Ethernet switch indexes and Ethernet Switch port indexes. Non reentrant for the same SwitchPortIdx.	
Parameters (in):	SwitchIdxEthIf_ClearSwitchPortSignalQuality.SwitchIdx	Index of the Ethernet switch within the context of the Ethernet Interface
	SwitchPortIdxEthIf_ClearSwitchPortSignalQuality.SwitchPortIdx	Index of the Ethernet switch port within the context of the Ethernet Interface
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: The signal quality cleared successfully E_NOT_OK: The signal quality cleared not successfully

Description:	Clear the stored signal quality of the link of the given Ethernet switch port

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77676: [EthIf][EthSwt][EthTrcv] Complete handling to determine signal quality

  Problem description:

  AR4.3.0 introduce APIs EthSwt_GetPortSignalQuality and EthTrcv_GetPhySignalQuality. This API's return the result of the signal quality meassured by a dedicated Ethernet transceiver. But there is a lack of how the meassurement for the signal quality shall be triggert respectively handled.

  Agreed solution:

  === EthTrcv ===
  ~[SWS_EthTrcv_91001] EthTrcv_GetPhySignalQuality
  ~ uint8* SignalQualityPtr change to uint32* SignalQualityPtr
  ~ change the description of parameter SignalQualityPtr to "Pointer to the memory where the signal quality shall be stored."

  === EthSwt ===
  ~[SWS_EthSwt_91014] EthSwt_GetPortSignalQuality
  ~ uint8* SignalQualityPtr change to uint32* SignalQualityPtr
  ~ change the description of parameter SignalQualityPtr to "Pointer to the memory where the signal quality shall be stored."
  ~ change the description: "...If no transceiver is referenced the signal quality shall be set to 0xFFFFFFFF."
  ~[SWS_EthSwt_00293] change "0xFF" to "0xFFFFFFFF"

  === EthIf ===
  ~ch.8.2
  +[EthIf_xxxxx1] EthIf_SignalQualityResultType
  Type Structure
  Element uint32 HighestSignalQuality the highest signal quality of a link since last clear
  Element uint32 LowestSignalQuality the lowest link signal quality of a link since last clear
  Element uint32 ActualSignalQuality the actual signal quality

  ~ch. 8.3
  ~ [SWS_EthIf_91019] set EthIf_GetPhySignalQuality to "deprecated"

+[EthIf_xxxxx2] EthIf_GetTrcvSignalQuality
Syntax: Std_ReturnType EthIf_GetTrcvSignalQuality(
uint8 TrcvIdx,
EthIf_SignalQualityResultType* ResultPtr)
Sync/Async: Synchronous
Reentrancy: Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.
Parameters (in): TrcvIdx Index of the transceiver within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): ResultPtr Pointer to the memory where the signal quality in percent shall be stored.
Return value: Std_ReturnType
E_OK: The signal quality retrieved successfully
E_NOT_OK: The signal quality not retrieved successfully
Description: Retrieves the signal quality of the link of the given Ethernet transceiver

+[EthIf_xxxxx3] EthIf_GetSwitchPortSignalQuality
Syntax: Std_ReturnType EthIf_GetSwitchPortSignalQuality(
uint8 SwitchIdx,
uint8 SwitchPortIdx,
EthIf_SignalQualityResultType* ResultPtr)
Sync/Async: Synchronous
Reentrancy: Reentrant for different Ethernet switch indexes and Ethernet Switch port indexes. Non reentrant for the same SwitchPortIdx.
Parameters (in): SwitchIdx Index of the Ethernet switch within the context of the Ethernet Interface
SwitchPortIdx Index of the Ethernet switch port within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): ResultPtr Pointer to the memory where the signal quality in percent shall be stored.
Return value: Std_ReturnType
E_OK: The signal quality retrieved successfully
E_NOT_OK: The signal quality not retrieved successfully
Description: Retrieves the signal quality of the link of the given Ethernet switch port

+[EthIf_xxxxx4] EthIf_ClearTrcvSignalQuality
Syntax: Std_ReturnType EthIf_ClearTrcvSignalQuality(
uint8 TrcvIdx)
Sync/Async: Synchronous
Reentrancy: Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.
Parameters (in): TrcvIdx Index of the transceiver within the context of the Ethernet

Interface
Parameters (inout): None
Parameters (out): None
Return value: Std_ReturnType
E_OK: The signal quality cleared successfully
E_NOT_OK: The signal quality cleared not successfully
Description: Clear the stored signal quality of the link of the given Ethernet transceiver

+[EthIf_xxxxx5] EthIf_ClearSwitchPortSignalQuality
Syntax: Std_ReturnType EthIf_ClearSwitchPortSignalQuality(
uint8 SwitchIdx,
uint8 SwitchPortIdx)
Sync/Async: Synchronous
Reentrancy: Reentrant for different Ethernet switch indexes and Ethernet Switch port indexes. Non reentrant for the same SwitchPortIdx.
Parameters (in): SwitchIdx Index of the Ethernet switch within the context of the Ethernet Interface
SwitchPortIdx Index of the Ethernet switch port within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): None
Return value: Std_ReturnType
E_OK: The signal quality cleared successfully
E_NOT_OK: The signal quality cleared not successfully
Description: Clear the stored signal quality of the link of the given Ethernet switch port

~ch. 8.5 Scheduled functions
~ SWS_EthIf_91104 EthIf_MainFunctionState
~Description: The function is polling different communication hardware (Ethernet transceiver, Ethernet switch ports) related information, e.g. link state, signal quality.

+[EthIf_xxxx12] The EthIf_MainFunctionState shall poll Ethernet communication hardware related information with the period of EthIfMainFunctionStatePeriod.

+[EthIf_xxxxx6] For each Ethernet switch port where a link state of ETHSWT_LINK_STATE_ACTIVE is yielded and references an Ethernet Transceiver the function shall poll the signal quality by calling EthSwt_GetPortSignalQuality().

+[EthIf_xxxxx7] For each Ethernet transceiver where a link state of ETHTRCV_LINK_STATE_ACTIVE is yielded the function shall poll the signal quality by calling EthTrcv_GetPhySignalQuality().

+[EthIf_xxxxx9] The obtained signal quality value shall be stored as type of EthIf_SignalQualityResultType. The value shall always be stored as ActualSignalQuality. If the obtained signal quality is higher than the stored highest signal quality (HighestSignalQuality), then HighestSignalQuality shall be updated with the obtained signal quality. If the obtained signal quality is lower than the lowest signal quality (LowestSignalQuality), then LowestSignalQuality shall be updated with the obtained signal quality.

~ch. 10 Configuration specification
add the following parameters to section 10.1.2 EthIfGeneral
+ [ECUC_EthIf_xxxxx1] EthIfEnableSignalQualityApi
+ Description: enable/disable the APIs read and clear the signal quality
+ Multiplicity : 1
+ Type : EcucBooleanParamDef
+ Scope / Dependency : scope: local

+ [ECUC_EthIf_xxxxx2] EthIfSignalQualityCheckPeriod
+ Description : Specifies the period in units of seconds in which the signal quality it polled in the context of EthIf_MainfunctionState. The value shall be an integral multiple of EthIfMainFunctionStatePeriod.
+ Multiplicity : 0..1
+ Type : EcucFloatParamDef
+ Scope / Dependency : scope: local
dependency: If this parameter is defined, the EthIf_MainFunctionState shall be generated and parameter EthIfEnableSignalQualityApi shall be set to TRUE.
–Last change on issue 77676 comment 38–

BW-C-Level:

Application	Specification	Bus
1	4	1

1.105 Specification Item SWS_EthIf_91104

Trace References:

none

Content:

Service name:	EthIf_MainFunctionStateEthIf_MainFunctionState

Syntax:	void EthIf_MainFunctionState(void)
Service ID[hex]:	0x05
Sync/Async:	Asynchronous
Reentrancy:	Non Reentrant
Parameters (in):	None
Parameters (inout):	None
Parameters (out):	None
Return value:	None
Description:	The function is polling the link state of the used different communication hardware (Ethernet transceiver, Ethernet switch ports) related information, e.g. link state, signal quality.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77676: [EthIf][EthSwt][EthTrcv] Complete handling to determine signal quality

  Problem description:

  AR4.3.0 introduce APIs EthSwt_GetPortSignalQuality and EthTrcv_GetPhySignalQuality. This API's return the result of the signal quality meassured by a dedicated Ethernet transceiver. But there is a lack of how the meassurement for the signal quality shall be triggert respectively handled.

  Agreed solution:

  === EthTrcv ===
  ~[SWS_EthTrcv_91001] EthTrcv_GetPhySignalQuality
  ~ uint8* SignalQualityPtr change to uint32* SignalQualityPtr
  ~ change the description of parameter SignalQualityPtr to "Pointer to the memory where the signal quality shall be stored."

  === EthSwt ===
  ~[SWS_EthSwt_91014] EthSwt_GetPortSignalQuality
  ~ uint8* SignalQualityPtr change to uint32* SignalQualityPtr
  ~ change the description of parameter SignalQualityPtr to "Pointer to the memory where the signal quality shall be stored."
  ~ change the description: "...If no transceiver is referenced the signal quality shall be set to 0xFFFFFFFF."
  ~[SWS_EthSwt_00293] change "0xFF" to "0xFFFFFFFF"

  === EthIf ===
  ~ch.8.2

+[EthIf_xxxxx1] EthIf_SignalQualityResultType
Type Structure
Element uint32 HighestSignalQuality the highest signal quality of a link since last clear
Element uint32 LowestSignalQuality the lowest link signal quality of a link since last clear
Element uint32 ActualSignalQuality the actual signal quality

~ch. 8.3
~ [SWS_EthIf_91019] set EthIf_GetPhySignalQuality to "deprecated"

+[EthIf_xxxxx2] EthIf_GetTrcvSignalQuality
Syntax: Std_ReturnType EthIf_GetTrcvSignalQuality(
uint8 TrcvIdx,
EthIf_SignalQualityResultType* ResultPtr)
Sync/Async: Synchronous
Reentrancy: Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.
Parameters (in): TrcvIdx Index of the transceiver within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): ResultPtr Pointer to the memory where the signal quality in percent shall be stored.
Return value: Std_ReturnType
E_OK: The signal quality retrieved successfully
E_NOT_OK: The signal quality not retrieved successfully
Description: Retrieves the signal quality of the link of the given Ethernet transceiver

+[EthIf_xxxxx3] EthIf_GetSwitchPortSignalQuality
Syntax: Std_ReturnType EthIf_GetSwitchPortSignalQuality(
uint8 SwitchIdx,
uint8 SwitchPortIdx,
EthIf_SignalQualityResultType* ResultPtr)
Sync/Async: Synchronous
Reentrancy: Reentrant for different Ethernet switch indexes and Ethernet Switch port indexes. Non reentrant for the same SwitchPortIdx.
Parameters (in): SwitchIdx Index of the Ethernet switch within the context of the Ethernet Interface
SwitchPortIdx Index of the Ethernet switch port within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): ResultPtr Pointer to the memory where the signal quality in percent shall be stored.
Return value: Std_ReturnType

E_OK: The signal quality retrieved successfully
E_NOT_OK: The signal quality not retrieved successfully
Description: Retrieves the signal quality of the link of the given Ethernet switch port

+[EthIf_xxxxx4] EthIf_ClearTrcvSignalQuality
Syntax: Std_ReturnType EthIf_ClearTrcvSignalQuality(
uint8 TrcvIdx)
Sync/Async: Synchronous
Reentrancy: Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.
Parameters (in): TrcvIdx Index of the transceiver within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): None
Return value: Std_ReturnType
E_OK: The signal quality cleared successfully
E_NOT_OK: The signal quality cleared not successfully
Description: Clear the stored signal quality of the link of the given Ethernet transceiver

+[EthIf_xxxxx5] EthIf_ClearSwitchPortSignalQuality
Syntax: Std_ReturnType EthIf_ClearSwitchPortSignalQuality(
uint8 SwitchIdx,
uint8 SwitchPortIdx)
Sync/Async: Synchronous
Reentrancy: Reentrant for different Ethernet switch indexes and Ethernet Switch port indexes. Non reentrant for the same SwitchPortIdx.
Parameters (in): SwitchIdx Index of the Ethernet switch within the context of the Ethernet Interface
SwitchPortIdx Index of the Ethernet switch port within the context of the Ethernet Interface
Parameters (inout): None
Parameters (out): None
Return value: Std_ReturnType
E_OK: The signal quality cleared successfully
E_NOT_OK: The signal quality cleared not successfully
Description: Clear the stored signal quality of the link of the given Ethernet switch port

~ch. 8.5 Scheduled functions
~ SWS_EthIf_91104 EthIf_MainFunctionState
~Description: The function is polling different communication hardware (Ethernet transceiver, Ethernet switch ports) related information, e.g. link state, signal quality.

+[EthIf_xxxx12] The EthIf_MainFunctionState shall poll Ethernet communication hardware related information with the period of EthIfMainFunctionStatePeriod.

+[EthIf_xxxxx6] For each Ethernet switch port where a link state of ETHSWT_LINK_STATE_ACTIVE is yielded and references an Ethernet Transceiver the function shall poll the signal quality by calling EthSwt_GetPortSignalQuality().

+[EthIf_xxxxx7] For each Ethernet transceiver where a link state of ETHTRCV_LINK_STATE_ACTIVE is yielded the function shall poll the signal quality by calling EthTrcv_GetPhySignalQuality().

+[EthIf_xxxxx9] The obtained signal quality value shall be stored as type of EthIf_SignalQualityResultType. The value shall always be stored as ActualSignalQuality. If the obtained signal quality is higher than the stored highest signal quality (HighestSignalQuality), then HighestSignalQuality shall be updated with the obtained signal quality. If the obtained signal quality is lower than the lowest signal quality (LowestSignalQuality), then LowestSignalQuality shall be updated with the obtained signal quality.

~ch. 10 Configuration specification
add the following parameters to section 10.1.2 EthIfGeneral
+ [ECUC_EthIf_xxxxx1] EthIfEnableSignalQualityApi
+ Description: enable/disable the APIs read and clear the signal quality
+ Multiplicity : 1
+ Type : EcucBooleanParamDef
+ Scope / Dependency : scope: local

+ [ECUC_EthIf_xxxxx2] EthIfSignalQualityCheckPeriod
+ Description : Specifies the period in units of seconds in which the signal quality it polled in the context of EthIf_MainfunctionState. The value shall be an integral multiple of EthIfMainFunctionStatePeriod.
+ Multiplicity : 0..1
+ Type : EcucFloatParamDef
+ Scope / Dependency : scope: local
dependency: If this parameter is defined, the EthIf_MainFunctionState shall be generated and parameter EthIfEnableSignalQualityApi shall be set to TRUE.
–Last change on issue 77676 comment 38–

BW-C-Level:

Application	**Specification**	**Bus**
1	4	1