ID	Title
3509	util:Compare should refer to checker rule "uniform relation operand type"
	Description
	To ensure type safety, we should add the following remark to util:Compare: Associated checker rules: * Core_Chk049 – uniform relation operand type

ID	Title
4202	SetTestResultHandlingLevel argument is not optional
	Description
	In the TestResultHandling extension, the action SetResultHandlingLevel has this specification for <level>: > IMPORTANT — If this optional element is omitted, the default value shall > be TRACK. This note should be removed, as the element is not optional.

ID	Title
4203	Undefined behaviour if SetResultHandlingLevel not called
	Description
	It is not 100 % clear what test result handling level should be used before the SetTestResultHandlingLevel action is called for the first time. Suggestion: It should be ALL. Note: The following sentence: > IMPORTANT — If the ResultHandlingLevel declaration is not explicitly > initialized (omitted <init> element), the default value shall be TRACK. does not answer this question, as it only refers to ResultHandlingLevel declarations. Compare this to the Logging extension, which additionaly defines in Table 6 - Log-levels: ALL - All messages will be logged (this is the default setting).

ID	Title
4207	IFD:Eventhandling:MonitorChangeEventSource
	Description
	Change description of parameter variable to Represents the variable that shall be monitored. If the variable value changes or (for complex values) its contents change, the event shall be fired, causing a blocking WaitForEventAction to exit.

ID	Title
4208	CloseTestresultSession
	Description
	It is not stated,what happens if not closed - this should be added (compare with file extension)

ID	Title
4209	SetTolerancetestResult
	Description
	Threshold parameters missed (upper and lower limit)

ID	Title
4236	ComInterface: Missing MVCI system operation for IsEthernetActivated and

	IsInterfaceConnected
	Description
	In the specification of ComInterface terms IsEthernetActivated and IsInterfaceConnected the appropriate MVCI system operation is missing.
ID	Title
4248	StateMachine extension: Specification gap related to comment linking
	Description
	Original specification: 22.3.3.2 StateMachineProcedureRealisation ... – <comments> : otx:Comments [0..1] This is a container for an arbitrary-length list comments. They shall be used for commenting parts of the procedure flow implementation for the human reader. For more information please refer to Part 2 of ISO 13209. ... In the core standard all possible linking targets of comments are specified and should be also specified here in the following way. This sentence should be added after the text above: NOTE: Entities to which a comment's <link> may point to are described in Part 2 of ISO 13209. In addition, state machine comments can refer to states, transitions, transition conditions, triggers as well as state machine comments themselves.
ID	Title
4249	Wrong key constraints inside schema files
	Description
	If key constraints are contained in a schema, they are probably moved into the XML annotation, because of EA schema generation and thus lose their validity.
ID	Title
4257	Does util:Compare use lazy evaluation?
	Description
	Will this lead to an exception? util:Compare(1, 2, otx:Divide(1, 0)) I propose that it shouldn't lead to an exception, because otx:IsEqual(1, 2, otx:Divide(1, 0)) does not lead to an exception either (otx:IsEqual performs lazy evaluation). Suggestion: Add this sentence to util:Compare (I have copied it from otx:IsEqual): IMPORTANT — Since OTX terms do not have side-effects, the evaluation of Compare operand terms shall complete when the first unequal to others operand has been identified; Compare returns false then without evaluating the remaining operand terms. Note that one consequence of this is that exceptions which might have occurred in the remaining operands are bypassed.
ID	Title
4258	util:CopyByteField should be renamed to ByteFieldCopy

	Description
	The util:CopyByteField term should be renamed to util:ByteFieldCopy. Rationale: The name should be consistent with otx:ListCopy, otx:MapCopy and dataType:StructureCopy.
ID	Title
4259	BusMonitoring.BusMonitorModes seems to have no sense
	Description
	If BusMonitoring.BusMonitorModes == TRACING all terms like FetchBusMonitorBusFrames or BusMonitorEventSource have no functionality. The decision to write into a trace resource can be a project setting and should not be part of the test logic. Therefore we propose to delete the enumeration BusMonitoring.BusMonitorModes and all related terms and parameters (StartBusMonitoring.Mode).
ID	Title
4260	util:Min, Max unspecified for -0.0, NaN
	Description
	What should be the result of the following calls? 1. util:Max(1.0, NaN) * Java Math.max(1.0, Double.NaN) returns NaN * .NET Math.Max(1.0, Double.NaN) returns NaN * Python max(1.0, float("nan")) returns 1.0 2. util:Max(NaN, 1.0) * Java Math.max(Double.NaN, 1.0) returns NaN * .NET Math.Max(Double.NaN, 1.0) returns NaN * Python max(float("nan"), 1.0) returns nan 3. util:Max(-0.0, 0.0) * Java Math.max(-0.0, 0.0) returns 0.0 * I don't know how .NET behaves * Python max(-0.0, 0.0) returns -0.0 4. util:Max(0.0, -0.0) * Java Math.max(0.0, -0.0) returns 0.0 * I don't know how .NET behaves * Python max(0.0, -0.0) returns 0.0 Note: Java Collections.max(Arrays.asList(a, b)) behaves like Java Math.max(a, b) in the above cases. Java and Python behave differently because Python internally uses the normal "<" comparison, whereas java.lang.Double.compareTo() explicitly states that "comparisons performed by this method differ from those performed by the Java language numerical comparison operators (<, <=, ==, >= >)". I think that the Java behaviour should be preferred because * it is more consistent (operand order does not matter) * OTX already refers to Java as the standard implementation in similar cases Suggestion: Add this note to util:Min and util:Max: IMPORTANT — When comparing Float values, the reference implementation for comparison is the Java method ja-

	va.lang.Double.compareTo(java.lang.Double).
ID	Title
4261	GetXmlElementsByXPath should refer to XPath 1.0, not 2.0
	Description

	otxXml:22.7.3.7 GetXmlElementsByXPath specifies:
	> "Version 2.0 of the XPath standard shall be supported, as defined in
	> http://www.w3.org/TR/2010/REC-xpath20-20101214/."
	
	https://en.wikipedia.org/wiki/XPath states:
	> "There are several versions of XPath in use. XPath 1.0 was published in 1999,
	> XPath 2.0 in 2007 (with a second edition in 2010), XPath 3.0 in 2014, and XPath
	> 3.1 in 2017. However, XPath 1.0 is still the version that is most widely
	> available."
	
	XPath versions newer than 1.0 are mostly used for XSLT transformations; they are rarely used for DOM access and manipulation. Many commonly used programming platforms and open-source XML libraries only support XPath 1.0, e.g.:
	
	* Java:
	  * "As far as I know there is no XPath 2.0 support linked in to any non-XSLT-based XML processing library (JDOM/XOM/DOM/etc.)" https://stackoverflow.com/questions/14914655/xpath-2-0-feature-in-jdom-2-0
	* .NET:
	  * Does not support XPath 2.0 out of the box
	  * Third-party libraries are available: https://stackoverflow.com/questions/1525299/xpath-and-xslt-2-0-for-net
	* C/C++:
	  * The commonly used libxml2 library only supports XPath 1.0
	* Python:
	  * The commonly used lxml library is based on libxml2, and thus only supports XPath 1.0
	* TypeScript/node.js:
	  * XPath 1.0 available: https://www.npmjs.com/package/xpath-ts
	  * XPath 2.0 support can somehow be hacked in: https://stackoverflow.com/questions/52696137/evaluate-xpath2-0-in-node-js/52696870#52696870
	
	I am convinced that XPath 1.0 language features are sufficient for typical tasks in vehicle diagnostics.
	
	Suggestion: change the specification to:
	> "Version 1.0 of the XPath standard shall be supported, as defined in
	> https://www.w3.org/TR/1999/REC-xpath-19991116/."

ID	Title
4262	util:StringFormat uses C# format syntax
	Description
	The specification and examples for util:StringFormat (conversion and precision) seems to use the format string syntax of .NET: > https://docs.microsoft.com/en-us/dotnet/standard/base-types/composite-

	formatting?view=netframework-4.8 This is problematic for two reasons: 1. It is inconsistent with i18n:Translate, which specifies that message parameters should be handled "in a similar way like the string patterns specified by the Java class MessageFormat (java.text.MessageFormat)". MessageFormat uses a different syntax for conversion and precision. Thus, OTX authors are confronted with two incompatible syntaxes for argument formatting. 2. It is inconsistent with otx:ToString, especially regarding otx:Float values. otx:ToString refers to the Java implementation. This means that e.g. exponents are represented by 2 digits (not 3), and that there is no "+" for positive exponents. Thus, OTX authors get an unfamiliar output format when calling util:StringFormat. Suggestion: Use a specification like in i18n:Translate: "Strings should be formed in a similar way like the string patterns specified by the Java class MessageFormat (java.text.MessageFormat). This allows identifying parameters in the pattern unambiguously, e.g. the parameters {0} and {1} in "The resistance of injector {0} is {1}". This provided, StringFormat shall function like MessageFormat.format(String pattern, Object[] arguments), where the arguments substitute the message parameters according to their position."
ID	Title
4268	XML extension: Use namespace URIs instead of prefixes
	Description
	In several places, the XML extension specifies that a string 'can contain an arbitrary namespace in form of "Prefix:Name"' (in one section it says "Namespace:Name"). Only using prefixes to define namespaces is problematic. Assume that we want to create an OTX sequence which writes this XML: <?xml version="1.0" encoding="UTF-8"?> <TestRootNode xmlns:abc="http://example.org/abc" attributeWithoutNamespace="test1" > <abc:ChildNode abc:attribute1="foobar" /> </TestRootNode> At the moment we call otxXml:CreateXmlElement(name="abc:ChildNode), the CreateXmlElement term cannot know that the prefix "abc" will be mapped to the namespace URI "http://example.org/abc". In most XML libraries, the namespace URI must be defined during XML element creation: * Java JDOM: ** http://www.jdom.org/docs/apidocs/org/jdom2/Element.html ** If a namespace should be set, the namespace URI must be given. ** Additionally, a prefix may be given. * Java w3c.dom: ** https://docs.oracle.com/javase/7/docs/api/org/w3c/dom/Document.html ** If a namespace should be set, the namespace URI (not the prefix) must

	be given. * Web API: ** https://developer.mozilla.org/de/docs/Web/API/Document ** If a namespace should be set, the namespace URI (not the prefix) must be given. * Python lxml: ** https://lxml.de/tutorial.html#namespaces ** If a namespace should be set, the namespace URI must be given. ** Clark notation is used, e.g. "{http://www.w3.org/1999/xhtml}html" * .NET XmlDocument.CreateElement(): ** https://docs.microsoft.com/en-us/dotnet/api/system.xml.xmldocument.createelement?redirectedfrom=MSDN&view=netframework-4.8#System_Xml_XmlDocument_CreateElement_System_String_System_String_System_String_ ** If a namespace should be set, the namespace URI must be given. ** Additionally, a prefix may be given. Thus, we cannot rely on established namespace-handling routines to implement the OTX XML extension. Furthermore, it is not clearly specified how the xmlns:abc="http://example.org/abc" namespace declaration should be added to the root node. Shall it be passed in like a normal attribute? E.g.: otxXml:CreateXmlElement("TestRootNode", otx:MapLiteral("xmlns:abc"->"http://example.org/abc", "attributeWithoutNamespace"->"test1"))? This would be quite unusual. Other DOM APIs treat namespace declarations separate from attributes, and they offer a dedicated method like org.jdom2.Element.addNamespaceDeclaration() to map a prefix to a namespace URI. Suggestion: * Instead of using Prefix:Name syntax, we should specify that Clark notation should be used, i. e. the format "{namespaceUri}localname". * Explicitly state that it is runtime-dependent which namespace prefixes will be used when writing XML. (This restriction is acceptable, as namespace prefixes are meant only for human readability. Two XML documents are semantically equivalent even if e.g. the first document uses the prefix "diag" and the second uses the prefix "diagCom".) With this, the OTX pseudo-code for the above example would be: child = otxXml:CreateXmlElement("{http://example.org/abc}ChildNode", attributes = otx:MapLiteral("{http://example.org/abc}attribute1"->"foobar")) root = otxXml:CreateXmlElement("TestRootNode", attributes = otx:MapLiteral("attributeWithoutNamespace"->"test1")) otxXml:AddXmlChildElement(child, root) document = otxXml:CreateXmlDocument(rootNode) bf1 = otxXml:XmlToByteField(document) The OTX runtime may then write the XML data using an arbitrary prefix, e.g:

	<?xml version="1.0" encoding="UTF-8"?> <TestRootNode xmlns:ns0="http://example.org/abc" attributeWithoutNamespace="test1" > <ns0:ChildNode ns0:attribute1="foobar" /> </TestRootNode>
ID	Title
4269	Xml extension: Example lacks prolog line
	Description
	otxXml:XmlSaveToFile specifies: "The resulting file shall contain a prolog which specifies the character encoding and XML version." Thus, in 22.8 Example, there should be a prolog: 19 // Create simple HTML Tags which contains a header tag including a href 20 // INSERT HERE: <?xml version="1.0" encoding="UTF-8"?> 21 // <html>
ID	Title
4270	Xml extension: ValidateXml <errorMessages> should be singular
	Description
	otxXml:ValidateXml has this parameter: • <errorMessages> : otx:StringVariable [0..1] Error message if errors occurred during the validation. It should be renamed to <errorMessage> because it is a String, not a List of Strings.
ID	Title
4271	OTX extension: ValidateXml example wrong
	Description
	Section "22.8 Example" has an example "validateDOM()". It is misleading because it gives a wrong impression of the isValid parameter. The isValid parameter not only shows whether schema information has been found; it also shows whether the document is valid. Thus, both the variable name "validationDone" and the UI output are totally wrong.
ID	Title
4278	FlashPlus, Util: Wrong extension prefixes in exception annotations
	Description
	Some exception annotations are wrong because XML namespace prefixes are case-sensitive. otxIFD_FlashPlus.xsd has three occurences of: <otx:exception xsi:type="flashplus:FlashPlusException"> Instead it must be: <otx:exception xsi:type="flashPlus:FlashPlusException"> with a capital P. Similarly, otxIFD_Util.xsd has one occurence of:

	<otx:exception xsi:type="Util:UtilException"> which must instead be: <otx:exception xsi:type="util:UtilException"> with a lowercase u.
ID	Title
4281	Typo in "StatVar_Chk001 – State variables shall not read"
	Description
	A word is missing in the checker rule title. Corrected title would be: "State variables shall not be read" Actually, to be more in line with the naming style of the original checker rules, I would prefer something like this: "Write-only state variables" Moreover, the ID "StatVar_Chk001" uses the uncommon abbreviation "StatVar". The ID should be "StateVar_Chk001" or "StateVariable_Chk001".
ID	Title
4282	Remove type-safety checker rules concerning return types
	Description
	The following checker rules concerns the key/value type of MapTerm: > Xml_Chk003 – type-safe GetXmlElementAttributes > Criterion: The GetXmlElementAttributes MapTerm shall have a key datatype > <String> and value datatype <String>. We should delete this checker rule. We don't write checker rules for the return types of ListTerms and MapTerms. Compare e.g. string:SplitString, data:GetComChannelList or otxXml:GetXmlElementsByXPath, which have no such checker rules. The type safety of e.g. myMap = GetXmlElementAttributes(myElement) is guaranteed by Core_Chk023 – type-safe assignments.
ID	Title
4283	otxXml:GetXmlElementsByXPath behaviour on syntax error
	Description
	otxXml:GetXmlElementsByXPath does not specify what to do if the XPath expression has a syntax error, e.g. unbalanced parentheses or quotation marks: /rootNode/child[@attrib=="value Suggestion: Amend the existing exception specification: Exceptions: otxXml:XPathException • If the path has a syntax error, or if it points to an invalid target.
ID	Title
4284	otxXml:AddXmlChildElement behaviour if insertBefore is an unrelated element
	Description
	otxXml:GetXmlElementsByXPath does not specify what to do if the XPath expression has a syntax error, e.g. unbalanced parentheses or quotation marks: /rootNode/child[@attrib=="value

	Suggestion: Amend the existing exception specification: Exceptions: otxXml:XPathException • If the path has a syntax error, or if it points to an invalid target.
ID	Title
4285	DataType.ResourceLocation: Location should be defined by an URI
	Description
	Currently the format of the location is not specified. For the exchangeability I recommend that the location should be defined by an URI. Following changes (>>> modification <<<): 11.2.3.2 ResourceLocation A ResourceLocation represents a string value which addresses a location of a resource associated with an ID. >>> The string value should be defined by an URI. <<< 11.7.4.1 CreateResourceLocation defaultLocation Represents the optional default location. This value should be used, if the related resource ID cannot be mapped. >>> The value should be defined by an URI. <<< 11.7.4.4 ResourceLocationLiteral defaultLocation Represents the optional hard-coded default location of the ResourceLocationLiteral. This value should be used, if the related resource ID cannot be mapped. >>> The value should be defined by an URI. <<<
ID	Title
4286	otxXml:CopyXmlElement should not deep-copy the parent relationship
	Description
	The CopyXmlElement term only specifies: > Returns a deep copy of an XmlElement. An XmlElement has an internal reference to its parent element, and if it is a root element, it has an internal reference to the document (otherwise we could not implement the exception behaviour of AddXmlChildElement). When calling CopyXmlElement, we should specify that these upward references should not be deep-copied. For comparison: Java JDOM2 Element.clone(): > This returns a deep clone of this element. The new element is detached

	from > its parent, and getParent() on the clone will return null. > http://www.jdom.org/docs/apidocs/org/jdom2/Element.html#clone() .NET XmlNode.Clone(): > Clones the element node, its attributes, and any child nodes. > https://docs.microsoft.com/en-us/dotnet/api/system.xml.xmlnode.clone?view=netframework-4.8 Web API Node.cloneNode(): > The newClone has no parent and is not part of the document, until it is added > to another node that is part of the document (using Node.appendChild() or a > similar method). > https://developer.mozilla.org/en-US/docs/Web/API/Node/cloneNode Suggested new specification: > Returns a deep copy of an XmlElement. The returned XmlElement has no parent > element, and it does not belong to a document.
ID	Title
4289	otxXml:SetXmlComment behaviour on "--"
	Description
	In XML comments, the character sequence "--" (double hyphen) is forbidden, due to some obscure SGML backwards compatibility. How should otxXml:SetXmlComment behave if the given string contains double hyphens? It could even be that the root element already had a comment ending with a hyphen. In this case, appending a comment starting with a hyphen would produce invalid XML, e.g.: SetXmlComment(append=false, comment="Test-", myDoc) SetXmlComment(append=true, comment="-String", myDoc) This would produce the following invalid XML: <!--Test--String--> <myRootElement/> Option 1: Specify that an exception shall be thrown: Exceptions: XmlChangeException • If this term would have caused the resulting comment to contain a sequence of two hyphens ("--"), which is illegal in XML comments. Option 2: Specify that double hyphens shall be escaped: If the resulting comment would have a sequence of two hyphens ("--"), these hyphens shall be replaced by numeric character references (e.g. "--") to ensure that the string is a valid XML comment.

ID	Title
4290	Where should otxXml:SetXmlElementText insert the text?
	Description
	If the XmlElement has child elements: Where should the text be added? Example 1: <root><child1/><child2/></root> SetXmlElementText(rootElement, "foobar") Option 1a: before the first child element: <root>foobar<child1/><child2/></root> Option 1b: after the last child element: <root><child1/><child2/>foobar</root> Furthermore, if there was already text before and after the child elements, shall both texts be "overwritten"? Example 2: <root>AAA<child1>BBB</child1><child2>CCC</child2>DDD</root> SetXmlElementText(rootElement, "foobar") What is the result?
ID	Title
4291	otxXml: Behavior when illegal characters prevent saving
	Description
	Example 1: root = CreateXmlElement("BœufMorte") SetXmlElementText(root, "€") doc = CreateXmlDocument(root, encoding=ISO-8859-1) bf = XmlToByteField(doc) Example 2 (slightly different order, and last line differs): root = CreateXmlElement("BœufMorte") doc = CreateXmlDocument(root, encoding=ISO-8859-1) SetXmlElementText(root, "€") bf = XmlSaveToFile(doc, "file:///C:/foo.xml") Note: Neither the oe ligature (œ) nor the Euro sign (€) can be represented in ISO-8859-1. In the case of element text, escaping the character as € would be an option, but that doesn't solve the problem of the œ in the element name. Suggestion: Add to otxXml:XmlToByteField: Exceptions: XmlFormatException * If the content cannot be represented in the encoding of the XmlDocument. Insert in otxXml:XmlSaveToFile: Exceptions: XmlFormatException * If the content cannot be represented in the encoding of the XmlDocument.

	file:FileSaveException * If saving failed.
ID	Title
4295	otxXml:ValidateXml fallbackEncoding parameter should be optional
	Description
	Like in the other actions/terms that have a fallbackEncoding, otxXml:ValidateXml specifies it as: > Optional encoding value of the XML document (Default = "UTF-8") However, it has cardinality [1..1]. A schema change is necessary to change it to [0..1].
ID	Title
4297	Reformulate Persistence_Chk001 – PersistableDataTypes
	Description
	The checker rule Persistence_Chk001 – PersistableDataTypes currently has this criterion: > Used data types shall be persistable, otherwise an exchange is not guaranteed. The only place that describes which data types are persistable are the three requirements P01, P02 and P03 (I find it weird that requirement texts are used for detailed normative specifications). There is no statement anywhere that all other data types are non-persistable. With this, I could decide to make it possible in our runtime system to persist e.g. a diag:Parameter, and to allow diag:Parameter in our implementation of Persistence_Chk001. But that would mean that there is no objective validation anymore. I suggest to reformulate the criterion and to explicitly name the persistable data types there: The data type of the variable (referenced by the <value> element) shall be a persistable data type. Only the following data types are persistable: - All subtypes of otx:SimpleType - otx:ByteField - Data types defined in an extension which states that they can be persisted - otx:List, if its item type is persistable - otx:Map, if its key type and value type are persistable
ID	Title
4298	otxXml:ValidateXml exception behaviour if schema invalid
	Description
	otxXml:ValidateXml does not specify what to do if one of the referenced schema files is not a valid XML schema (e.g. if it is not well-formed, or if its root element is unlike <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" ...>. Suggestion: amend the existing XmlFormatException specification: otxXml:XmlFormatException • If an invalid encoding (BIN, OCT, HEX) is specified, or if one of the referenced schema files is not a valid XML schema
ID	Title
4302	ComInterface.GetComInterfaceNameListFromEthernet wrong MVCI system

	operation
	Description
	The current specification of GetComInterfaceNameListFromEthernet is Returns the names (MCDInterface unique SHORT-NAME) of the ComInterface as a ListTerm of String. NOTE — The appropriate MVCI system operation is MCDInterface::detectInterfaces(). Problem is that two methods to detect interfaces exists. One at MCDSystem level and one at MCDInterface level. From our point of view the method at system level should be called and not the method at MCDSystem level. Therfore the NOTE should be changed as follows: NOTE — The appropriate MVCI system operation is MCDSystem::detectInterfaces().
ID	Title
4305	otxXml:AddXmlChildElement and DeleteXmlChildElement refer to wrong exception
	Description
	AddXmlChildElement specifies: > If this element has already been added to another parent or it is used as a > root node, the XmlException shall be thrown and the element is not added to the > parent. DeleteXmlChildElement specifies: > If the given child element is not a child of the given parent, the XmlException > shall be thrown and no deletion shall take place. But according to their "Exceptions" sections, the actions throw an XmlChangeException, not an XmlException. We should change the two sentences so that they say "an XmlChangeException shall be thrown".
ID	Title
4308	otxXml:CreateXmlDocument should throw exception if root node is already in use
	Description
	In the AddXmlChildElement, we took great care to make sure that no element occurs twice within the document, or within two documents: > Exceptions: XmlChangeException - If this element has already been added to another parent or it is used as a root node. However, in CreateXmlDocument we fail to check whether the given root element is already in use. This is problematic because some XML libraries don't allow an element to be used twice. Suggestion: Add this to the Exceptions block of CreateXmlDocument:

	XmlChangeException - If the given element is already the root element of another document, or if it has a parent element.
ID	Title
4309	busMon:StopBusMonitoring unspecified if variable uninitialised or monitor already stopped
	Description
	The StopBusMonitoring action does not specify what to do if the given variable is uninitialised (because it has never been initialised, or because it has been stopped before). Suggestion: In analogy to diag:CloseComChannel (see correction sheet) and event:CloseEventSource, specify that StopBusMonitoring should perform nothing (NOP) in that case.
ID	Title
4310	busMon:BusMonitorEventSource, IsBusMonitorEvent can throw InvalidReferenceException
	Description
	In busMon:BusMonitorEventSource and busMon:IsBusMonitorEvent, we must specify that an InvalidReferenceException shall be thrown if no value has been assigned to the given variable, or if the monitor has already been stopped.
ID	Title
4311	comIf:CloseComInterface unspecified if variable uninitialised or interface already closed
	Description
	The CloseComInterface action does not specify what to do if the given variable is uninitialised, or if the interface has been closed before. Suggestion: In analogy to diag:CloseComChannel (see correction sheet) and event:CloseEventSource, specify that CloseComInterface should perform nothing (NOP) in that case.

About This Document

This document lists known issues for the standard and version as identified in the document header. Issues in the context of ASAM standards have one of the following characteristics:

- Error: unintended or wrong content.
- Contradiction: inconsistent or contradictory content.
- Specification gap: missing content required for a functional system and for complete understanding.
- Lack of clarity: Unclear, vague or ambiguous description, which leads to misunderstandings and misinterpretations.

The issue may exist in the base standard, in associate standards, schema files, interface definition files, model files, examples or any other supplements of the standard.

For each issue, the table contains an ID, title and description.

ID: Unique identification number assigned by the ASAM change request system.

Title: Summary of the issue description in headline style

Description: Identifies the parts of the standard that are affected by the issue, provides a reason why this is considered as an issue and allows the reader to understand the technical implications of the issue. Optionally, the description includes a resolution proposal and a proposed workaround for the issue.

Issue are resolved in the release of a new version of a standard. Please regularly check ASAM's web page and news publications to stay informed about new versions. If an issue has been resolved in a new version, then it is not listed in the List of Known Issues document for this version any longer.

The List of Known Issues document for former versions of the same standard will be frozen and will not be further maintained. ASAM advises all users of its standards to always use the latest version of its standards.