Document Title	TPS_DiagnosticExtractTemplate: 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 TPS_DiagnosticExtractTemplate

1.1 Specification Item constr_1325

Trace References:

none

Content:

The allowed attributes of SwDataDefProps for the aggregation in the role DiagnosticDataElement.DiagnosticDataElement.swDataDefProps are defined in table [REF table_3a_SwDataDefPropsForDiagnosticDataElement]. 71860 7186078069

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #74446: Explanation for the applicability of VARIABLE_LENGTH missing

  Problem description:

  The SWCT contains constr_1011 and constr_1013 that mention the value VARIABLE_LENGTH as a possible setting for SwBaseType.category.

  There is no clear description when this value would be applicable. In fact, doubts are warranted whether the value (and with it, the attribute maxBaseTypeSize) should be applicable in AUTOSAR at all.

  Background: the definition of a SwBaseType is supposed to refer to an atomic data type. The "base type" for an array is still the "base type" of the array elements, as opposed to the definition of a base type for the array itself.

  The difference between the definition of a "base type" for the array elements and a "base type" for the entire array manifests in the attributes baseTypeSize, resp. maxBaseTypeSize.

  In some cases definitions of SwBaseType can be found in the wild that look similar to this specimen:

  ```
  <SW-BASE-TYPE>
  <SHORT-NAME>FunnyBaseType</SHORT-NAME>
  <CATEGORY>VARIABLE_LENGTH</CATEGORY>
  <MAX-BASE-TYPE-SIZE>1433</MAX-BASE-TYPE-SIZE>
  <BASE-TYPE-ENCODING>WINDOWS-1252</BASE-TYPE-ENCODING>
  </SW-BASE-TYPE>
  ```

I guess that this base type has been build on the assumption that a string deserves a base type for the entire Windows-1252-encoded string.

This assumption would not only result in countless definitions of SwBaseTypes for each possible combination of array length and data type, it would also create an unholy dependency to the value of ImplementationDataTypeElement.arraySize. In other words, if the value of array size changes than the value of maxBaseTypeSize would have to change accordingly.

This does not make much sense.

I tried to get some information about the use case for VARIABLE_LENGTH that was originally intended. Bernhard mentioned that it was meant to support edge cases for the generation of A2L files, in particular for string implementations where the first element denotes the length and the rest is reserved for the payload.

If there are further known use cases, please feel free to leave a comment ...

I would like to provide a better explanation to the applicability of VARIABLE_LENGTH and also better explain that the SwBaseType is not meant to be applied to composite data types.

At the moment the document is rather indifferent about these two aspects.

Agreed solution:

SWCT:
—–

1. Reformulate:

[constr_1011] category of SwBaseType
d The attribute SwBaseType.category must be set and only the values FIXED_LENGTH and VOID are supported. c

[constr_1422] Value of category is VOID
d If the value of the attribute SwBaseType.category is set to VOID then the attribute baseTypeSize shall not exist. c()

[constr_1012] Value of category is FIXED_LENGTH
d If the value of the attribute SwBaseType.category is set to FIXED_LENGTH then the attribute baseTypeSize shall be filled with content. c()

[constr_1229] category of ImplementationDataType boils down to VALUE subElement.category is set to VALUE or TYPE_REFERENCE that eventually boils down to VALUE and the subElement refers to a SwBaseType where baseTypeSize is set to the value 8 and the baseTypeEncoding is set to NONE.

[constr_1220] Compatibility of SwBaseType
d Two SwBaseTypes are compatible if and only if attributes baseTypeSize respectively byteOrder, memAlignment, baseTypeEncoding, and nativeDeclaration have identical values. c()

2. Remove [constr_1013].

3. Set attribute maxBaseTypeSize to ’obsolete’.

4. Adapt Listing 5.4: Example for the definition of a string SwBaseType: (add CATEGORY)

```
<AR-PACKAGE>
<ELEMENTS>
<SW-BASE-TYPE>
<SHORT-NAME>MyTextBaseType</SHORT-NAME>
<CATEGORY>FIXED_LENGTH</CATEGORY>
<BASE-TYPE-SIZE>8</BASE-TYPE-SIZE>
<BASE-TYPE-ENCODING>UTF-8</BASE-TYPE-ENCODING>
</SW-BASE-TYPE>
<SW-BASE-TYPE>
<SHORT-NAME>uint8BaseType</SHORT-NAME>
<CATEGORY>FIXED_LENGTH</CATEGORY>
<BASE-TYPE-SIZE>8</BASE-TYPE-SIZE>
</SW-BASE-TYPE>
</ELEMENTS>
</AR-PACKAGE>
```

5. Fix all object diagrams showing SwBaseType (add CATEGORY).

DEXT:
——

1. Remove in Table 4.8: Allowed attributes of SwDataDefProps for DiagnosticDataElement.swDataDefProps the line baseType.baseTypeDefinition.maxBaseTypeSize N/A.
–Last change on issue 74446 comment 7–

BW-C-Level:

Application	Specification	Bus
3	3	1

1.2 Specification Item constr_1356

Trace References:

none

Content:

The value of DiagnosticExtendedDataRecord.recordNumber shall be unique among all DiagnosticExtendedDataRecords in the context of the enclosing DiagnosticContributionSet.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76630: constr_1356 and constr_1358 are too narrow

  Problem description:

  constr_1356 and constr_1358 limit the numbers of EDRs/SRs to be ECU unique. But actually they shall be unique within context of a certain fault memory (primary, user defined).

  Agreed solution:

  constr_xxxx recordNumber shall be unique within primary fault memory
  For all DiagnosticTroubleCodeProps that refer to DiagnosticMemoryDestinationPrimary in the role memoryDestination there shall be no two freezeFrame.recordNumber or extendedDataRecord.recordNumber with the same value.

  constr_xxx1 recordNumber shall be unique within mirror fault memory
  For all DiagnosticTroubleCodeProps that refer to DiagnosticMemoryDestinationMirror in the role memoryDestination there shall be no two freezeFrame.recordNumber or extendedDataRecord.recordNumber with the same value.

  constr_xxx2 recordNumber shall be unique within user-defined fault memory
  For all DiagnosticTroubleCodeProps that refer to DiagnosticMemoryDestinationUserDefined in the role memoryDestination there shall be no two freezeFrame.recordNumber or extendedDataRecord.recordNumber with the same value

for any DiagnosticMemoryDestinationUserDefined referenced as DiagnosticTroubleCodeProps.memoryDestination with a given value of memoryId.
–Last change on issue 76630 comment 4–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.3 Specification Item constr_1358

Trace References:

none

Content:

The value of DiagnosticFreezeFrame.recordNumber shall be unique among all DiagnosticFreezeFrames in the context of the enclosing DiagnosticContributionSet.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76630: constr_1356 and constr_1358 are too narrow

  Problem description:

  constr_1356 and constr_1358 limit the numbers of EDRs/SRs to be ECU unique. But actually they shall be unique within context of a certain fault memory (primary, user defined).

  Agreed solution:

  constr_xxxx recordNumber shall be unique within primary fault memory
  For all DiagnosticTroubleCodeProps that refer to DiagnosticMemoryDestinationPrimary in the role memoryDestination there shall be no two freezeFrame.recordNumber or extendedDataRecord.recordNumber with the same value.

  constr_xxx1 recordNumber shall be unique within mirror fault memory
  For all DiagnosticTroubleCodeProps that refer to DiagnosticMemoryDestinationMirror in the role memoryDestination there shall be no two freezeFrame.recordNumber or extendedDataRecord.recordNumber with the same value.

  constr_xxx2 recordNumber shall be unique within user-defined fault memory
  For all DiagnosticTroubleCodeProps that refer to DiagnosticMemoryDestina-

tionUserDefined in the role memoryDestination there shall be no two freezeFrame.recordNumber or extendedDataRecord.recordNumber with the same value for any DiagnosticMemoryDestinationUserDefined referenced as DiagnosticTroubleCodeProps.memoryDestination with a given value of memoryId.
–Last change on issue 76630 comment 4–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.4 Specification Item constr_1450

Trace References:

none

Content:

if a DiagnosticServiceSwMapping or DiagnosticServiceDataMapping refers to a DiagnosticRequestCurrentPowertrainData and a DiagnosticDataElement that is aggregated by a DiagnosticParameterIdentifier then the then one of two alternative model configurations shall exist:

- SwcServiceDependency referenced by the same DiagnosticServiceSwMapping resp. DiagnosticServiceDataMapping shall aggregate a an ObdPidServiceNeeds in the role SwcServiceDependency.serviceNeeds.
- The BswServiceDependencyIdent referenced by the same DiagnosticServiceSw Mapping shall aggregate an ObdPidServiceNeeds in the role BswServiceDependency.serviceNeeds.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #73489: Service Use Cases for for OBD Mode1 /9 to BSW

  Problem description:

  During the concept review (see bug # 73032) of the DEXT (extension 1) the following issue was identified:

  "In constr_o002 and constr_o28 reference to SwcServiceDependency is given. But OBD Mode 1 and 9 also offers a FNC interface. Is then with this also a BswServiceDependency to mention?"

  The consideration of the BswServiceDependency for the service mapping requires a proper description of the applicable service use case in the BSWMDT.

Therefore, the following activities are required:

1. Define service use cases in the BSWMDT for OBD Mode 1 and 9. These involve the usage of BwcServiceDependencies that aggregate ObdPidServiceNeeds/ObdPÍnfoServiceNeeds.

2. Extend the constraints regarding the service mapping for mode 1 and 9 towards the support for BswServiceDependency.

Agreed solution:

DEXT

====

Replace

if a DiagnosticServiceSwMapping or DiagnosticServiceDataMapping refers to a DiagnosticRequestCurrentPowertrainData and a
DiagnosticDataElement that is aggregated by a DiagnosticParameterIdentifier then the SwcServiceDependency referenced by the same DiagnosticServiceSwMapping resp. DiagnosticServiceDataMapping shall aggregate a ObdPidServiceNeeds in the role serviceNeeds.

by

If a DiagnosticServiceSwMapping or DiagnosticServiceDataMapping refers to a DiagnosticRequestCurrentPowertrainData and a
DiagnosticDataElement that is aggregated by a DiagnosticParameterIdentifier then one of two alternative model configurations shall exist:
* The SwcServiceDependency referenced by the same DiagnosticServiceSwMapping resp. DiagnosticServiceDataMapping shall aggregate an ObdPidServiceNeeds in the role serviceNeeds.
* The BswServiceDependencyIdent referenced by the same DiagnosticServiceSwMapping shall aggregate an ObdPidServiceNeeds in the role serviceNeeds.

Replace

If a DiagnosticServiceSwMapping refers to DiagnosticRequestVehicleInfo and a DiagnosticDataElement that is aggregated by a DiagnosticInfoType then the SwcServiceDependency referenced by the same DiagnosticServiceSwMapping shall aggregate a ObdInfoServiceNeeds in the role serviceNeeds.

by

If a DiagnosticServiceSwMapping refers to DiagnosticRequestVehicleInfo and a DiagnosticDataElement that is aggregated by a DiagnosticInfoType then one of two alternative model configurations shall exist:
* The SwcServiceDependency referenced by the same DiagnosticServiceSwMapping shall aggregate a ObdInfoServiceNeeds in the role serviceNeeds.
* The BswServiceDependencyIdent referenced by the same DiagnosticServiceSwMapping shall aggregate an ObdInfoServiceNeeds in the role serviceNeeds.

BSWMDT
======

Add new subchapter 13.2.2.4 in the document:

Scenario: a Basic Software Module offers a BswModuleEntry to read value via OBD services.
[TPS_BSWMDT_0xxxx] Basic Software Module offers a BswModuleEntry to read value via OBD services
ServiceNeeds kind ObdPidServiceNeeds
RoleBasedBswModuleEntryAssignment valid roles:
Xxx_ReadData [1] (1 in case read is supported)
RoleBasedDataAssignment
N/A
RoleBasedDataTypeAssignment
N/A

Scenario: a Basic Software Module offers a BswModuleEntry to read vehicle information via OBD services.
[TPS_BSWMDT_0xxx1] Basic Software Module offers a BswModuleEntry to read vehicle information via OBD services
ServiceNeeds kind ObdInfoServiceNeeds
RoleBasedBswModuleEntryAssignment valid roles:
Xxx_GetInfoTypeValueData [1] (1 in case read is supported)
RoleBasedDataAssignment
N/A
RoleBasedDataTypeAssignment
N/A
–Last change on issue 73489 comment 6–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.5 Specification Item constr_1451

Trace References:

none

Content:

if a DiagnosticServiceSwMapping refers to DiagnosticRequestVehicleInfo and a DiagnosticDataElement that is aggregated by a DiagnosticInfoType then the one of two alternative model configurations shall exist:

- The SwcServiceDependency referenced by the same DiagnosticServiceSwMapping shall aggregate a ObdInfoServiceNeeds in the role SwcServiceDependency.serviceNeeds.
- The BswServiceDependencyIdent referenced by the same DiagnosticServiceSwMapping shall aggregate an ObdInfoServiceNeeds in the role BswServiceDependency.serviceNeeds.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #73489: Service Use Cases for for OBD Mode1 /9 to BSW

  Problem description:

  During the concept review (see bug # 73032) of the DEXT (extension 1) the following issue was identified:

  "In constr_o002 and constr_o28 reference to SwcServiceDependency is given. But OBD Mode 1 and 9 also offers a FNC interface. Is then with this also a BswServiceDependency to mention?"

  The consideration of the BswServiceDependency for the service mapping requires a proper description of the applicable service use case in the BSWMDT.

  Therefore, the following activities are required:

  1. Define service use cases in the BSWMDT for OBD Mode 1 and 9. These involve the usage of BwcServiceDependencies that aggregate ObdPidService-

Needs/ObdPÍnfoServiceNeeds.

2. Extend the constraints regarding the service mapping for mode 1 and 9 towards the support for BswServiceDependency.

Agreed solution:

DEXT
====

Replace

if a DiagnosticServiceSwMapping or DiagnosticServiceDataMapping refers to a DiagnosticRequestCurrentPowertrainData and a
DiagnosticDataElement that is aggregated by a DiagnosticParameterIdentifier then the SwcServiceDependency referenced by the same DiagnosticServiceSwMapping resp. DiagnosticServiceDataMapping shall aggregate a ObdPidServiceNeeds in the role serviceNeeds.

by

If a DiagnosticServiceSwMapping or DiagnosticServiceDataMapping refers to a DiagnosticRequestCurrentPowertrainData and a
DiagnosticDataElement that is aggregated by a DiagnosticParameterIdentifier then one of two alternative model configurations shall exist:
* The SwcServiceDependency referenced by the same DiagnosticServiceSwMapping resp. DiagnosticServiceDataMapping shall aggregate an ObdPidServiceNeeds in the role serviceNeeds.
* The BswServiceDependencyIdent referenced by the same DiagnosticServiceSwMapping shall aggregate an ObdPidServiceNeeds in the role serviceNeeds.

Replace

If a DiagnosticServiceSwMapping refers to DiagnosticRequestVehicleInfo and a DiagnosticDataElement that is aggregated by a DiagnosticInfoType then the SwcServiceDependency referenced by the same DiagnosticServiceSwMapping shall aggregate a ObdInfoServiceNeeds in the role serviceNeeds.

by

If a DiagnosticServiceSwMapping refers to DiagnosticRequestVehicleInfo and a DiagnosticDataElement that is aggregated by a DiagnosticInfoType then one of two alternative model configurations shall exist:

* The SwcServiceDependency referenced by the same DiagnosticServiceSwMapping shall aggregate a ObdInfoServiceNeeds in the role serviceNeeds.
* The BswServiceDependencyIdent referenced by the same DiagnosticServiceSwMapping shall aggregate an ObdInfoServiceNeeds in the role serviceNeeds.

BSWMDT
======

Add new subchapter 13.2.2.4 in the document:

Scenario: a Basic Software Module offers a BswModuleEntry to read value via OBD services.
[TPS_BSWMDT_0xxxx] Basic Software Module offers a BswModuleEntry to read value via OBD services
ServiceNeeds kind ObdPidServiceNeeds
RoleBasedBswModuleEntryAssignment valid roles:
Xxx_ReadData [1] (1 in case read is supported)
RoleBasedDataAssignment
N/A
RoleBasedDataTypeAssignment
N/A

Scenario: a Basic Software Module offers a BswModuleEntry to read vehicle information via OBD services.
[TPS_BSWMDT_0xxx1] Basic Software Module offers a BswModuleEntry to read vehicle information via OBD services
ServiceNeeds kind ObdInfoServiceNeeds
RoleBasedBswModuleEntryAssignment valid roles:
Xxx_GetInfoTypeValueData [1] (1 in case read is supported)
RoleBasedDataAssignment
N/A
RoleBasedDataTypeAssignment
N/A
–Last change on issue 73489 comment 6–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.6 Specification Item constr_1509

Trace References:

none

Content:

For all DiagnosticTroubleCodeProps that refer to DiagnosticMemoryDestinationPrimary in the role DiagnosticTroubleCodeProps.memoryDestination there shall be no two DiagnosticTroubleCodeProps.extendedDataRecord.DiagnosticExtendedDataRecord.record Number with the same value.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76630: constr_1356 and constr_1358 are too narrow

  Problem description:

  constr_1356 and constr_1358 limit the numbers of EDRs/SRs to be ECU unique. But actually they shall be unique within context of a certain fault memory (primary, user defined).

  Agreed solution:

  constr_xxxx recordNumber shall be unique within primary fault memory
  For all DiagnosticTroubleCodeProps that refer to DiagnosticMemoryDestinationPrimary in the role memoryDestination there shall be no two freezeFrame.recordNumber or extendedDataRecord.recordNumber with the same value.

  constr_xxx1 recordNumber shall be unique within mirror fault memory
  For all DiagnosticTroubleCodeProps that refer to DiagnosticMemoryDestinationMirror in the role memoryDestination there shall be no two freezeFrame.recordNumber or extendedDataRecord.recordNumber with the same value.

  constr_xxx2 recordNumber shall be unique within user-defined fault memory
  For all DiagnosticTroubleCodeProps that refer to DiagnosticMemoryDestinationUserDefined in the role memoryDestination there shall be no two freezeFrame.recordNumber or extendedDataRecord.recordNumber with the same value for any DiagnosticMemoryDestinationUserDefined referenced as DiagnosticTroubleCodeProps.memoryDestination with a given value of memoryId.
  –Last change on issue 76630 comment 4–

  BW-C-Level:

Application	Specification	Bus
1	1	1

1.7 Specification Item constr_1510

Trace References:

none

Content:

For all DiagnosticTroubleCodeProps that refer to DiagnosticMemoryDestinationMirror in the role DiagnosticTroubleCodeProps.memoryDestination there shall be no two DiagnosticTroubleCodeProps.extendedDataRecord.DiagnosticExtendedDataRecord.record Number with the same value.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76630: constr_1356 and constr_1358 are too narrow

  Problem description:

  constr_1356 and constr_1358 limit the numbers of EDRs/SRs to be ECU unique. But actually they shall be unique within context of a certain fault memory (primary, user defined).

  Agreed solution:

  constr_xxxx recordNumber shall be unique within primary fault memory
  For all DiagnosticTroubleCodeProps that refer to DiagnosticMemoryDestinationPrimary in the role memoryDestination there shall be no two freezeFrame.recordNumber or extendedDataRecord.recordNumber with the same value.

  constr_xxx1 recordNumber shall be unique within mirror fault memory
  For all DiagnosticTroubleCodeProps that refer to DiagnosticMemoryDestinationMirror in the role memoryDestination there shall be no two freezeFrame.recordNumber or extendedDataRecord.recordNumber with the same value.

  constr_xxx2 recordNumber shall be unique within user-defined fault memory
  For all DiagnosticTroubleCodeProps that refer to DiagnosticMemoryDestinationUserDefined in the role memoryDestination there shall be no two freezeFrame.recordNumber or extendedDataRecord.recordNumber with the same value for any DiagnosticMemoryDestinationUserDefined referenced as DiagnosticTrou-

bleCodeProps.memoryDestination with a given value of memoryId.
–Last change on issue 76630 comment 4–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.8 Specification Item constr_1511

Trace References:

none

Content:

For all DiagnosticTroubleCodeProps that refer to DiagnosticMemoryDestinationUserDefined in the role DiagnosticTroubleCodeProps.memoryDestination there shall be no two extendedDataRecord.DiagnosticExtendedDataRecord.recordNumber with the same value for any DiagnosticMemoryDestinationUserDefined referenced as DiagnosticTrouble CodeProps.DiagnosticTroubleCodeProps.memoryDestination with a given value of DiagnosticMemoryDestinationUserDefined.memoryId.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76630: constr_1356 and constr_1358 are too narrow

  Problem description:

  constr_1356 and constr_1358 limit the numbers of EDRs/SRs to be ECU unique. But actually they shall be unique within context of a certain fault memory (primary, user defined).

  Agreed solution:

  constr_xxxx recordNumber shall be unique within primary fault memory
  For all DiagnosticTroubleCodeProps that refer to DiagnosticMemoryDestinationPrimary in the role memoryDestination there shall be no two freezeFrame.recordNumber or extendedDataRecord.recordNumber with the same value.

  constr_xxx1 recordNumber shall be unique within mirror fault memory
  For all DiagnosticTroubleCodeProps that refer to DiagnosticMemoryDestinationMirror in the role memoryDestination there shall be no two freezeFrame.recordNumber or extendedDataRecord.recordNumber with the same value.

constr_xxx2 recordNumber shall be unique within user-defined fault memory
For all DiagnosticTroubleCodeProps that refer to DiagnosticMemoryDestinationUserDefined in the role memoryDestination there shall be no two freezeFrame.recordNumber or extendedDataRecord.recordNumber with the same value for any DiagnosticMemoryDestinationUserDefined referenced as DiagnosticTroubleCodeProps.memoryDestination with a given value of memoryId.
–Last change on issue 76630 comment 4–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.9 Specification Item constr_1512

Trace References:

none

Content:

For all DiagnosticTroubleCodeProps that refer to DiagnosticMemoryDestinationPrimary in the role DiagnosticTroubleCodeProps.memoryDestination there shall be no two DiagnosticTroubleCodeProps.freezeFrame.DiagnosticFreezeFrame.recordNumber with the same value.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76630: constr_1356 and constr_1358 are too narrow

  Problem description:

  constr_1356 and constr_1358 limit the numbers of EDRs/SRs to be ECU unique. But actually they shall be unique within context of a certain fault memory (primary, user defined).

  Agreed solution:

  constr_xxxx recordNumber shall be unique within primary fault memory
  For all DiagnosticTroubleCodeProps that refer to DiagnosticMemoryDestinationPrimary in the role memoryDestination there shall be no two freezeFrame.recordNumber or extendedDataRecord.recordNumber with the same value.

constr_xxx1 recordNumber shall be unique within mirror fault memory
For all DiagnosticTroubleCodeProps that refer to DiagnosticMemoryDestinationMirror in the role memoryDestination there shall be no two freezeFrame.recordNumber or extendedDataRecord.recordNumber with the same value.

constr_xxx2 recordNumber shall be unique within user-defined fault memory
For all DiagnosticTroubleCodeProps that refer to DiagnosticMemoryDestinationUserDefined in the role memoryDestination there shall be no two freezeFrame.recordNumber or extendedDataRecord.recordNumber with the same value for any DiagnosticMemoryDestinationUserDefined referenced as DiagnosticTroubleCodeProps.memoryDestination with a given value of memoryId.
–Last change on issue 76630 comment 4–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.10 Specification Item constr_1513

Trace References:

none

Content:

For all DiagnosticTroubleCodeProps that refer to DiagnosticMemoryDestinationMirror in the role DiagnosticTroubleCodeProps.memoryDestination there shall be no two DiagnosticTroubleCodeProps.freezeFrame.DiagnosticFreezeFrame.recordNumber with the same value.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76630: constr_1356 and constr_1358 are too narrow

  Problem description:

  constr_1356 and constr_1358 limit the numbers of EDRs/SRs to be ECU unique. But actually they shall be unique within context of a certain fault memory (primary, user defined).

  Agreed solution:

  constr_xxxx recordNumber shall be unique within primary fault memory
  For all DiagnosticTroubleCodeProps that refer to DiagnosticMemoryDestinationPrimary in the role memoryDestination there shall be no two freeze-

Frame.recordNumber or extendedDataRecord.recordNumber with the same value.

constr_xxx1 recordNumber shall be unique within mirror fault memory
For all DiagnosticTroubleCodeProps that refer to DiagnosticMemoryDestinationMirror in the role memoryDestination there shall be no two freezeFrame.recordNumber or extendedDataRecord.recordNumber with the same value.

constr_xxx2 recordNumber shall be unique within user-defined fault memory
For all DiagnosticTroubleCodeProps that refer to DiagnosticMemoryDestinationUserDefined in the role memoryDestination there shall be no two freezeFrame.recordNumber or extendedDataRecord.recordNumber with the same value for any DiagnosticMemoryDestinationUserDefined referenced as DiagnosticTroubleCodeProps.memoryDestination with a given value of memoryId.
–Last change on issue 76630 comment 4–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.11 Specification Item constr_1514

Trace References:

none

Content:

For all DiagnosticTroubleCodeProps that refer to DiagnosticMemoryDestinationUserDefined in the role DiagnosticTroubleCodeProps.memoryDestination there shall be no two DiagnosticTroubleCodeProps.freezeFrame.DiagnosticFreezeFrame.recordNumber with the same value for any DiagnosticMemoryDestinationUserDefined referenced as Diagnostic TroubleCodeProps.DiagnosticTroubleCodeProps.memoryDestination with a given value of DiagnosticMemoryDestinationUserDefined.memoryId.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76630: constr_1356 and constr_1358 are too narrow

  Problem description:

constr_1356 and constr_1358 limit the numbers of EDRs/SRs to be ECU unique. But actually they shall be unique within context of a certain fault memory (primary, user defined).

Agreed solution:

constr_xxxx recordNumber shall be unique within primary fault memory
For all DiagnosticTroubleCodeProps that refer to DiagnosticMemoryDestinationPrimary in the role memoryDestination there shall be no two freezeFrame.recordNumber or extendedDataRecord.recordNumber with the same value.

constr_xxx1 recordNumber shall be unique within mirror fault memory
For all DiagnosticTroubleCodeProps that refer to DiagnosticMemoryDestinationMirror in the role memoryDestination there shall be no two freezeFrame.recordNumber or extendedDataRecord.recordNumber with the same value.

constr_xxx2 recordNumber shall be unique within user-defined fault memory
For all DiagnosticTroubleCodeProps that refer to DiagnosticMemoryDestinationUserDefined in the role memoryDestination there shall be no two freezeFrame.recordNumber or extendedDataRecord.recordNumber with the same value for any DiagnosticMemoryDestinationUserDefined referenced as DiagnosticTroubleCodeProps.memoryDestination with a given value of memoryId.
–Last change on issue 76630 comment 4–

BW-C-Level:

Application	Specification	Bus
1	1	1

1.12 Specification Item constr_1515

Trace References:

none

Content:

The reference from DiagnosticRoutineControl (via its abstract base class DiagnosticServiceInstance) in the role DiagnosticServiceInstance.accessPermissionto meta-class DiagnosticAccessPermission shall not be used.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76406: Routine authorisation shall be per sub-function

  Problem description:

  With ASR 4.3.0 the DEXT models routine authorization by referencing a DiagnosticAccessPermisssion from DiagnosticRoutine.
  But the Dcm uses individual authorizations by sub-function (see RfC# 65079).

  Agreed solution:

  Introduce a new meta-class DiagnosticRoutineSubfunction (description: "this meta-class acts as an abstract base class to routine subfunctions") that inherits from Identifiable, and that acts as a base class for DiagnosticStartRoutine, DiagnosticStopRoutine, and DiagnosticRequestRoutineResults. DiagnosticRoutineSubfunction refers (description: "this reference represents the access permission of the owning routine subfunction") to DiagnosticAccessPermission in the role accessPermission. The multiplicity of the reference is 0..1.

  Add a constraint: "A reference from DiagnosticRoutineControl to DiagnosticAccessPermission is not valid."
  –Last change on issue 76406 comment 15–

  BW-C-Level:

Application	Specification	Bus
1	3	1

1.13 Specification Item TPS_DEXT_01073

Trace References:

RS_DEXT_00048

Content:

Diagnostic properties that are specific to an individual EcuInstance are modeled by means of the meta-class DiagnosticEcuProps that is aggregated at EcuInstance in the role EcuInstance.diagnosticProps.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #77364: [DEXT] clarification of existence DiagnosticEcuProps / DiagnosticEcuInstanceProps

  Problem description:

Both the classes contains similar information:

DiagnosticEcuProps
isObdRelevant -> no upstream mapping exists
sendRespPendOnTransToBoot -> upstream mapping exits

DiagnosticEcuInstanceProps
obdSupport -> upstream mapping exits
sendRespPendOnTransToBoot -> no upstream mapping exists

Are both classes required? Is it allowed to have both classes configured? How is then the handling / upstream mapping?

Agreed solution:

hint:
Please implement RfC 77365 at first or together.

Set DiagnosticEcuProps.obdSupport to obsolete.

Set the entire meta-class DiagnosticEcuProps to obsolete as after this RfC the meta-class has no more used attributes. But the emptiness of the container depends on the outcome of RfC# 77365.
Remove TPS_DEXT_01073
Remove "Figure 4.8: ECU-specific diagnostic properties"
Remove "Table 4.27: DiagnosticEcuProps"
–Last change on issue 77364 comment 12–

BW-C-Level:

Application	**Specification**	**Bus**
1	3	1

1.14 Specification Item TPS_DEXT_01087

Trace References:

RS_DEXT_00054

Content:

Different types of DiagnosticOperationCycles are supported and defined by the Diagnostic OperationCycle.type attribute, e.g. time between powering up and powering down the ECU or the time between ignition on and ignition off.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76710: Clarify DiagnosticOperationCycleTypeEnum

  Problem description:

  Description of DiagnosticOperationCycle.DiagnosticOperationCycleTypeEnum mentiones: "Operation cycles types for the Dem to be supported by cycle-state APIs."

  1) Dem shall not be mentioned in DEXT
  2) Actually no Dem API uses the operation cycle types

  Agreed solution:

  Set the attributes power, time to obsolete.

  Change the Description of DiagnosticOperationCycle.DiagnosticOperationCycleTypeEnum to : "Operation cycles types used to identify certain Operation cycles with a certain semantics."

  Change the [TPS_DEXT_01087] to
  Semantics of DiagnosticOperationCycle d Different types of DiagnosticOperationCycles are supported and defined by the type attribute,
  e.g. time between ignition on and ignition off. c(RS_DEXT_00054)

  Change in the table Class DiagnosticOperationCycle in the Note of Type : Operation cycles types for the Dem.
  –Last change on issue 76710 comment 10–

  BW-C-Level:

Application	Specification	Bus
1	4	1

1.15 Specification Item TPS_DEXT_01140

Trace References:

none

Content:

The values specified in a given DiagnosticExtract shall be taken for the derivation of basic software modules in the diagnostic stack.

This means that parameter values in the Ecuc are created that could be subject to constraints, e.g. a valid interval.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #75002: [Dext] Add Constraints for Range checks

  Problem description:

  There are no range checks defined for Attributes like
  BitOffset / MAX-NUMBER-OF-ELEMENTS.
  This may result in ECUC Paramater range validation errors after the UpStreamMapping (DcmDspRoutineSignalPos / DcmDspRoutineSignalLength/ DcmDspDidDataPos / DcmDspDataSize).

  Therefore already on DEXT side there shall be the according range check introduced.

  Agreed solution:

  Add to ch. 3.2:

  [TPS_DEXT_0xxxx] Values contained in Diagnostic Extract shall be taken for the derivation of basic software modules
  The values specified in a given Diagnostic extract shall be taken for the derivation of basic software modules in the diagnostic stack. This means that parameter values in the Ecuc are created that could be subject to constraints, e.g. a valid interval.

  Therefore, model elements in the Diagnostic Extract shall be carefully checked against the constraints formulated in EcuC. However, this document does not contain specific constraints on a detailed basis.
  –Last change on issue 75002 comment 12–

  BW-C-Level:

Application	Specification	Bus
1	1	1