Document Title	Log And Trace Extract Template
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	1024

Document Status	published
Part of AUTOSAR Standard	Foundation
Part of Standard Release	R24-11

Document Change History

Date	Release	Changed by	Description
2024-11-27	R24-11	AUTOSAR Release Management	• Adapted definition of Strings as networkRepresentation of a DltArgument • Improved descriptions of examples • Added imposition times to constraints
2023-11-23	R23-11	AUTOSAR Release Management	• Changed existing statements into formal constraints; for details please see the change history • editorial changes
2022-11-24	R22-11	AUTOSAR Release Management	• Added modeling support for Privacy Flags
2021-11-25	R21-11	AUTOSAR Release Management	• Initial Release

Disclaimer

Contents

References

[1] Log and Trace Protocol Specification
AUTOSAR_FO_PRS_LogAndTraceProtocol

[2] System Template
AUTOSAR_CP_TPS_SystemTemplate

[3] Specification of Manifest
AUTOSAR_AP_TPS_ManifestSpecification

[4] Software Component Template
AUTOSAR_CP_TPS_SoftwareComponentTemplate

[5] Generic Structure Template
AUTOSAR_FO_TPS_GenericStructureTemplate

1 Introduction

The Log and Trace functionality (DLT) in AUTOSAR allows Classic Platform SW-Cs/Modules, Adaptive Platform Applications/Functional Clusters to forward log and trace information onto external Dlt Log Viewers if supported on the underlying platform.

The format of the DLT messages is described in more detail in the Log and Trace Protocol Specification [1].

Please note that the configuration of the source of log and trace messages (application or module that produces the logging information) is described in Chapter 2.

The ARXML format of `DltMessage`s is described in Chapter 3. This format can be used in Classic Platform System Template [2] descriptions and Adaptive Platform Manifest [3] descriptions to define log and trace messages sent out by Classic Platform SW-Cs/Modules or Adaptive Platform Applications/Functional Clusters.

The description of the `DltMessage` is particularly important to support the Non-Verbose Mode of the Log And Trace Protocol. To reduce the amount of traffic on the bus, no meta data about the interpretation of the `DltMessage` is transmitted on the communications bus in the Non-Verbose Mode. Instead, the ARXML file that describes the `DltMessage` holds the information about the interpretation of the `DltMessage`.

The following figure shows the placement of the Log and Trace Extract in the AUTOSAR Methodology.

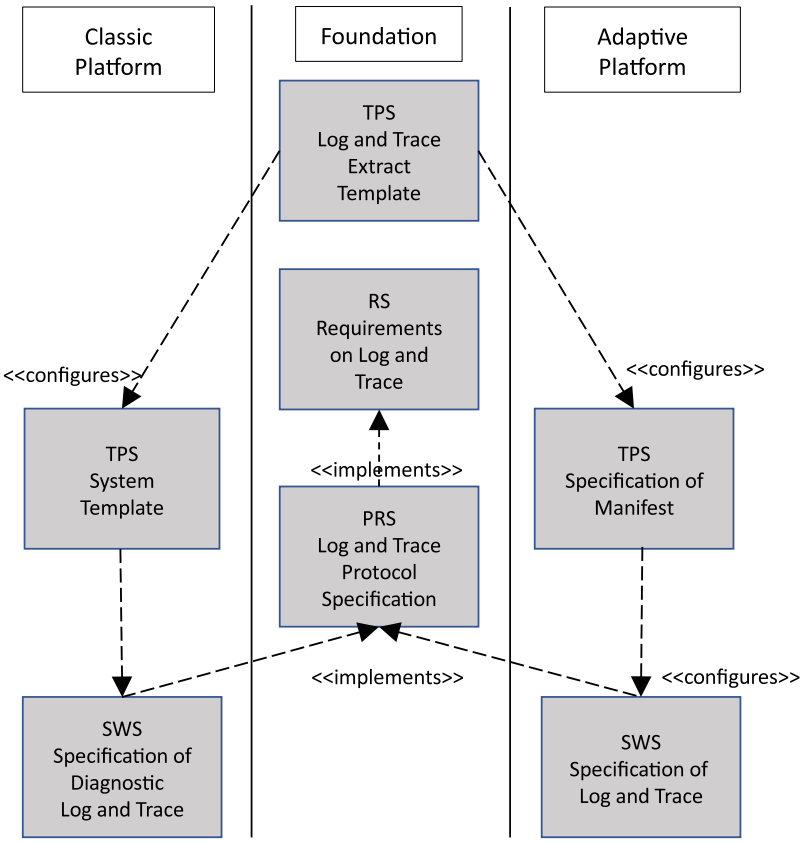

Figure 1.1: Methodological placement of Log and Trace Extract

1.1 Document Conventions

Technical terms are typeset in mono spaced font, e.g. `PortPrototype`. As a general rule, plural forms of technical terms are created by adding "s" to the singular form, e.g. `PortPrototype`s. By this means the document resembles terminology used in the AUTOSAR XML Schema.

This document contains constraints in textual form that are distinguished from the rest of the text by a unique numerical constraint ID, a headline, and the actual constraint text starting after the ⌈ character and terminated by the ⌋ character.

The purpose of these constraints is to literally constrain the interpretation of the AUTOSAR meta-model such that it is possible to detect violations of the standardized behavior implemented in an instance of the meta-model (i.e. on M1 level).

Makers of AUTOSAR tools are encouraged to add the numerical ID of a constraint that corresponds to an M1 modeling issue as part of the diagnostic message issued by the tool.

The attributes of the classes introduced in this document are listed in form of class tables. They have the form shown in the example of the top-level element AUTOSAR:

Please note that constraints are not supposed to be enforceable at any given time in an AUTOSAR workflow. During the development of a model, constraints may legitimately be violated because an incomplete model will obviously show inconsistencies.

However, at specific points in the workflow, constraints shall be enforced as a safeguard against misconfiguration.

The points in the workflow where constraints shall be enforced, sometimes also known as the "binding time" of the constraint, are different for each model category, e.g. on the classic platform, the constraints defined for software-components are typically enforced prior to the generation of the RTE while the constraints against the definition of an Ecu extract shall be applied when the Ecu configuration for the Com stack is created.

For each document, possible binding times of constraints are defined and the binding times are typically mentioned in the constraint themselves to give a proper orientation for implementers of AUTOSAR authoring tools.

Let `AUTOSAR` be an example of a typical class table. The first rows in the table have the following meaning:

Class: The name of the class as defined in the UML model.

Package: The UML package the class is defined in. This is only listed to help locating the class in the overall meta model.

Note: The comment the modeler gave for the class (class note). Stereotypes and UML tags of the class are also denoted here.

Base Classes: If applicable, the list of direct base classes.

The headers in the table have the following meaning:

Attribute: The name of an attribute of the class. Note that AUTOSAR does not distinguish between class attributes and owned association ends.

Type: The type of an attribute of the class.

Mul.: The assigned multiplicity of the attribute, i.e. how many instances of the given data type are associated with the attribute.

Kind: Specifies, whether the attribute is aggregated in the class (`aggr` aggregation), an UML attribute in the class (`attr` primitive attribute), or just referenced by it (`ref` reference). Instance references are also indicated (`iref` instance reference) in this field.

Note: The comment the modeler gave for the class attribute (role note). Stereotypes and UML tags of the class are also denoted here.

Please note that the chapters that start with a letter instead of a numerical value represent the appendix of the document. The purpose of the appendix is to support the explanation of certain aspects of the document and does not represent binding conventions of the standard.

2 Dlt Ecu

This chapter describes the DltEcu that represents the Classic Platform “ECU” or Adaptive Platform “Machine” on which DltApplications are deployed that are generating DltMessages. The DltApplication and DltContext are used to identify the source of the log and trace message.

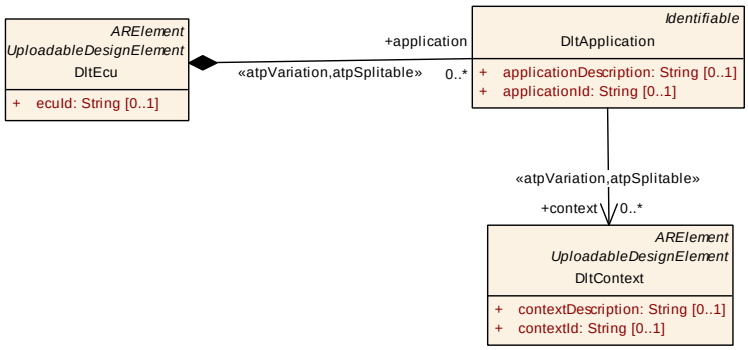

Figure 2.1: Modeling of DltEcu

Class	**DltEcu**			
Package	M2::AUTOSARTemplates::LogAndTraceExtract			
Note	This element represents an Ecu or Machine that produces logging and tracing information. **Tags:** atp.recommendedPackage=DltEcus			
Base	*ARElement*, *ARObject*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *Packageable Element*, *Referrable*, *UploadableDesignElement*, *UploadablePackageElement*			
Aggregated by	ARPackage.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
application	DltApplication	*	aggr	Application on DltEcu that provides log or trace data. **Stereotypes:** atpSplitable; atpVariation **Tags:** atp.Splitkey=application.shortName, application.variationPoint.shortLabel vh.latestBindingTime=systemDesignTime
ecuId	String	0..1	attr	This attribute defines the name of the ECU for use within the Dlt protocol.

Table 2.1: DltEcu

Class	**DltApplication**			
Package	M2::AUTOSARTemplates::LogAndTraceExtract			
Note	This meta-class represents the application from which the log and trace message originates.			
Base	*ARObject*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	DltEcu.application			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
applicationDescription	String	0..1	attr	This attribute can be used to describe the applicationId that is used in the log and trace message in more detail.
applicationId	String	0..1	attr	This attribute identifies the SW-C/BSW module in the log and trace message.

▽

△

Class	**DltApplication**			
context	DltContext	*	ref	Definition of ContextIds for the Application. **Stereotypes:** atpSplitable; atpVariation **Tags:** atp.Splitkey=context.dltContext, context.variation Point.shortLabel vh.latestBindingTime=systemDesignTime

Table 2.2: DltApplication

Class	**DltContext**			
Package	M2::AUTOSARTemplates::LogAndTraceExtract			
Note	This meta-class represents the Context that groups Log and Trace Messages that are generated by an application. **Tags:** atp.recommendedPackage=DltContexts			
Base	*ARElement*, *ARObject*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *Packageable Element*, *Referrable*, *UploadableDesignElement*, *UploadablePackageElement*			
Aggregated by	ARPackage.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
context Description	String	0..1	attr	This attribute can be used to describe the contextId that is used in the log and trace message in more detail.
contextId	String	0..1	attr	This attribute is used to group log and trace messages produced by an application to distinguish functionality.
dltMessage	DltMessage	*	ref	Group of Log and Trace Messages assigned to the Dlt Context **Stereotypes:** atpSplitable; atpVariation **Tags:** atp.Splitkey=dltMessage.dltMessage, dlt Message.variationPoint.shortLabel vh.latestBindingTime=systemDesignTime

Table 2.3: DltContext

[constr_5294] Existence of `DltEcu.ecuId`

Imposition time: `IT_LogTrace`

⌈For each `DltEcu`, the attribute `ecuId` shall exist when the Log And Trace Extract is created.⌋

[constr_5295] Existence of `DltApplication.context`

Imposition time: `IT_LogTrace`

⌈Each `DltApplication` shall reference at least one `DltContext` in the role `context` when the Log And Trace Extract is created.⌋

[constr_5296] Existence of `DltApplication.applicationId`

Imposition time: `IT_LogTrace`

⌈For each `DltApplication`, the attribute `applicationId` shall exist when the Log And Trace Extract is created.⌋

[constr_5297] Existence of `DltApplication.applicationDescription`

Imposition time: `IT_LogTrace`

⌈For each `DltApplication`, the attribute `applicationDescription` shall exist when the Log And Trace Extract is created.⌋

[constr_5298] Existence of `DltContext.contextId`

Imposition time: `IT_LogTrace`

⌈For each `DltContext`, the attribute `contextId` shall exist when the Log And Trace Extract is created.⌋

[constr_5299] Existence of `DltContext.contextDescription`

Imposition time: `IT_LogTrace`

⌈For each `DltContext`, the attribute `contextDescription` shall exist when the Log And Trace Extract is created.⌋

[constr_5300] Existence of `DltContext.dltMessage`

Imposition time: `IT_LogTrace`

⌈Each `DltContext` shall reference at least one `DltMessage` in the role `dltMessage` when the Log And Trace Extract is created.⌋

3 Dlt Message

This chapter describes the modeling of DltMessages. Please note that some information of the Dlt message format that is described in [1] like the Message Info and the Number of Arguments can be derived from the DltMessage itself. Other parts in the Dlt message format represent the source of the log or trace messages (e.g. ecuId, applicationId, contextId) and this information can be derived from the DltEcu, the DltApplication that is aggregated by the DltEcu and the DltContext that is referenced by the DltApplication.

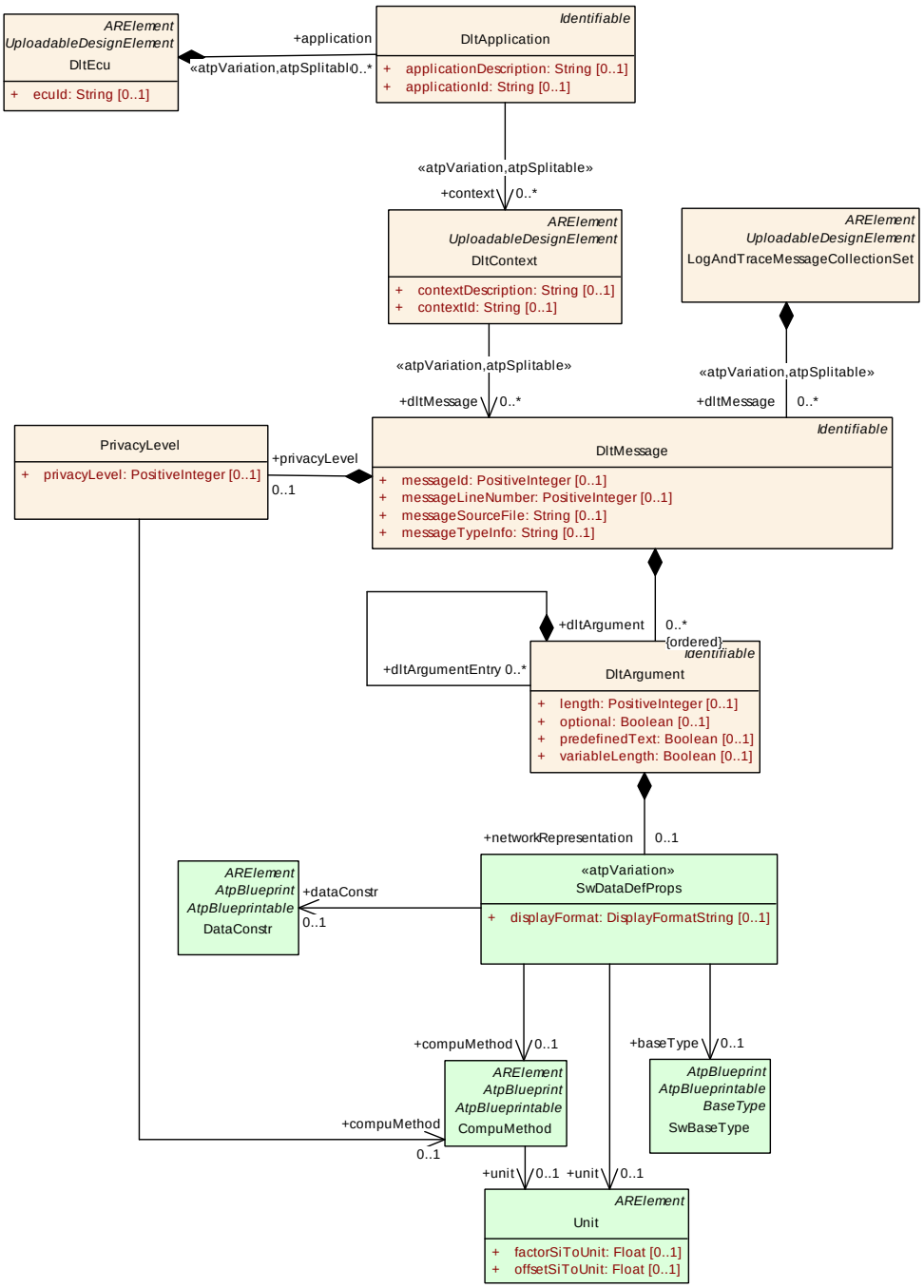

Figure 3.1: Modeling of DltMessages

Class	LogAndTraceMessageCollectionSet			
Package	M2::AUTOSARTemplates::LogAndTraceExtract			
Note	Collection of DltMessages **Tags:** atp.recommendedPackage=LogAndTraceMessageCollectionSets			
Base	*ARElement*, *ARObject*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *Packageable Element*, *Referrable*, *UploadableDesignElement*, *UploadablePackageElement*			
Aggregated by	ARPackage.element			
Attribute	**Type**	**Mult.**	**Kind**	**Note**
dltMessage	DltMessage	*	aggr	Collection of DltMessages in the DltMessageCollection Set. **Stereotypes:** atpSplitable; atpVariation **Tags:** atp.Splitkey=dltMessage.shortName, dltMessage.variationPoint.shortLabel vh.latestBindingTime=systemDesignTime

Table 3.1: LogAndTraceMessageCollectionSet

Class	DltMessage			
Package	M2::AUTOSARTemplates::LogAndTraceExtract			
Note	This element defines a DltMessage.			
Base	*ARObject*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	LogAndTraceMessageCollectionSet.dltMessage			
Attribute	**Type**	**Mult.**	**Kind**	**Note**
dltArgument (ordered)	DltArgument	*	aggr	Ordered collection of DltArguments in the DltMessage.
messageId	PositiveInteger	0..1	attr	This attribute defines the unique Id for the DltMessage.
messageLine Number	PositiveInteger	0..1	attr	This attribute describes the position in the source file in which this log message was called.
messageSource File	String	0..1	attr	This attribute describes the source file in which this log message was called.
messageType Info	String	0..1	attr	This attribute describes the message Type
privacyLevel	PrivacyLevel	0..1	aggr	The Privacy Level helps to identify the Log and Trace content towards the degree of privacy to it.

Table 3.2: DltMessage

[TPS_DLTXT_00001] Log or trace message representation ⌈One log or trace message is represented by one `DltMessage` element.⌋

[constr_5301] Existence of `DltMessage.messageId`

Imposition time: `IT_LogTrace`

⌈For each `DltMessage`, the attribute `messageId` shall exist when the Log And Trace Extract is created.⌋

[TPS_DLTXT_00002] DLT message ⌈The `DltMessage` is described with the following information:

- Message Type (MSTP) - shall be derived from `DltMessage.messageTypeInfo`

- Message Info (MSIN) - shall be derived from DltMessage.messageTypeInfo
- Source file - shall be derived from DltMessage.messageSourceFile
- Line Number - shall be derived from DltMessage.messageLineNumber

⌋

Please note that the source file and the line number described in [TPS_DLTXT_00002] are not applicable to standardized DltMessages or to DltMessages that are meant to be reused in multiple locations.

Class	**DltArgument**			
Package	M2::AUTOSARTemplates::LogAndTraceExtract			
Note	This element defines an Argument in a DltMessage.			
Base	*ARObject*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	DltArgument.dltArgumentEntry, DltMessage.dltArgument			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
dltArgument Entry	DltArgument	*	aggr	This aggregation is used to describe subElements of a Dlt Argument that defines a Structure.
length	PositiveInteger	0..1	attr	Describes the DltArgument length in case of Arrays and Strings in number of BaseTypes.
network Representation	SwDataDefProps	0..1	aggr	Definition of the networkRepresentation of the Dlt Argument.
optional	Boolean	0..1	attr	This attribute defines whether the argument is optional or not. If set to true, the argument can be omitted from the payload of a DLT message.
predefinedText	Boolean	0..1	attr	This attribute defines whether the DltArgument is a predefinedText (Static Data).
variableLength	Boolean	0..1	attr	This attribute defines whether the length of the Dlt Argument is variable (determined at runtime) or not.

Table 3.3: DltArgument

[TPS_DLTXT_00003] User data of the log or trace message ⌈The user data of the log or trace message shall be represented by DltArguments that are ordered in the DltMessage.⌋

[TPS_DLTXT_00004] DltArgument kinds ⌈Each DltArgument can be either:

- predefined text (static)
- assembled data

⌋

Please note that the Log and Trace Protocol Specification [1] is using the terms “Static-Data” for predefined text and “Non-Static-Data” for assembled data.

[TPS_DLTXT_00005] predefined text DltArgument ⌈If a DltArgument is a predefined text, then the DltArgument shall only contain a shortName, a desc and the predefinedText. The text shall be placed into the desc field. The predefinedText attribute of the DltArgument shall be set to true.⌋

[TPS_DLTXT_00006] assembled data DltArgument ⌈If a DltArgument represents assembled data then DltArgument.networkRepresentation.unit and the DltArgument.networkRepresentation.baseType shall be defined.⌋

Please note that only the assembled data of the DltMessage is send to external Dlt Log Viewers. The "predefined text" is available only in the ARXML File.

[TPS_DLTXT_00014] Semantics of DltArgument.optional ⌈The optional attribute defines whether the DltArgument is optional (true) or mandatory (false) in the Dlt payload.⌋

[constr_5302] Restriction in usage of DltArgument.optional attribute

Imposition time: IT_LogTrace

⌈The optional attribute shall not be set in a DltArgument that represents an array dimension.⌋

The meaning of the term "array dimension" in [constr_5302] refers to [constr_5364].

[TPS_DLTXT_00009] Semantics of DltArgument.length ⌈The length attribute defines the length of the DltArgument of an Array or a String. If the DltArgument.variableLength is set to true this attribute defines the maximum length of the DltArgument.⌋

Please note that for primitive types like Boolean, Signed Integer, Unsigned Integer, and Float the length of the DltArgument shall be derived from the baseTypeSize.

[constr_5303] Restriction of baseTypeSize of a DltArgument

Imposition time: IT_LogTrace

⌈The baseTypeSize in the networkRepresentation of a DltArgument is restricted to 8, 16, 32, and 64 Bits.⌋

[TPS_DLTXT_00010] Semantics of DltArgument.length in case of a String ⌈If the networkRepresentation of a DltArgument is a String which is described with a SwBaseType defined according to [TPS_DLTXT_00007] then the length attribute specifies the number of UTF-8/UTF-16 code points in the String. Please note the size in bytes depends on the encoding in the corresponding SwBaseType.⌋

[TPS_DLTXT_00011] Description of DltArgument with one-dimensional Array type ⌈A DltArgument that has a one-dimensional Array type is described by a DltArgument that has the length attribute set to a value. The data type of the array is described by the SwBaseType in the networkRepresentation.⌋

Please note that a DltArgument of type String is described as an Array of characters.

[TPS_DLTXT_00012] Description of DltArgument with n-dimensional Array type ⌈A DltArgument that has a n-dimensional Array type is described by a DltArgument that aggregates further DltArguments in the role dltArgumentEntry and all the aggregated DltArguments have the length attribute set to a value. Each dltArgumentEntry defines one dimension in the Array that is represented by the aggregating DltArgument.

The length attribute of the aggregating DltArgument defines the number of dimensions.

The length attribute in the dltArgumentEntry defines the number of entries in the dimension.

The data type of the Array is described by the SwBaseType in the networkRepresentation of the aggregating DltArgument.⌋

[constr_5304] Datatype of an Array

Imposition time: IT_LogTrace

⌈The dltArgumentEntry that is aggregated by a DltArgument that has the length attribute set to a value (represents an Array) shall not define a SwBaseType in the networkRepresentation since the data type of the Array is described by the SwBaseType in the networkRepresentation of the aggregating DltArgument.⌋

[TPS_DLTXT_00013] Description of DltArgument with Structure type ⌈A DltArgument that has a Structure type is described by a DltArgument that aggregates further DltArguments in the role dltArgumentEntry and that does not have the length attribute set to a value.⌋

The following table summarizes the modeling of the different Payload data types:

[constr_5363] Allowed usage of attributes for description of payload data types

Imposition time: IT_LogTrace

⌈

Type	length	dltArgumentEntry	SwBaseType of top level DltArgument
Predefined Text	NA	NA	NA
primitive Type	NA	NA	D
String	D	NA	D
1-dimensional Array	D	NA	D
n-dimensional Array	D	D	D
Struct	NA	D	NA

⌋

The following table summarizes the usage of `dltArgumentEntry`:

[constr_5364] Allowed usage of attributes in case of a `dltArgumentEntry`

Imposition time: IT_LogTrace

⌈

DltArgumentEntry type	length	dltArgumentEntry	SwBaseType of DltArgumentEntry
Struct member	NA	D	D
Array dimension	D	D	NA

⌋

The following settings apply in [constr_5363] and [constr_5364]:

D Attribute can be **defined** in the scope of this element.

NA Attribute is **not applicable** for usage in the scope of this element.

[constr_5098] Allowed SwDataDefProps attributes for DltArgument.networkRepresentation

Imposition time: IT_LogTrace

⌈

Attributes of SwDataDefProps	networkRepresentation
annotation	N/A
baseType	D
compuMethod	D
dataConstr	D
displayFormat	D
displayPresentation	N/A
invalidValue	N/A
swComparisonVariable	N/A
swHostVariable	N/A
swTextProps	D
unit	D

⌋

Please note that the set of attributes of SwDataDefProps listed in [constr_5098] represents the subset that is visible in the *AUTOSAR foundation*.

The following settings apply in [constr_5098]:

D Attribute can be **defined** in the scope of this element.

NA Attribute is **not applicable** for usage in the scope of this element.

Please note that the DltMessage is not necessary transmitted over the network, even if the DltArgument defines a networkRepresentation. The logging information may be provided onto the communication bus, the console, or to the file system.

The dataConstr attribute can be used to describe value ranges of the DltArgument. More details about Data Constraints can be found in the Software Component Template [4]. Please note that the Data Constraints are irrelevant for the Dlt Protocol.

The displayFormat attribute defines a format specifier for the display of values in tools. More details about the potential value settings can be found in the Generic Structure Template [5].

[constr_5305] CompuMethod in DltArgument.networkRepresentation

Imposition time: IT_LogTrace

⌈The CompuMethod that is used in the networkRepresentation of a DltArgument is limited to category TEXTTABLE.⌋

With the TEXTTABLE CompuMethod it is possible to define an enumeration as networkRepresentation of a DltArgument. More details can be found in the Software Component Template [4].

[TPS_DLTXT_00007] Description of a String as networkRepresentation of a DltArgument ⌈If the networkRepresentation of a DltArgument is a String this can be described with:

- swTextProps that references a SwBaseType in the role baseType that in turn defines the encoding of a UTF-8/UTF-16 code point
- baseType that defines the underlying storage of a UTF-8/UTF-16 code point

⌋

[TPS_DLTXT_00008] Standardized values of DltMessage.messageTypeInfo ⌈The following values of attributes DltMessage.messageTypeInfo are standardized by AUTOSAR:

- DLT_LOG_OFF
- DLT_LOG_FATAL
- DLT_LOG_ERROR
- DLT_LOG_WARN
- DLT_LOG_INFO
- DLT_LOG_DEBUG
- DLT_LOG_VERBOSE
- DLT_TRACE_VARIABLE
- DLT_TRACE_FUNCTION_IN
- DLT_TRACE_FUNCTION_OUT
- DLT_TRACE_STATE
- DLT_TRACE_VFB
- DLT_NW_TRACE_IPC
- DLT_NW_TRACE_CAN
- DLT_NW_TRACE_FLEXRAY
- DLT_NW_TRACE_MOST
- DLT_NW_TRACE_ETHERNET
- DLT_NW_TRACE_SOMEIP

- DLT_NW_TRACE_0x7
- DLT_NW_TRACE_0x8
- DLT_NW_TRACE_0x9
- DLT_NW_TRACE_0x10
- DLT_NW_TRACE_0x11
- DLT_NW_TRACE_0x12
- DLT_NW_TRACE_0x13
- DLT_NW_TRACE_0x14
- DLT_NW_TRACE_0x15
- DLT_CONTROL_REQUEST
- DLT_CONTROL_RESPONSE

⌋

Please note that only the values defined in [TPS_DLTXT_00008] are allowed to be used in `DltMessage.messageTypeInfo`. Custom values are currently not allowed.

Class	**PrivacyLevel**			
Package	M2::AUTOSARTemplates::LogAndTraceExtract			
Note	This meta-class defines the Privacy Level for a Log and Trace content.			
Base	*ARObject*			
Aggregated by	DltMessage.privacyLevel			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
compuMethod	CompuMethod	0..1	ref	Reference to CompuMethod of category TEXTTABLE that defines the supported user-defined privacy levels.
privacyLevel	PositiveInteger	0..1	attr	The value that represents the privacy level and is transported in the Extension Header.

Table 3.4: PrivacyLevel

[constr_5340] Range of `DltMessage.privacyLevel.privacyLevel`

Imposition time: `IT_LogTrace`

⌈The value of `DltMessage.privacyLevel.privacyLevel` shall be in the range between 0 and 255.⌋

[constr_5341] Range of `PrivacyLevel.compuMethod`

Imposition time: `IT_LogTrace`

⌈The `CompuMethod` that is referenced from `PrivacyLevel` in the role `compuMethod` shall have the category TEXTTABLE.⌋

A Examples

This chapter provides some examples for the modeling of model elements discussed in the scope of this document.

A.1 Position in Source Code

A DltMessage can provide information to the position in the source code (in this example: line 72 in demo.c) to which it is affiliated, see Listing A.1.

```
<MESSAGE-ID>1</MESSAGE-ID>
<MESSAGE-LINE-NUMBER>72</MESSAGE-LINE-NUMBER>
<MESSAGE-SOURCE-FILE>demo.c</MESSAGE-SOURCE-FILE>
<MESSAGE-TYPE-INFO>DLT_LOG_DEBUG</MESSAGE-TYPE-INFO>
```

Listing A.1: Example for the definition of the position in source code

A.2 Predefined Text

The definition of a DltArgument that represents predefined text is depicted in the example in Listing A.2. Please note the value of attribute predefinedText.

```
<DLT-ARGUMENT>
  <SHORT-NAME>Temperature_measurement</SHORT-NAME>
  <DESC>
    <L-2 L="EN">Temperature measurement</L-2>
  </DESC>
  <PREDEFINED-TEXT>true</PREDEFINED-TEXT>
</DLT-ARGUMENT>
```

Listing A.2: Example for the definition of predefined text

A.3 Example of DLT Arguments for Temperature Measurement

In this section, example definitions of DltArguments for the purpose of temperature measurement are discussed.

Of course, there is nothing special in the definition of DltArguments for temperature measurement, i.e., these examples could directly be mapped to other quantities.

A.3.1 Measurement without Unit

The definition of a DltArgument that represents a **measurement without a unit** is depicted in the example in Listing A.3. The indication that no unit shall be used in this case is provided by the reference to the Unit with the shortName set to “NoUnit”.

```
<DLT-ARGUMENT>
  <SHORT-NAME>measurement_point</SHORT-NAME>
  <DESC>
    <L-2 L="EN">Temperature measurement</L-2>
  </DESC>
  <NETWORK-REPRESENTATION>
    <SW-DATA-DEF-PROPS-VARIANTS>
      <SW-DATA-DEF-PROPS-CONDITIONAL>
        <BASE-TYPE-REF DEST="SW-BASE-TYPE">/AUTOSAR_Platform/BaseTypes/
            uint8</BASE-TYPE-REF>
        <UNIT-REF DEST="UNIT">/PhysicalUnits/Units/NoUnit</UNIT-REF>
      </SW-DATA-DEF-PROPS-CONDITIONAL>
    </SW-DATA-DEF-PROPS-VARIANTS>
  </NETWORK-REPRESENTATION>
</DLT-ARGUMENT>
```

Listing A.3: Example for a measurement without a unit

The corresponding SwBaseType used for the DltArgument in this example is depicted in Listing A.4.

```
<SW-BASE-TYPE>
  <SHORT-NAME>uint8</SHORT-NAME>
  <CATEGORY>FIXED_LENGTH</CATEGORY>
  <BASE-TYPE-SIZE>8</BASE-TYPE-SIZE>
  <BASE-TYPE-ENCODING>NONE</BASE-TYPE-ENCODING>
</SW-BASE-TYPE>
```

Listing A.4: Example for a measurement without a unit

A.3.2 Measurement with Reference to a Unit

The definition of a DltArgument that represents a **measurement with a unit** is depicted in the example in Listing A.5. The difference to the example presented in Section A.3.1 is the reference to the Unit with the shortName set to “Kelvin”.

```
<DLT-ARGUMENT>
  <SHORT-NAME>reading</SHORT-NAME>
  <DESC>
    <L-2 L="EN">reading</L-2>
  </DESC>
  <NETWORK-REPRESENTATION>
    <SW-DATA-DEF-PROPS-VARIANTS>
      <SW-DATA-DEF-PROPS-CONDITIONAL>
        <BASE-TYPE-REF DEST="SW-BASE-TYPE">/AUTOSAR_Platform/BaseTypes/
            float32</BASE-TYPE-REF>
        <UNIT-REF DEST="UNIT">/PhysicalUnits/Units/Kelvin</UNIT-REF>
      </SW-DATA-DEF-PROPS-CONDITIONAL>
    </SW-DATA-DEF-PROPS-VARIANTS>
  </NETWORK-REPRESENTATION>
</DLT-ARGUMENT>
```

Listing A.5: Example for a measurement with a unit

The corresponding SwBaseType used for the DltArgument in this example is depicted in Listing A.6.

```
<SW-BASE-TYPE>
  <SHORT-NAME>float32</SHORT-NAME>
  <CATEGORY>FIXED_LENGTH</CATEGORY>
  <BASE-TYPE-SIZE>32</BASE-TYPE-SIZE>
  <BASE-TYPE-ENCODING>IEEE754</BASE-TYPE-ENCODING>
</SW-BASE-TYPE>
```

Listing A.6: Example for a measurement without a unit

A.3.3 Measurement that describes the Temperature Distribution in a Room

The definition of a DltArgument that that describes the temperature distribution (as a three-dimensional array) in a room is depicted in the example in Listing A.7.

For this example, it is assumed that there are a number of probes distributed in a room for spatial temperature measurement:

- The number of probes in the in the distribution of the **height** of the room (z-coordinate) is 4.
- The number of probes in the in the distribution of the **depth** of the room (y-coordinate) is 5.
- The number of probes in the in the distribution of the **width** of the room (x-coordinate) is 6.

In other words, the DltArgument needs to carry 120 individual measurements in total.

```
<DLT-ARGUMENT>
  <SHORT-NAME>My_3dim_Array_4x5x6</SHORT-NAME>
  <DESC>
    <L-2 L="EN">Heat distribution map of MyRoom</L-2>
  </DESC>
  <DLT-ARGUMENT-ENTRYS>
    <DLT-ARGUMENT>
      <SHORT-NAME>FirstDim_ArrayOfLayers</SHORT-NAME>
      <DESC>
        <L-2 L="EN">HeatMap_z</L-2>
      </DESC>
      <LENGTH>4</LENGTH>
      <!-- The length attribute in the dltArgumentEntry defines the number
           of entries in the dimension. -->
    </DLT-ARGUMENT>
    <DLT-ARGUMENT>
      <SHORT-NAME>SecondDim_ArrayOfRows</SHORT-NAME>
      <DESC>
        <L-2 L="EN">HeatMap_y</L-2>
      </DESC>
      <LENGTH>5</LENGTH>
      <!-- The length attribute in the dltArgumentEntry defines the number
           of entries in the dimension. -->
```

```
    </DLT-ARGUMENT>
    <DLT-ARGUMENT>
      <SHORT-NAME>ThirdDim_ArrayOfProbes</SHORT-NAME>
      <DESC>
        <L-2 L="EN">HeatMap_x</L-2>
      </DESC>
      <LENGTH>6</LENGTH>
      <!-- The length attribute in the dltArgumentEntry defines the number
         of entries in the dimension. -->
    </DLT-ARGUMENT>
  </DLT-ARGUMENT-ENTRYS>
  <LENGTH>3</LENGTH>
  <!-- The length attribute in the DltArgument defines the number of
     Dimensions. -->
  <NETWORK-REPRESENTATION>
    <SW-DATA-DEF-PROPS-VARIANTS>
      <SW-DATA-DEF-PROPS-CONDITIONAL>
        <BASE-TYPE-REF DEST="SW-BASE-TYPE">/AUTOSAR_Platform/BaseTypes/
           float32</BASE-TYPE-REF>
        <UNIT-REF DEST="UNIT">/PhysicalUnits/Units/Kelvin</UNIT-REF>
      </SW-DATA-DEF-PROPS-CONDITIONAL>
    </SW-DATA-DEF-PROPS-VARIANTS>
  </NETWORK-REPRESENTATION>
</DLT-ARGUMENT>
```

Listing A.7: Example for a measurement of temperatur distribution

A.4 Example of a DLT Argument that represents a String

This chapter sketches the definition of a DltArgument that represents a string, in which case another reference to an SwBaseType in the role DltArgument.networkRepresentation.swTextProps is required.

```
<DLT-ARGUMENT>
  <SHORT-NAME>ProcessName</SHORT-NAME>
  <DESC>
    <L-2 L="EN">Process name</L-2>
  </DESC>
  <LENGTH>32</LENGTH>
  <NETWORK-REPRESENTATION>
    <SW-DATA-DEF-PROPS-VARIANTS>
      <SW-DATA-DEF-PROPS-CONDITIONAL>
        <BASE-TYPE-REF DEST="SW-BASE-TYPE">/AUTOSAR_Platform/BaseTypes/
           uint8</BASE-TYPE-REF>
        <SW-TEXT-PROPS>
          <BASE-TYPE-REF DEST="SW-BASE-TYPE">/BaseTypes/utf8_codepoint</
             BASE-TYPE-REF>
        </SW-TEXT-PROPS>
        <UNIT-REF DEST="UNIT">/AISpecification/Units/NoUnit</UNIT-REF>
      </SW-DATA-DEF-PROPS-CONDITIONAL>
    </SW-DATA-DEF-PROPS-VARIANTS>
  </NETWORK-REPRESENTATION>
```

```
</DLT-ARGUMENT>
```

Listing A.8: Example of a DltArgument that represents a string

Please note that the SwBaseType referenced in the role DltArgument.networkRepresentation.swTextProps intentionally does not define attribute baseTypeSize because its job is only to clarify the encoding of the string, which in this specific case is set to “UTF-8”.

```
<SW-BASE-TYPE>
  <SHORT-NAME>utf8_codepoint</SHORT-NAME>
  <CATEGORY>FIXED_LENGTH</CATEGORY>
  <BASE-TYPE-ENCODING>UTF-8</BASE-TYPE-ENCODING>
</SW-BASE-TYPE>
```

Listing A.9: Definition of the encoding of the string

A.5 Example of a DLT Argument that represents an Enumeration

This chapter sketches the definition of a DltArgument that represents an enumeration. The definition of the DltArgument can be found in Listing A.10.

```
<DLT-ARGUMENT>
  <SHORT-NAME>State</SHORT-NAME>
  <DESC>
    <L-2 L="EN">Process State Change</L-2>
  </DESC>
  <NETWORK-REPRESENTATION>
    <SW-DATA-DEF-PROPS-VARIANTS>
      <SW-DATA-DEF-PROPS-CONDITIONAL>
        <BASE-TYPE-REF DEST="SW-BASE-TYPE">/AUTOSAR_Platform/BaseTypes/
            uint32</BASE-TYPE-REF>
        <COMPU-METHOD-REF DEST="COMPU-METHOD">/CompuMethods/ProcessState</
            COMPU-METHOD-REF>
        <UNIT-REF DEST="UNIT">/AISpecification/Units/NoUnit</UNIT-REF>
      </SW-DATA-DEF-PROPS-CONDITIONAL>
    </SW-DATA-DEF-PROPS-VARIANTS>
  </NETWORK-REPRESENTATION>
</DLT-ARGUMENT>
```

Listing A.10: Example of a DltArgument that represents an enumeration

The definition of an enumeration on this level requires a reference to a CompuMethod of category, which is sketched in Listing A.11.

```
  <SHORT-NAME>ProcessState</SHORT-NAME>
  <LONG-NAME>
    <L-4 L="EN">Process state enumeration</L-4>
  </LONG-NAME>
  <CATEGORY>TEXTTABLE</CATEGORY>
  <UNIT-REF DEST="UNIT">/AISpecification/Units/NoUnit</UNIT-REF>
  <COMPU-INTERNAL-TO-PHYS>
    <COMPU-SCALES>
      <COMPU-SCALE>
```

```
        <DESC>
          <L-2 L="EN">Idle Process state</L-2>
        </DESC>
        <LOWER-LIMIT INTERVAL-TYPE="CLOSED">0</LOWER-LIMIT>
        <UPPER-LIMIT INTERVAL-TYPE="CLOSED">0</UPPER-LIMIT>
        <COMPU-CONST>
          <VT>Idle</VT>
        </COMPU-CONST>
      </COMPU-SCALE>
      <COMPU-SCALE>
        <DESC>
          <L-2 L="EN">Starting Process state</L-2>
        </DESC>
        <LOWER-LIMIT INTERVAL-TYPE="CLOSED">1</LOWER-LIMIT>
        <UPPER-LIMIT INTERVAL-TYPE="CLOSED">1</UPPER-LIMIT>
        <COMPU-CONST>
          <VT>Starting</VT>
        </COMPU-CONST>
      </COMPU-SCALE>
      <COMPU-SCALE>
        <DESC>
          <L-2 L="EN">Running Process state</L-2>
        </DESC>
        <LOWER-LIMIT INTERVAL-TYPE="CLOSED">2</LOWER-LIMIT>
        <UPPER-LIMIT INTERVAL-TYPE="CLOSED">2</UPPER-LIMIT>
        <COMPU-CONST>
          <VT>Running</VT>
        </COMPU-CONST>
      </COMPU-SCALE>
      <COMPU-SCALE>
        <DESC>
          <L-2 L="EN">Terminating Process state</L-2>
        </DESC>
        <LOWER-LIMIT INTERVAL-TYPE="CLOSED">3</LOWER-LIMIT>
        <UPPER-LIMIT INTERVAL-TYPE="CLOSED">3</UPPER-LIMIT>
        <COMPU-CONST>
          <VT>Terminating</VT>
        </COMPU-CONST>
      </COMPU-SCALE>
      <COMPU-SCALE>
        <DESC>
          <L-2 L="EN">Terminated Process state</L-2>
        </DESC>
        <LOWER-LIMIT INTERVAL-TYPE="CLOSED">4</LOWER-LIMIT>
        <UPPER-LIMIT INTERVAL-TYPE="CLOSED">4</UPPER-LIMIT>
        <COMPU-CONST>
          <VT>Terminated</VT>
        </COMPU-CONST>
      </COMPU-SCALE>
    </COMPU-SCALES>
  </COMPU-INTERNAL-TO-PHYS>
</COMPU-METHOD>
```

Listing A.11: Definition of a CompuMethod for an enumeration

B Imposition Times of Constraints

The constraints formulated in this document have different *actual* imposition times which denote the steps in the workflow when the respective constraint has to be imposed.

Some imposition times “include” other imposition times, an example for this relation is discussed in the Table B.1

The imposition times that are considered applicable in the scope of this document[1] are listed in Table B.1.

Please note that the imposition times are intentionally rendered as technical terms such that it is possible to link back from each constraint to the definition of the affected imposition time in Table B.1.

Imposition Time	Description	Motivation
IT_LogTrace	Log and Trace Extract is complete	This imposition time denotes the step in the workflow, where the Log and Trace Extract is about to be finished.

Table B.1: Imposition Times of constraints in this document

[1] Different imposition times may be defined in the context of other AUTOSAR standard documents

C History of Constraints and Specification Items

Please note that the lists in this chapter also include constraints and specification items that have been removed from the specification in a later version. These constraints and specification items do not appear as hyperlinks in the document.

C.1 Constraint and Specification Item History of this document according to AUTOSAR Release R21-11

C.1.1 Added Specification Items in R21-11

Number	Heading
[TPS_DLTXT_00001]	Log or trace message representation
[TPS_DLTXT_00002]	DLT message
[TPS_DLTXT_00003]	User data of the log or trace message
[TPS_DLTXT_00004]	`DltArgument` kinds
[TPS_DLTXT_00005]	predefined text `DltArgument`
[TPS_DLTXT_00006]	assembled data `DltArgument`
[TPS_DLTXT_00007]	Description of a String as `networkRepresentation` of a `DltArgument`
[TPS_DLTXT_00008]	Standardized values of `DltMessage.messageTypeInfo`
[TPS_DLTXT_00009]	Semantics of `DltArgument.length`
[TPS_DLTXT_00010]	Semantics of `DltArgument.length` in case of a String
[TPS_DLTXT_00011]	Description of `DltArgument` with one-dimensional Array type
[TPS_DLTXT_00012]	Description of `DltArgument` with n-dimensional Array type
[TPS_DLTXT_00013]	Description of `DltArgument` with Structure type
[TPS_DLTXT_00014]	Semantics of `DltArgument.optional`

Table C.1: Added Specification Items in R21-11

C.1.2 Changed Specification Items in R21-11

none

C.1.3 Deleted Specification Items in R21-11

none

C.1.4 Added Constraints in R21-11

Number	Heading
[constr_5098]	Usage of DltArgument.networkRepresentation
[constr_5294]	Existence of DltEcu.ecuId
[constr_5295]	Existence of DltApplication.context
[constr_5296]	Existence of DltApplication.applicationId
[constr_5297]	Existence of DltApplication.applicationDescription
[constr_5298]	Existence of DltContext.contextId
[constr_5299]	Existence of DltContext.contextDescription
[constr_5300]	Existence of DltContext.dltMessage
[constr_5301]	Existence of DltMessage.messageId
[constr_5302]	Restriction in usage of DltArgument.optional attribute
[constr_5303]	Restriction of baseTypeSize of a DltArgument
[constr_5304]	Datatype of an Array
[constr_5305]	CompuMethod in DltArgument.networkRepresentation

Table C.2: Added Constraints in R21-11

C.1.5 Changed Constraints in R21-11

none

C.1.6 Deleted Constraints in R21-11

none

C.2 Constraint and Specification Item History of this document according to AUTOSAR Release R22-11

C.2.1 Added Specification Items in R22-11

none

C.2.2 Changed Specification Items in R22-11

none

C.2.3 Deleted Specification Items in R22-11

none

C.2.4 Added Constraints in R22-11

Number	Heading
[constr_5340]	Range of `DltMessage.privacyLevel.privacyLevel`
[constr_5341]	Range of `PrivacyLevel.compuMethod`

Table C.3: Added Constraints in R22-11

C.2.5 Changed Constraints in R22-11

Number	Heading
[constr_5098]	Allowed `SwDataDefProps` attributes for `DltArgument.networkRepresentation`
[constr_5302]	Restriction in usage of `DltArgument.optional` attribute

Table C.4: Changed Constraints in R22-11

C.2.6 Deleted Constraints in R22-11

none

C.3 Constraint and Specification Item History of this document according to AUTOSAR Release R23-11

C.3.1 Added Specification Items in R23-11

none

C.3.2 Changed Specification Items in R23-11

none

C.3.3 Deleted Specification Items in R23-11

none

C.3.4 Added Constraints in R23-11

Number	Heading
[constr_5363]	Allowed usage of attributes for description of payload data types
[constr_5364]	Allowed usage of attributes in case of a `dltArgumentEntry`

Table C.5: Added Constraints in R23-11

C.3.5 Changed Constraints in R23-11

none

C.3.6 Deleted Constraints in R23-11

none

C.4 Constraint and Specification Item History of this document according to AUTOSAR Release R24-11

C.4.1 Added Specification Items in R24-11

none

C.4.2 Changed Specification Items in R24-11

Number	Heading
[TPS_DLTXT_00007]	Description of a String as `networkRepresentation` of a `DltArgument`
[TPS_DLTXT_00010]	Semantics of `DltArgument.length` in case of a String

Table C.6: Changed Specification Items in R24-11

C.4.3 Deleted Specification Items in R24-11

none

C.4.4 Added Constraints in R24-11

none

C.4.5 Changed Constraints in R24-11

Number	Heading
[constr_5098]	Allowed SwDataDefProps attributes for DltArgument. networkRepresentation

Table C.7: Changed Constraints in R24-11

C.4.6 Deleted Constraints in R24-11

none

D Mentioned Class Tables

For the sake of completeness, this chapter contains a set of class tables representing meta-classes mentioned in the context of this document but which are not contained directly in the scope of describing specific meta-model semantics.

Class	**AUTOSAR**			
Package	M2::AUTOSARTemplates::AutosarTopLevelStructure			
Note	Root element of an AUTOSAR description, also the root element in corresponding XML documents. **Tags:** xml.globalElement=true			
Base	*ARObject*			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
adminData	AdminData	0..1	aggr	This represents the administrative data of an Autosar file. **Stereotypes:** atpSplitable **Tags:** atp.Splitkey=adminData xml.sequenceOffset=10
arPackage	ARPackage	*	aggr	This is the top level package in an AUTOSAR model. **Stereotypes:** atpSplitable; atpVariation **Tags:** atp.Splitkey=arPackage.shortName, arPackage.variationPoint.shortLabel vh.latestBindingTime=blueprintDerivationTime xml.sequenceOffset=30
fileInfoComment	FileInfoComment	0..1	aggr	This represents a possibility to provide a structured comment in an AUTOSAR file. **Stereotypes:** atpStructuredComment **Tags:** xml.roleElement=true xml.sequenceOffset=-10 xml.typeElement=false
introduction	DocumentationBlock	0..1	aggr	This represents an introduction on the Autosar file. It is intended for example to represent disclaimers and legal notes. **Tags:** xml.sequenceOffset=20

Table D.1: AUTOSAR

Class	**BaseTypeDirectDefinition**			
Package	M2::MSR::AsamHdo::BaseTypes			
Note	This BaseType is defined directly (as opposite to a derived BaseType)			
Base	*ARObject*, *BaseTypeDefinition*			
Aggregated by	*BaseType*.baseTypeDefinition			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
baseTypeEncoding	BaseTypeEncodingString	0..1	attr	This specifies, how an object of the current BaseType is encoded, e.g. in an ECU within a message sequence. **Tags:** xml.sequenceOffset=90
baseTypeSize	PositiveInteger	0..1	attr	Describes the length of the data type specified in the container in bits. **Tags:** xml.sequenceOffset=70

▽

△

Class	**BaseTypeDirectDefinition**			
byteOrder	ByteOrderEnum	0..1	attr	This attribute specifies the byte order of the base type. **Tags:** xml.sequenceOffset=110
memAlignment	PositiveInteger	0..1	attr	This attribute describes the alignment of the memory object in bits. E.g. "8" specifies, that the object in question is aligned to a byte while "32" specifies that it is aligned four byte. If the value is set to "0" the meaning shall be interpreted as "unspecified". **Tags:** xml.sequenceOffset=100
native Declaration	NativeDeclarationString	0..1	attr	This attribute describes the declaration of such a base type in the native programming language, primarily in the Programming language C. This can then be used by a code generator to include the necessary declarations into a header file. For example BaseType with shortName: "MyUnsignedInt" native Declaration: "unsigned short" Results in typedef unsigned short MyUnsignedInt; If the attribute is not defined the referring Implementation DataTypes will not be generated as a typedef by RTE. If a nativeDeclaration type is given it shall fulfill the characteristic given by basetypeEncoding and baseType Size. This is required to ensure the consistent handling and interpretation by software components, RTE, COM and MCM systems. **Tags:** xml.sequenceOffset=120

Table D.2: BaseTypeDirectDefinition

Class	**CompuMethod**			
Package	M2::MSR::AsamHdo::ComputationMethod			
Note	This meta-class represents the ability to express the relationship between a physical value and the mathematical representation. Note that this is still independent of the technical implementation in data types. It only specifies the formula how the internal value corresponds to its physical pendant. **Tags:** atp.recommendedPackage=CompuMethods			
Base	*ARElement*, *ARObject*, *AtpBlueprint*, *AtpBlueprintable*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *PackageableElement*, *Referrable*			
Aggregated by	ARPackage.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
compuInternalToPhys	Compu	0..1	aggr	This specifies the computation from internal values to physical values. **Stereotypes:** atpSplitable **Tags:** atp.Splitkey=compuInternalToPhys xml.sequenceOffset=80

▽

△

Class	**CompuMethod**			
compuPhysTo Internal	Compu	0..1	aggr	This represents the computation from physical values to the internal values. **Stereotypes:** atpSplitable **Tags:** atp.Splitkey=compuPhysToInternal xml.sequenceOffset=90
displayFormat	DisplayFormatString	0..1	attr	This property specifies, how the physical value shall be displayed e.g. in documents or measurement and calibration tools. **Tags:** xml.sequenceOffset=20
unit	Unit	0..1	ref	This is the physical unit of the Physical values for which the CompuMethod applies. **Tags:** xml.sequenceOffset=30

Table D.3: CompuMethod

Class	***Identifiable*** (abstract)
Package	M2::AUTOSARTemplates::GenericStructure::GeneralTemplateClasses::Identifiable
Note	Instances of this class can be referred to by their identifier (within the namespace borders). In addition to this, Identifiables are objects which contribute significantly to the overall structure of an AUTOSAR description. In particular, Identifiables might contain Identifiables.
Base	*ARObject*, *MultilanguageReferrable*, *Referrable*
Subclasses	ARPackage, *AbstractDoIpLogicAddressProps*, *AbstractEvent*, *AbstractImplementationDataTypeElement*, *AbstractSecurityEventFilter*, *AbstractSecurityIdsmInstanceFilter*, *AbstractServiceInstance*, Application Endpoint, ApplicationError, AppliedStandard, ArtifactChecksum, *AtpBlueprint*, *AtpBlueprintable*, *Atp Classifier*, *AtpFeature*, AutosarOperationArgumentInstance, AutosarVariableInstance, BlockState, *Build ActionEntity*, BuildActionEnvironment, Chapter, ClassContentConditional, ClientIdDefinition, ClientServer Operation, Code, *CollectableElement*, ComManagementMapping, *CommConnectorPort*, *Communication Connector*, *CommunicationController*, Compiler, ConsistencyNeeds, ConsumedEventGroup, Coupling Port, *CouplingPortAbstractShaper*, *CouplingPortStructuralElement*, CryptoKeySlot, *CryptoService Mapping*, DataPrototypeGroup, DataPrototypeTransformationPropsIdent, DataTransformation, DdsCp Domain, DdsCpPartition, DdsCpQosProfile, DdsCpTopic, DependencyOnArtifact, *DiagEventDebounce Algorithm*, DiagnosticAuthTransmitCertificateEvaluation, DiagnosticConnectedIndicator, DiagnosticData Element, DiagnosticDebounceAlgorithmProps, DiagnosticFunctionInhibitSource, DiagnosticParameter Element, *DiagnosticRoutineSubfunction*, DltApplication, DltArgument, DltMessage, DoIpInterface, DoIp LogicAddress, DoIpRoutingActivation, EndToEndProtection, EthernetWakeupSleepOnDatalineConfig, EventHandler, ExclusiveArea, *ExecutableEntity*, *ExecutionTime*, FMAttributeDef, FMFeatureMap Assertion, FMFeatureMapCondition, FMFeatureMapElement, FMFeatureRelation, FMFeatureRestriction, FMFeatureSelection, FlexrayArTpNode, FlexrayTpPduPool, *FrameTriggering*, GeneralParameter, Global TimeGateway, *GlobalTimeMaster*, *GlobalTimeSlave*, *HeapUsage*, HwAttributeDef, HwAttributeLiteral Def, HwPin, HwPinGroup, *IEEE1722TpAcfBus*, *IEEE1722TpAcfBusPart*, IPSecRule, IPv6ExtHeader FilterList, ISignalToIPduMapping, ISignalTriggering, *IdentCaption*, ImpositionTime, InternalTriggering Point, Keyword, LifeCycleState, Linker, MacAddressVlanMembership, MacMulticastGroup, MacSecKay Participant, McDataInstance, MemorySection, ModeDeclaration, ModeDeclarationMapping, ModeSwitch Point, NetworkEndpoint, *NmCluster*, *NmNode*, *PackageableElement*, ParameterAccess, PduActivation RoutingGroup, PduToFrameMapping, PduTriggering, PerInstanceMemory, *PhysicalChannel*, PortGroup, *PortInterfaceMapping*, ResourceConsumption, RootSwCompositionPrototype, RptComponent, Rpt Container, RptExecutableEntity, RptExecutableEntityEvent, RptExecutionContext, RptProfile, RptService Point, RunnableEntityGroup, *SdgAttribute*, SdgClass, SecOcJobRequirement, SecureCommunication AuthenticationProps, SecureCommunicationFreshnessProps, SecurityEventContextDataElement, SecurityEventContextProps, *ServiceNeeds*, SignalServiceTranslationEventProps, SignalService TranslationProps, SocketAddress, SomeipTpChannel, *SpecElementReference*, *StackUsage*, Static SocketConnection, StructuredReq, SwGenericAxisParamType, SwServiceArg, SwcServiceDependency, SwitchAsynchronousTrafficShaperGroupEntry, SystemMapping, *TimeBaseResource*, *TimingClock*, TimingClockSyncAccuracy, TimingCondition, *TimingConstraint*, *TimingDescription*, TimingExtension Resource, TimingModeInstance, Topic1, TpAddress, TraceableTable, TraceableText, *TracedFailure*, TransformationISignalPropsIdent, *TransformationProps*, TransformationTechnology, Trigger, Variable Access, VariationPointProxy, ViewMap, VlanConfig, WaitPoint

▽

△

Class	***Identifiable*** (abstract)			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
adminData	AdminData	0..1	aggr	This represents the administrative data for the identifiable object. **Stereotypes:** atpSplitable **Tags:** atp.Splitkey=adminData xml.sequenceOffset=-40
annotation	Annotation	*	aggr	Possibility to provide additional notes while defining a model element (e.g. the ECU Configuration Parameter Values). These are not intended as documentation but are mere design notes. **Tags:** xml.sequenceOffset=-25
category	CategoryString	0..1	attr	The category is a keyword that specializes the semantics of the Identifiable. It affects the expected existence of attributes and the applicability of constraints. **Tags:** xml.sequenceOffset=-50
desc	MultiLanguageOverview Paragraph	0..1	aggr	This represents a general but brief (one paragraph) description what the object in question is about. It is only one paragraph! Desc is intended to be collected into overview tables. This property helps a human reader to identify the object in question. More elaborate documentation, (in particular how the object is built or used) should go to "introduction". **Tags:** xml.sequenceOffset=-60
introduction	DocumentationBlock	0..1	aggr	This represents more information about how the object in question is built or is used. Therefore it is a DocumentationBlock. **Tags:** xml.sequenceOffset=-30
uuid	String	0..1	attr	The purpose of this attribute is to provide a globally unique identifier for an instance of a meta-class. The values of this attribute should be globally unique strings prefixed by the type of identifier. For example, to include a DCE UUID as defined by The Open Group, the UUID would be preceded by "DCE:". The values of this attribute may be used to support merging of different AUTOSAR models. The form of the UUID (Universally Unique Identifier) is taken from a standard defined by the Open Group (was Open Software Foundation). This standard is widely used, including by Microsoft for COM (GUIDs) and by many companies for DCE, which is based on CORBA. The method for generating these 128-bit IDs is published in the standard and the effectiveness and uniqueness of the IDs is not in practice disputed. If the id namespace is omitted, DCE is assumed. An example is "DCE:2fac1234-31f8-11b4-a222-08002b34c003". The uuid attribute has no semantic meaning for an AUTOSAR model and there is no requirement for AUTOSAR tools to manage the timestamp. **Tags:** xml.attribute=true

Table D.4: Identifiable

Class	***PortPrototype*** (abstract)			
Package	M2::AUTOSARTemplates::SWComponentTemplate::Components			
Note	Base class for the ports of an AUTOSAR software component. The aggregation of PortPrototypes is subject to variability with the purpose to support the conditional existence of ports.			
Base	*ARObject*, *AtpBlueprintable*, *AtpFeature*, *AtpPrototype*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Subclasses	*AbstractProvidedPortPrototype*, *AbstractRequiredPortPrototype*			
Aggregated by	*AtpClassifier*.atpFeature, *SwComponentType*.port			
Attribute	**Type**	**Mult.**	**Kind**	**Note**
–	–	–	–	–

Table D.5: PortPrototype

Class	***Referrable*** (abstract)			
Package	M2::AUTOSARTemplates::GenericStructure::GeneralTemplateClasses::Identifiable			
Note	Instances of this class can be referred to by their identifier (while adhering to namespace borders).			
Base	*ARObject*			
Subclasses	*AtpDefinition*, BswDistinguishedPartition, *BswModuleCallPoint*, BswModuleClientServerEntry, BswVariableAccess, CouplingPortTrafficClassAssignment, *DiagnosticEnvModeElement*, EthernetPriorityRegeneration, ExclusiveAreaNestingOrder, *HwDescriptionEntity*, *ImplementationProps*, ModeTransition, *MultilanguageReferrable*, PncMappingIdent, *SingleLanguageReferrable*, SoConIPduIdentifier, SocketConnectionBundle, TimeSyncServerConfiguration, TpConnectionIdent			
Attribute	**Type**	**Mult.**	**Kind**	**Note**
shortName	Identifier	1	attr	This specifies an identifying shortName for the object. It needs to be unique within its context and is intended for humans but even more for technical reference. **Stereotypes:** atpIdentityContributor **Tags:** xml.enforceMinMultiplicity=true xml.sequenceOffset=-100
shortName Fragment	ShortNameFragment	*	aggr	This specifies how the Referrable.shortName is composed of several shortNameFragments. **Tags:** xml.sequenceOffset=-90

Table D.6: Referrable

Class	**SwBaseType**			
Package	M2::MSR::AsamHdo::BaseTypes			
Note	This meta-class represents a base type used within ECU software. **Tags:** atp.recommendedPackage=BaseTypes			
Base	*ARElement*, *ARObject*, *AtpBlueprint*, *AtpBlueprintable*, *BaseType*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *PackageableElement*, *Referrable*			
Aggregated by	ARPackage.element			
Attribute	**Type**	**Mult.**	**Kind**	**Note**
–	–	–	–	–

Table D.7: SwBaseType

Class	«atpVariation» **SwDataDefProps**			
Package	M2::MSR::DataDictionary::DataDefProperties			
Note	This class is a collection of properties relevant for data objects under various aspects. One could consider this class as a "pattern of inheritance by aggregation". The properties can be applied to all objects of all classes in which SwDataDefProps is aggregated. **Tags:** vh.latestBindingTime=codeGenerationTime			
Base	*ARObject*			
Aggregated by	*AutosarDataType*.swDataDefProps, CompositeNetworkRepresentation.networkRepresentation, CppImplementationDataTypeElement.swDataDefProps, *DataPrototype*.swDataDefProps, DataPrototypeTransformationProps.networkRepresentationProps, DiagnosticDataElement.swDataDefProps, DiagnosticEnvDataElementCondition.swDataDefProps, DltArgument.networkRepresentation, FlatInstanceDescriptor.swDataDefProps, ImplementationDataTypeElement.swDataDefProps, InstantiationDataDefProps.swDataDefProps, ISignal.networkRepresentationProps, McDataInstance.resultingProperties, ParameterAccess.swDataDefProps, PerInstanceMemory.swDataDefProps, *ReceiverComSpec*.networkRepresentation, SecurityEventContextDataElement.networkRepresentation, *SenderComSpec*.networkRepresentation, SomeipDataPrototypeTransformationProps.networkRepresentation, SwPointerTargetProps.swDataDefProps, SwServiceArg.swDataDefProps, SwSystemconst.swDataDefProps, SystemSignal.physicalProps			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
annotation	Annotation	*	aggr	This aggregation allows to add annotations (yellow pads ...) related to the current data object. **Tags:** xml.roleElement=true xml.roleWrapperElement=true xml.sequenceOffset=20 xml.typeElement=false xml.typeWrapperElement=false
baseType	SwBaseType	0..1	ref	Base type associated with the containing data object. **Tags:** xml.sequenceOffset=50
compuMethod	CompuMethod	0..1	ref	Computation method associated with the semantics of this data object. **Tags:** xml.sequenceOffset=180
dataConstr	DataConstr	0..1	ref	Data constraint for this data object. **Tags:** xml.sequenceOffset=190
displayFormat	DisplayFormatString	0..1	attr	This property describes how a number is to be rendered e.g. in documents or in a measurement and calibration system. **Tags:** xml.sequenceOffset=210
display Presentation	DisplayPresentation Enum	0..1	attr	This attribute controls the presentation of the related data for measurement and calibration tools.
invalidValue	ValueSpecification	0..1	aggr	Optional value to express invalidity of the actual data element. **Tags:** xml.sequenceOffset=255
swComparison Variable	SwVariableRefProxy	*	aggr	Variables used for comparison in an MCD process. **Tags:** xml.sequenceOffset=170 xml.typeElement=false
swHostVariable	SwVariableRefProxy	0..1	aggr	Contains a reference to a variable which serves as a host-variable for a bit variable. Only applicable to bit objects. **Tags:** xml.sequenceOffset=220 xml.typeElement=false
swTextProps	SwTextProps	0..1	aggr	the specific properties if the data object is a text object. **Tags:** xml.sequenceOffset=120

△

Class	«atpVariation» **SwDataDefProps**			
unit	Unit	0..1	ref	Physical unit associated with the semantics of this data object. This attribute applies if no compuMethod is specified. If both units (this as well as via compuMethod) are specified the units shall be compatible. **Tags:** xml.sequenceOffset=350

Table D.8: SwDataDefProps

Class	**SwTextProps**			
Package	M2::MSR::DataDictionary::DataDefProperties			
Note	This meta-class expresses particular properties applicable to strings in variables or calibration parameters.			
Base	*ARObject*			
Aggregated by	SwDataDefProps.swTextProps			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
arraySize Semantics	ArraySizeSemantics Enum	0..1	attr	This attribute controls the semantics of the arraysize for the array representing the string in an Implementation DataType. It is there to support a safe conversion between ApplicationDatatype and ImplementationDatatype, even for variable length strings as required e.g. for Support of SAE J1939.
baseType	SwBaseType	0..1	ref	This is the base type of one character in the string. In particular this baseType denotes the intended encoding of the characters in the string on level of ApplicationData Type. **Tags:** xml.sequenceOffset=30
swFillCharacter	Integer	0..1	attr	Filler character for text parameter to pad up to the maximum length swMaxTextSize. The value will be interpreted according to the encoding specified in the associated base type of the data object, e.g. 0x30 (hex) represents the ASCII character zero as filler character and 0 (dec) represents an end of string as filler character. The usage of the fill character depends on the arraySize Semantics. **Tags:** xml.sequenceOffset=40
swMaxTextSize	Integer	0..1	attr	Specifies the maximum text size in characters. Note the size in bytes depends on the encoding in the corresponding baseType. **Stereotypes:** atpVariation **Tags:** vh.latestBindingTime=preCompileTime xml.sequenceOffset=20

Table D.9: SwTextProps

Class	**Unit**			
Package	M2::MSR::AsamHdo::Units			
Note	This is a physical measurement unit. All units that might be defined should stem from SI units. In order to convert one unit into another factor and offset are defined. For the calculation from SI-unit to the defined unit the factor (factorSiToUnit) and the offset (offsetSiTo Unit) are applied as follows: x [{unit}] := y * [{siUnit}] * factorSiToUnit [[unit]/{siUnit}] + offsetSiToUnit [{unit}] For the calculation from a unit to SI-unit the reciprocal of the factor (factorSiToUnit) and the negation of the offset (offsetSiToUnit) are applied. y {siUnit} := (x*{unit} - offsetSiToUnit [{unit}]) / (factorSiToUnit [[unit]/{siUnit}] **Tags:** atp.recommendedPackage=Units			
Base	*ARElement*, *ARObject*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *Packageable Element*, *Referrable*			
Aggregated by	ARPackage.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
displayName	SingleLanguageUnit Names	0..1	aggr	This specifies how the unit shall be displayed in documents or in user interfaces of tools.The displayName corresponds to the Unit.Display in an ASAM MCD-2MC file. **Tags:** xml.sequenceOffset=20
factorSiToUnit	Float	0..1	attr	This is the factor for the conversion from SI Units to units. The inverse is used for conversion from units to SI Units. **Tags:** xml.sequenceOffset=30
offsetSiToUnit	Float	0..1	attr	This is the offset for the conversion from and to siUnits. **Tags:** xml.sequenceOffset=40
physical Dimension	PhysicalDimension	0..1	ref	This association represents the physical dimension to which the unit belongs to. Note that only values with units of the same physical dimensions might be converted. **Tags:** xml.sequenceOffset=50

Table D.10: Unit