EAST-ADLv2.1.12 XSD schema

The purpose of this document is to give the pattern rules for the EAST-ADL (v2.1.12) XSD schema generation in comparation with the existing XSD schema (the AUTOSAR TR XML persistence rules). Each subsection proposes the pattern rule for the corresponding case listed in the table EAST-ADL-PersistencePattern-2.1.12.xlsx. The proposed pattern rules are based on RMF solutions. The subsections marked as x.x.x propose the XSD schema that have not been defined in the current RMF XSD specification document. The others give the current RMF solution and its XML example.

The global proposed EAST-ADL XSD schema structure is similar as RMF ReqIF (more details in <http://www.omg.org/spec/ReqIF/1.0.1/>):

1. EA Schema ::=

<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"

xmlns:EA="http://east-adl.info/2.1.12"

targetNamespace="http://east-adl.info/2.1.12"

elementFormDefault="qualified"

attributeFormDefault="unqualified">

2:PackageSchema

</xsd:schema>

For Package (“eastadl21”):

2. PackageSchema ::=

3:GlobalElement

( 5:ClassTypeDef )\*

( 6:EnumSchema )\*

( 7:DatatypeSchema )\*

( 8:ReferencedSimpleType )\*

For the global element (“EAXML”):

3. GlobalElement ::=

<xsd:element name="// Name of Global Element //" type="EA : // Name of Global Element Type //" >

</xsd:element>"

Note: there should be exactly one element (“EAXML” in EAST-ADL) with tag xml.global\_element set to true.

For Class:

5. ClassTypeDef ::=

<xsd:complexType name="//Name of Class//">

<xsd:sequence>

( 5a:ClassElementAttribute )\*

( 5c:ClassCompositions) \*

( 5i:EReference\_contained1100Many)\*

( 5j:EReference\_contained0101Single)\*

( 5l: EReference\_referenced1100Many)\*

( 5m:EReference\_referenced0100Many)\*

( 5n:EReference\_referenced0100Single)\*

</xsd:sequence>

( 5d:ClassAttribute )\*

</xsd:complexType>

Case attribute0100Single :

5a. ClassElementAttribute ::=

<xsd:element name=" //Name of Attribute// " minOccurs=" // Minimum // " maxOccurs="//Maximum//"

( fixed=" // fixed value // " )?

type=" //Name of Attribute Type// "/>

Case contained1001Many :

5c. ClassCompositions ::=

<xsd:element name='" // Name of Target Property (plural) // "' minOccurs="( "'0'" | "'1'" )" maxOccurs='1'>

<xsd:complexType>

<xsd:choice minOccurs='"//Minimum of Target Property//"' maxOccurs='"// Max of Target Property// "'>

("<xsd:element

name='" // Name of Target (Sub) Class // "'

type='" //namespace// ":" // Name of Target Class // "' /> )+

"</xsd:choice>

</xsd:complexType>

</xsd:element>

Case attribute (kind=attribute):

5d. ClassAttribute ::=

<xsd:attribute name="// XML name of Attribute //" type="//Type of Attribute // " use= "prohibited" | "optional" | "required" />

For Enumeration:

6. EnumSchema ::=

<xsd:simpleType name=" // Name of Enumeration Class // ">

<xsd:restriction base="xsd:string">"

(<xsd:enumeration value=" // (upper case) Name of Literal from Enumeration Class // '/>)\*

</xsd:restriction>

</xsd:simpleType>"

For Data Type:

7. DatatypeSchema ::= (7a | 7b)

1. *If isGeneralPrimitiveType() == false:*

*isGeneralPrimitiveType() returns true only if: general case: the mapped xsd type of the instance class is in the xsd primitive type set*

7a ::= <xsd:complexType name=" // Name of Datatype // " >

<xsd:sequence>

<xsd:any namespace=" // Namespace URI of Datatype // "

processContents=" (skip | lax | strict)"

minOccurs=" // Minimum // "

maxOccurs=" // Maximum // "/>

</xsd:any>

</xsd:sequence>

</xsd:complexType>

1. *If isGeneralPrimitiveType() == true && isXMLPrimitiveXsdType()== true:*

*isXMLPrimitiveXsdType() returns true only if:*

*== general case: isGeneralPrimitiveType() = true, no pattern, xmlCustomSimpleType= false*

*== EAST-ADL: isGeneralPrimitiveType() = true, sterotype=primitive, no pattern, xmlCustomSimpleType= false*

Do not create any of type, the reference to this type use the corresponding xsd type that maps to the instance class instead directly

1. *If isGeneralPrimitiveType() == true && isXMLPrimitiveXsdType()== false*

7b ::= <xsd:simpleType name="//xmlName//">

<xsd:restriction base="xsd://xmlXsdSimpleType//">

<xsd:pattern value="//xmlXsdPattern//"/>

</xsd:restriction>

</xsd:simpleType>

For Class that is referenced:

8. ReferencedSimpleType ::=

<xsd:simpleType name=”//xmlName//--SUBTYPES-ENUM”>

<xsd:restriction base=”xsd:string”>

(<xsd:enumeration value=”//(sub) typeXmlName//”/>)+

</xsd:restriction>

</xsd:simpleType>

The following sections gives the XSD schema and examples of the properties (attributes and references). The proposed declaration will be added into rule 5.ClassTypeDef that describes the declaration of the class.

# 1.Attribute

## 1.1 row 2 : (kind=attribute)

**Proposed RMF persistence pattern: 5d. ClassAttribute**

* + 1. target pattern

existing target pattern

**TR\_APRXML\_00019 ::=**

<xsd:attribute name="//XML name of attribute//" type="// typeXmlNsPrefix : typeXmlName//" use= "required" | "optional" />

Note: The value of “use” is derived from the multiplicity of the attribute from the metamodel. If the lower multiplicity is greater than 0, then “required” is used, else “use” is set to “optional”.

For each class, if at least one property is marked by the tagged value “xml.attribute = true”, then a xsd:attributeGroup is created:

**TR\_APRXML\_00002 ::=**

<xsd:attributeGroup name="//XML name of class//"

(TR\_APRXML\_00019)+

</xsd:attributeGroup>

proposed target pattern

**5d. ClassAttribute** ::=

<xsd:attribute name="// XML name of Attribute //" type="//Type of Attribute // " use= "prohibited" | "optional" | "required" />

Note: The value of “use” is derived from the multiplicity of the attribute from the metamodel. If the upper value is 0, then “prohibited” is used, else if the lower value is 0, then “use” is set to “optional”, else if the lower value is greater than 0, “required” is used.

* + 1. XML schema example

Attribute: Idenfitiable.uuid

existing XML schema example

<xsd:attributeGroup name="*IDENTIFIABLE*">

<xsd:attribute name="*UUID*" type="*xsd:string*">

</xsd:attribute>

…

</xsd:attributeGroup>

proposed XML schema example

<xsd:complexType name="*IDENTIFIABLE*">

<xsd:attribute name="*UUID*" type="*xsd:string*" use="*optional*"/>

…

</xsd:complexType>

* + 1. XML instance example

existing XML instance example

<EA-PACKAGE UUID="*EARoot-uuid*">

…

</EA-PACKAGE>

proposed XML instance example

<EA-PACKAGE UUID="*EARoot-uuid*">

…

</EA-PACKAGE>

## 1.2 row 3 : Eattribute0100Single

**RMF persistence pattern : 5a. ClassElementAttribute**

1.2.1 target pattern

Existing target pattern

**TR\_APRXML\_00013** ::=

*if (types.length > 1) :*

<xsd:element name="//roleXmlName//" minOccurs="//lowerMultiplicity//" maxOccurs="//upperMultiplicity//">

<xsd:complexType>

<xsd:choice minOccurs="//lowerMultiplicity//" maxOccurs="1">

(<xsd:group ref="//typeXmlNsPrefix : typeXmlName//"/>)+

</xsd:choice>

</xsd:complexType>

</xsd:element>

*if (NOT type.length >1):*

<xsd:element name="//roleXmlName//" type="//typeXmlNsPrefix : typeXmlName//"

minOccurs="//lowerMultiplicity//" maxOccurs="//upperMultiplicity//"/>

This declaration will be added into the xsd:group of the class:

**TR\_APRXML\_00001** ::=

<xsd:group name="//XML name of class//">

<xsd:sequence>

(TR\_APRXML\_00013)\*

(TR\_APRXML\_00023)\*

(TR\_APRXML\_00010)\*

(TR\_APRXML\_00012)\*

…

</xsd:sequence>

</xsd:group>

proposed target pattern

**5a. ClassElementAttribute** ::=

<xsd:element

name=" //Name of Attribute (single) // " minOccurs=" // Minimum // " maxOccurs=" // Maximum // "

( fixed=" // fixed value // " )?

type=" //Name of Attribute Type// "/>

1.2.2 XML schema example

Attribute: FunctionBehavior.path

existing XML schema example

<xsd:group name="*FUNCTON-BEHAVIOR*" >

<xsd:sequence>

<xsd:element maxOccurs="*1*" minOccurs="*1*" name="*PATH*" type="*xsd:string*"/>

…

</xsd:sequence>

</xsd:group>

proposed XML schema example

<xsd:complexType name=" *FUNCTON-BEHAVIOR*">

<xsd:sequence>

<xsd:element maxOccurs="*1*" minOccurs="*1*" name="*PATH* " type="*xsd:string*"/>

…

</xsd: sequence >

</xsd:complexType>

1.2.3 XML instance example

existing XML instance example

<FUNCTION-BEHAVIOR>

…

<PATH>*path1*</PATH>

</FUNCTION-BEHAVIOR>

proposed XML instance example

<FUNCTION-BEHAVIOR>

…

<PATH>*path1*</PATH>

</FUNCTION-BEHAVIOR>

# 2. Reference

## 2.1 row 6 : EReference\_contained1100Many (proposed to be replaced by EReference\_contained1001Many to avoid xsi:type, same as 2.2)

2.1.1 target pattern

existing target pattern

**TR\_APRXML\_00023** ::=

*if (types.length > 1) :*

<xsd:element name="//roleXmlNamePlural//" minOccurs="// if (lowerMultiplicity>0) then 1 else 0//" maxOccurs="1">

<xsd:complexType>

<xsd:choice minOccurs="//lowerMultiplicity//" maxOccurs="//upperMultiplicity//">

<xsd:element name="//roleXmlName//">

<xsd:complexType>

<xsd:choice minOccurs="//if (lowerMultiplicity>0) then 1 else 0//" maxOccurs="1">

(<xsd:group ref="//typeXmlNsPrefix :typeXmlName//"/>)+

</xsd:choice>

</xsd:complexType>

</xsd:element>

</xsd:choice>

</xsd:complexType>

</xsd:element>

*if (NOT type.length >1):*

<xsd:element name="//roleXmlNamePlural//" minOccurs="//if (lowerMultiplicity>0) then 1 else 0//" maxOccurs="1">

<xsd:complexType>

<xsd:choice minOccurs="//lowerMultiplicity//" maxOccurs="//upperMultiplicity//">

<xsd:element name="//roleXmlName//" type="//typeXmlNsPrefix : typeXmlName//"/>

</xsd:choice>

</xsd:complexType>

</xsd:element>

This declaration will be added into the xsd:group of the class (TR\_APRXML\_00001).

Proposed target pattern

5i:EReference\_contained1100Many ::=

<xsd:element name=" // XML Name of Target Property (plural) // " minOccurs="( "0" | "1" )" maxOccurs="1">

<xsd:complexType>

<xsd:choice minOccurs="//Minimum of Target Property//" maxOccurs="// Max of Target Property// ">

(<xsd:element

name=" // feature XML Name // "

type=" //namespace// ":" // XML Name of Target Class // " /> )+

</xsd:choice>

</xsd:complexType>

</xsd:element>

2.1.2 XML schema example

Reference: Refine.refinedBy

existing XML schema example

<xsd:group name="*REFINE*" >

<xsd:sequence>

<xsd:element maxOccurs="*1*" minOccurs="*1*" name="*REFINED-BY-IREFS*">

<xsd:complexType>

<xsd:choice maxOccurs="*unbounded*" minOccurs="*1*">

<xsd:element name="*REFINED-BY-IREF*" type="*EA:REFINE--REFINED-BY-IREF*"/>

</xsd:choice>

</xsd:complexType>

</xsd:element>

</xsd:sequence>

</xsd:group>

proposed XML schema example

<xsd:complexType name="*REFINE*" >

<xsd: sequence >

<xsd:element maxOccurs=*"1"* minOccurs=*"1"* name=*" REFINED-BY-IREFS "*>

<xsd:complexType>

<xsd:choice maxOccurs=*"unbounded"* minOccurs=*"1"*>

<xsd:element name*=" REFINED-BY-IREF"* type=*"EA: REFINE--REFINED-BY-IREF"*/>

</xsd:choice>

</xsd:complexType>

</xsd:element>

</xsd: sequence >

</xsd:complexType>

2.1.3 XML instance example

existing XML instance example

<REFINED-BY-IREFS>

<REFINED-BY-IREF>

<IDENTIFIABLE-TARGET-REF DEST="FUNCTION-BEHAVIOR">*/EARoot/functionBehavior2*</IDENTIFIABLE-TARGET-REF>

…

</REFINED-BY-IREF>

<REFINED-BY-IREF>

<IDENTIFIABLE-TARGET-REF DEST="BEHAVIOR">*/EARoot/behavior1*</IDENTIFIABLE-TARGET-REF>

…

</REFINED-BY-IREF>

</REFINED-BY-IREFS>

proposed XML instance example

<REFINED-BY-IREFS >

<REFINED-BY-IREF>

<IDENTIFIABLE-TARGET-REF TYPE="FUNCTION BEHAVIOR">*/EARoot/functionBehavior2*</IDENTIFIABLE-TARGET-REF>

….

</REFINED-BY-IREF >

<REFINED-BY-IREF>

<IDENTIFIABLE-TARGET-REF TYPE="BEHAVIOR">*/EARoot/behavior1*</IDENTIFIABLE-TARGET-REF>

…

</REFINED-BY-IREF>

</REFINED-BY-IREFS >

## 2.2 row 7 : EReference\_contained1001Many

**RMF persistence pattern : 5c. ClassCompositions**

2.2.1 Target pattern:

existing target pattern

**TR\_APRXML\_00010** ::=

<xsd:element name="//roleXmlNamePlural//" minOccurs="//if (lowerMultiplicity>0) then 1 else 0//" maxOccurs="1">

<xsd:complexType>

<xsd:choice minOccurs="//lowerMultiplicity//" maxOccurs="//upperMultiplicity//">

(<xsd:element name="//typeXmlName//" type="//typeXmlNsPrefix : typeXmlName//"/>)+

</xsd:choice>

</xsd:complexType>

</xsd:element>

This declaration will be added into the xsd:group of the class (TR\_APRBXML\_00001).

proposed target pattern

**5c. ClassCompositions** ::=

<xsd:element name=" // XML Name of Target Property (plural) // " minOccurs="( "0" | "1" )" maxOccurs="1">

<xsd:complexType>

<xsd:choice minOccurs="//Minimum of Target Property//" maxOccurs="// Max of Target Property// ">

(<xsd:element

name=" // XML Name of Target (Sub) Class // "

type=" //namespace// ":" // XML Name of Target Class // " /> )+

</xsd:choice>

</xsd:complexType>

</xsd:element>

2.2.2 XML schema example

Reference: Behavior.behavior

existing XML schema example

<xsd:element maxOccurs="*1*" minOccurs="*0*" name="*BEHAVIORS*">

<xsd:complexType>

<xsd:choice maxOccurs="*unbounded*" minOccurs="*0*">

<xsd:element name="*FUNCTION-BEHAVIOR*" type="*EA:FUNCTION-BEHAVIOR*"/>

</xsd:choice>

</xsd:complexType>

</xsd:element>

proposed XML schema example

<xsd:element maxOccurs=*"1"* minOccurs=*"0"* name=*" BEHAVIORS "*>

<xsd:complexType>

<xsd:choice maxOccurs=*"unbounded"* minOccurs=*"0"*>

<xsd:element name=*" FUNCTION-BEHAVIOR"* type=*"EA: FUNCTION-BEHAVIOR"*/>

</xsd:choice>

</xsd:complexType>

</xsd:element>

2.2.3 XML instance example

existing XML instance example

<BEHAVIORS>

<FUNCTION-BEHAVIOR>

…

<NAME>*behaviorRef1-functionBehavior1*</NAME>

</FUNCTION-BEHAVIOR>

<FUNCTION-BEHAVIOR>

…

<NAME>*behaviorRef1-functionBehavior2*</NAME>

</FUNCTION-BEHAVIOR>

</BEHAVIORS>

proposed XML instance example

<BEHAVIORS>

<FUNCTION-BEHAVIOR>

…

<NAME>*behaviorRef1-functionBehavior1*</NAME>

</FUNCTION-BEHAVIOR>

<FUNCTION-BEHAVIOR>

…

<NAME>*behaviorRef1-functionBehavior2*</NAME>

</FUNCTION-BEHAVIOR>

</BEHAVIORS>

## 2.3 row 8 : EReference\_contained0101Single

2.3.1 target pattern

existing target pattern

**TR\_APRXML\_00012** ::=

<xsd:element name="//roleXmlName//" minOccurs="//lowerMultiplicity//" maxOccurs="//upperMultiplicity//">

<xsd:complexType>

<xsd:choice minOccurs="//lowerMultiplicity//" maxOccurs="1">

(<xsd:element name="//(sub)typeXmlName//" type="//typeXmlNsPrefix : typeXmlName//"/>)+

</xsd:choice>

</xsd:complexType>

</xsd:element>

This declaration will be added into the xsd:group of the class (TR\_APRXML\_00001).

proposed target pattern

5j:EReference\_contained0101Single ::=

*if (type.length >1):*

<xsd:element name="//Name of Target Property (single)//" minOccurs="0|1" maxOccurs="1">

<xsd:complexType>

<xsd:choice minOccurs="//Minimum of Target Property//" maxOccurs="1 ">

(<xsd:element

name=" // XML Name of Target (Sub) Class // "

type="//namespace// ":" // XML Name of Target Class // " /> )+

</xsd:choice>

</xsd:complexType>

</xsd:element>

*if (NOT type.length >1):*

<xsd:element name='" // Name of Target Property (single) // "' type="// xml Name of Target Class //"

minOccurs='"//Minimum of Target Property//"' maxOccurs='"// Max of Target Property// "'>

</xsd:element>

2.3.2 XML schema example

Reference: FunctionFlowPort.defaultValue

existing XML schema example

<xsd:element maxOccurs="*1*" minOccurs="*0*" name="*DEFAULT-VALUE*">

<xsd:complexType>

<xsd:choice minOccurs="*0*" maxOccurs="*1*">

<xsd:element name="*EA-ARRAY-VALUE*" type="*EA:EA-ARRAY-VALUE*"/>

<xsd:element name="*EA-BOOLEAN-VALUE*" type="*EA:EA-BOOLEAN-VALUE*"/>

<xsd:element name="*EA-STRING-VALUE*" type="*EA:EA-STRING-VALUE*"/>

…

</xsd:choice>

</xsd:complexType>

</xsd:element>

proposed XML schema example

<xsd:element maxOccurs=*"1"* minOccurs=*"0"* name=*"DEFAULT-VALUE"*>

<xsd:complexType>

<xsd:choice maxOccurs=*"1"* minOccurs=*"0"*>

<xsd:element name=*"EA-ARRAY-VALUE"* type=*"EA:EA-ARRAY-VALUE"*/>

<xsd:element name=*"EA-BOOLEAN-VALUE"* type=*"EA:EA-BOOLEAN-VALUE"*/>

<xsd:element name=*"EA-STRING-VALUE"* type=*"EA:EA-STRING-VALUE"*/>

…

</xsd:choice>

</xsd:complexType>

</xsd:element>

2.3.3 XML instance model

existing XML instance example

<DEFAULT-VALUE>

<EA-STRING-VALUE>

<TYPE-TREF DEST="EA-STRING">*/EARoot/str1*</TYPE-TREF>

</EA-STRING-VALUE>

</DEFAULT-VALUE>

proposed XML instance example

<DEFAULT-VALUE>

<EA-STRING-VALUE>

<TYPE-TREF TYPE="EA-STRING">*/EARoot/str1*</TYPE-TREF>

</EA-STRING-VALUE>

</DEFAULT-VALUE>

## 2.4 row 9 : EReference\_contained0100Single (proposed to be replaced by EReference\_contained0101Single to avoid xsi:type, same as 2.3)

2.4.1 target pattern

existing target pattern

**TR\_APRXML\_00013** ::=

*if (types.length > 1) :*

<xsd:element name="//roleXmlName//" minOccurs="//lowerMultiplicity//" maxOccurs="//upperMultiplicity//">

<xsd:complexType>

<xsd:choice minOccurs="//lowerMultiplicity//" maxOccurs="1">

(<xsd:group ref="//typeXmlNsPrefix : typeXmlName//"/>)+

</xsd:choice>

</xsd:complexType>

</xsd:element>

*if (NOT type.length >1):*

<xsd:element name="//roleXmlName//" type="//typeXmlNsPrefix : typeXmlName//"

minOccurs="//lowerMultiplicity//" maxOccurs="//upperMultiplicity//"/>

proposed target pattern

5j:EReference\_contained0101Single ::=

*if (types.length > 1) :*

<xsd:element name='" // Name of Target Property (single) // "'

minOccurs='"//Minimum of Target Property//"' maxOccurs='"// Max of Target Property// "'>

<xsd:complexType>

<xsd:choice minOccurs='"//Minimum of Target Property//"' maxOccurs="1">

(<xsd:element

name='" // xml Name of Target (Sub) Class // "'

type="//namespace// ":" // xml Name of Target Class // "' /> )+

</xsd:choice>

</xsd:complexType>

</xsd:element>

*if (NOT type.length >1):*

<xsd:element name='" // Name of Target Property (single) // "' type="// xml Name of Target Class //"

minOccurs='"//Minimum of Target Property//"' maxOccurs='"// Max of Target Property// "'>

</xsd:element>

2.4.2 XML schema example

Reference: AgeConstraint.minimum

existing XML schema example

<xsd:element maxOccurs="*1*" minOccurs="*0*" name="*MINIMUM*" type="*EA:TIMEING-EXPRESSION*">

</xsd:element>

proposed XML schema example

<xsd:element maxOccurs=*"1"* minOccurs=*"0"* name=*"MINIMUM"* type=*"EA:TIMING-EXPRESSION"*/>

2.4.3 XML instance model

existing XML instance example

<AGE-CONSTRAINT>

<SHORT-NAME>*ageConstraint1*</SHORT-NAME>

…

<MINIMUM>

<TYPE-TREF DEST="EA-STRING">*/EARoot/str1*</TYPE-TREF>

</MINIMUM>

…

</AGE-CONSTRAINT>

proposed XML instance example

<AGE-CONSTRAINT>

<SHORT-NAME>*ageConstraint1*</SHORT-NAME>

…

<MINIMUM>

<TYPE-TREF TYPE="EA-STRING">*/EARoot/str1*</TYPE-TREF>

</MINIMUM>

…

</AGE-CONSTRAINT>

## 2.5 row 11 : EReference\_referenced1100Many( replaced by EReference\_ referenced 1001Many)

2.5.1 Target pattern:

existing target pattern

**TR\_APRXML\_00017** ::=

<xsd:element name="//roleXmlNamePlural//" minOccurs="//lowerMultiplicity//" maxOccurs="1">

<xsd:complexType>

<xsd:choice minOccurs="//lowerMultiplicity//" maxOccurs="//upperMultiplicity//">

<xsd:element name="//roleXmlName//">

</xsd:complexType>

<xsd:simpleContent>

<xsd:extension base="EA:REF">

<xsd:attribute name="DEST" type="//typeXmlNsPrefix : typeXmlName//-

-SUBTYPE-ENUM" use="required"/>

</xsd:extension>

</xsd:simpleContent>

</xsd:complexType>

</xsd:element>

</xsd:choice>

</xsd:complexType>

</xsd:element>

This declaration will be added into the xsd:group of the class (TR\_APRXML\_00001).

An xsd:simpleType that represents the lists possible concrete instances of the referenced class is created (TR\_APRXML\_00025):

**TR\_APRXML\_00025**::=

<xsd:simpleType name="//xmlName//--SUBTYPES-ENUM">

<xsd:restriction base="xsd:string">

(<xsd:enumeration value="//(sub) typeXmlName//"/>)+

</xsd:restriction>

</xsd:simpleType>

proposed target pattern

**5l:EReference\_referenced1100Many** ::=

<xsd:element name="//roleXmlNamePlural//" minOccurs="0 | 1" maxOccurs="1 ">

<xsd:complexType>

<xsd:choice minOccurs="//lowerMultiplicity//" maxOccurs="//upperMultiplicity//">

<xsd:element name="//roleXmlName//">

</xsd:complexType>

<xsd:simpleContent>

<xsd:extension base="EA:REF">

<xsd:attribute name="TYPE" type="//typeXmlNsPrefix : typeXmlName//-

-SUBTYPE-ENUM" use="required"/>

</xsd:extension>

</xsd:simpleContent>

</xsd:complexType>

</xsd:element>

</xsd:choice>

</xsd:complexType>

</xsd:element>

*For a general case:*

**5l:EReference\_referenced1100Many** ::=

<xsd:element name="//roleXmlNamePlural//" minOccurs="0 | 1" maxOccurs="1 ">

<xsd:complexType>

<xsd:choice minOccurs="//lowerMultiplicity//" maxOccurs="//upperMultiplicity//">

<xsd:element name="//roleXmlName//">

</xsd:complexType>

<xsd:attribute name="TYPE" type="//typeXmlNsPrefix : typeXmlName//--SUBTYPE-ENUM" use="required"/>

</xsd:complexType>

</xsd:element>

</xsd:choice>

</xsd:complexType>

</xsd:element>

*1) in case typeAttributeName is not "xsi:type"*

<xsd:element name="//roleXmlNamePlural//" minOccurs="//lowerMultiplicity//" maxOccurs="1">

<xsd:complexType>

<xsd:choice minOccurs="//lowerMultiplicity//" maxOccurs="//upperMultiplicity//">

<xsd:element name="//roleXmlName//">

</xsd:complexType>

<xsd:simpleContent>

<xsd:extension base="xsd:string">

<xsd:attribute name="*typeAttributeName*" type="//typeXmlNsPrefix : typeXmlName//--SUBTYPE-ENUM" use="optional"/>

</xsd:extension>

</xsd:simpleContent>

</xsd:complexType>

</xsd:element>

</xsd:choice>

</xsd:complexType>

</xsd:element>

*2) in case typeAttributeName is "xsi:type":*

<xsd:element name="//roleXmlNamePlural//" minOccurs="//lowerMultiplicity//" maxOccurs="1">

<xsd:complexType>

<xsd:choice minOccurs="//lowerMultiplicity//" maxOccurs="//upperMultiplicity//">

<xsd:element name="//roleXmlName//" type = "xsd:string"/>

</xsd:choice>

</xsd:complexType>

</xsd:element>

An xsd:simpleType that represents the lists possible concrete instances of the referenced class is created (8.ReferencedSimpleType):

**8. ReferencedSimpleType** ::=

<xsd:simpleType name="//xmlName//--SUBTYPES-ENUM">

<xsd:restriction base="xsd:string">

(<xsd:enumeration value="//(sub) typeXmlName//"/>)+

</xsd:restriction>

</xsd:simpleType>

2.5.2 XML schema example

Reference: FunctionBehavior.mode

existing XML schema example

<xsd:element maxOccurs=*"1"* minOccurs=*"0"* name=*"MODE-REFS"*>

<xsd:complexType>

<xsd:choice maxOccurs=*"unbounded"* minOccurs=*"0"*>

<xsd:element name=*"MODE-REF"*>

<xsd:complexType>

<xsd:simpleContent>

<xsd:extension base=*"EA:REF"*>

<xsd:attribute name=*"DEST"* type=*"EA:MODE--SUBTYPES-ENUM"* use=*"required"*/>

</xsd:extension>

</xsd:simpleContent>

</xsd:complexType>

</xsd:element>

</xsd:choice>

</xsd:complexType>

</xsd:element>

<xsd:simpleType name="*MODE--SUBTYPES-ENUM*">

<xsd:restriction base="*xsd:string*">

<xsd:enumeration value="*MODE*"/>

</xsd:restriction>

</xsd:simpleType>

proposed XML schema example

<xsd:element maxOccurs=*"1"* minOccurs=*"0"* name=*"MODE-REFS"*>

<xsd:complexType>

<xsd:choice maxOccurs=*"unbounded"* minOccurs=*"0"*>

<xsd:element name=*"MODE-REF"*>

<xsd:complexType>

<xsd:simpleContent>

<xsd:extension base=*"EA:REF"*>

<xsd:attribute name=*"TYPE"* type=*"EA:MODE--SUBTYPES-ENUM"* use=*"required"*/>

</xsd:extension>

</xsd:simpleContent>

</xsd:complexType>

</xsd:element>

</xsd:choice>

</xsd:complexType>

<xsd:simpleType name="*MODE--SUBTYPES-ENUM*">

<xsd:restriction base="*xsd:string*">

<xsd:enumeration value="*MODE*"/>

</xsd:restriction>

</xsd:simpleType>

2.5.3 XML instance example

existing XML instance example

<FUNCTION-BEHAVIOR>

<MODE-REFS>

<MODE-REF DEST="MODE">*/EARoot/modeGroup1/mode1*</MODE-REF>

<MODE-REF DEST="MODE">*/EARoot/modeGroup1/mode2*</MODE-REF>

</MODE-REFS>

</FUNCTION-BEHAVIOR>

proposed XML instance example

<FUNCTION-BEHAVIOR>

<MODE-REFS>

<MODE-REF TYPE="MODE">*/EARoot/modeGroup1/mode1*</MODE-REF>

<MODE-REF TYPE="MODE">*/EARoot/modeGroup1/mode2*</MODE-REF>

</MODE-REFS>

</FUNCTION-BEHAVIOR>

## 2.6 row 12 : EReference\_referenced0100Many

2.6.1 Target pattern:

existing target pattern

**TR\_APRXML\_00018** ::=

<xsd:element name="//roleXmlName//" minOccurs="//lowerMultiplicity//" maxOccurs="//upperMultiplicity//">

<xsd:complexType>

<xsd:simpleContent>

<xsd:extension base="EA:REF">

<xsd:attribute name="DEST" type="//typeXmlNsPrefix : typeXmlName//--SUBTYPES-ENUM" use="required"/>

</xsd:extension>

</xsd:simpleContent>

</xsd:complexType>

</xsd:element>

This declaration will be added into the xsd:group of the class (TR\_APRXML\_00001).

An xsd:simpleType that represents the lists possible concrete instances of the referenced class is created (TR\_APRXML\_00025).

proposed target pattern

**5m:EReference\_referenced0100Many** ::=

<xsd:element name="//roleXmlName//" minOccurs="//lowerMultiplicity//" maxOccurs="//upperMultiplicity//">

<xsd:complexType>

<xsd:simpleContent>

<xsd:extension base="EA:REF">

<xsd:attribute name="TYPE" type="//typeXmlNsPrefix : typeXmlName//--SUBTYPES-ENUM" use="required"/>

</xsd:extension>

</xsd:simpleContent>

</xsd:complexType>

</xsd:element>

*For a general case:*

*1) in case typeAttributeName is not "xsi:type"*

<xsd:element name="//roleXmlName//" minOccurs="//lowerMultiplicity//" maxOccurs="//upperMultiplicity//">

<xsd:complexType>

<xsd:simpleContent>

<xsd:extension base="xsd:string">

<xsd:attribute name="typeAttributeName" type="//typeXmlNsPrefix : typeXmlName//--SUBTYPES-ENUM" use="optional"/>

</xsd:extension>

</xsd:simpleContent>

</xsd:complexType>

</xsd:element>

*2) in case typeAttributeName is "xsi:type":*

<xsd:element name="roleXmlName" minOccurs="lowerMultiplicity" maxOccurs="upperMultiplicity" type="xsd:string"/>

An xsd:simpleType that represents the lists possible concrete instances of the referenced class is created (8.ReferencedSimpleType).

2.6.2 XML schema example

Reference: Refine\_refinedBy.identifiable\_context

existing XML schema example

<xsd:element maxOccurs="*unbounded*" minOccurs="*0*" name="*IDENTIFIABLE-CONTEXT-REF*">

<xsd:complexType>

<xsd:simpleContent>

<xsd:extension base="*EA:REF*">

<xsd:attribute name="*DEST*" type="*EA:IDENTIFIABLE--SUBTYPES-ENUM*" use="*required*"/>

</xsd:extension>

</xsd:simpleContent>

</xsd:complexType>

</xsd:element>

proposed XML schema example

<xsd:element maxOccurs="*unbounded*" minOccurs="*0*" name="*IDENTIFIABLE-CONTEXT-REF*">

<xsd:complexType>

<xsd:simpleContent>

<xsd:extension base="*EA:REF*">

<xsd:attribute name="*TYPE*" type="*EA:IDENTIFIABLE--SUBTYPES-ENUM*" use="*required*"/>

</xsd:extension>

</xsd:simpleContent>

</xsd:complexType>

</xsd:element>

2.6.3 XML instance example

existing XML instance example

<REFINED-BY-IREF>

<IDENTIFIABLE-CONTEXT-REF DEST="BEHAVIOR">*/EARoot/behavior1*</IDENTIFIABLE-CONTEXT-REF>

<IDENTIFIABLE-CONTEXT-REF DEST="BEHAVIOR">*/EARoot/behavior2*</IDENTIFIABLE-CONTEXT-REF>

…

</REFINED-BY-IREF>

proposed XML instance example

<REFINED-BY-IREF>

<IDENTIFIABLE-CONTEXT-REF TYPE="BEHAVIOR">*/EARoot/behavior1*</IDENTIFIABLE-CONTEXT-REF>

<IDENTIFIABLE-CONTEXT-REF TYPE="BEHAVIOR">*/EARoot/behavior2*</IDENTIFIABLE-CONTEXT-REF>

…

</REFINED-BY-IREF>

## 2.7 row 13 : EReference\_referenced0100Single

2.7.1 Target pattern:

existing target pattern

**TR\_APRXML\_00018** ::=

<xsd:element name="//roleXmlName//" minOccurs="//lowerMultiplicity//" maxOccurs="//upperMultiplicity//">

<xsd:complexType>

<xsd:simpleContent>

<xsd:extension base="EA:REF">

<xsd:attribute name="DEST" type="//typeXmlNsPrefix : typeXmlName//--SUBTYPES-ENUM" use="required"/>

</xsd:extension>

</xsd:simpleContent>

</xsd:complexType>

</xsd:element>

This declaration will be added into the xsd:group of the class (TR\_APRXML\_00001).

An xsd:simpleType that represents the lists possible concrete instances of the referenced class is created (TR\_APRXML\_00025).

proposed target pattern

**(same as 5m:ERference\_referenced0100Many)**

**5n:ERference\_referenced0100Single** ::=

<xsd:element name="//roleXmlName//" minOccurs="//lowerMultiplicity//" maxOccurs="//upperMultiplicity//">

<xsd:complexType>

<xsd:simpleContent>

<xsd:extension base="EA:REF">

<xsd:attribute name="TYPE" type="//typeXmlNsPrefix : typeXmlName//--SUBTYPES-ENUM" use="required"/>

</xsd:extension>

</xsd:simpleContent>

</xsd:complexType>

</xsd:element>

*For a general case:*

*1) in case typeAttributeName is not "xsi:type"*

<xsd:element name="//roleXmlName//" minOccurs="//lowerMultiplicity//" maxOccurs="//upperMultiplicity//">

<xsd:complexType>

<xsd:simpleContent>

<xsd:extension base="xsd:string">

<xsd:attribute name="typeAttributeName" type="//typeXmlNsPrefix : typeXmlName//--SUBTYPES-ENUM" use="optional"/>

</xsd:extension>

</xsd:simpleContent>

</xsd:complexType>

</xsd:element>

*2) in case typeAttributeName is "xsi:type":*

<xsd:element name="roleXmlName" minOccurs="lowerMultiplicity" maxOccurs="upperMultiplicity" type="xsd:string"/>

An xsd:simpleType that represents the lists possible concrete instances of the referenced class is created (8.ReferencedSimpleType).

2.7.2 XML schema example

Reference: FunctionBehavior.function

existing XML schema example

<xsd:element maxOccurs=*"1"* minOccurs=*"0"* name=*"FUNCTION-REF"*>

<xsd:complexType>

<xsd:simpleContent>

<xsd:extension base=*"EA:REF"*>

<xsd:attribute name=*"DEST"* type=*"EA:FUNCTION-TYPE--SUBTYPES-ENUM"* use=*"required"*/>

</xsd:extension>

</xsd:simpleContent>

</xsd:complexType>

</xsd:element>

proposed XML schema example

<xsd:element maxOccurs=*"1"* minOccurs=*"0"* name=*"FUNCTION-REF"*>

<xsd:complexType>

<xsd:simpleContent>

<xsd:extension base=*"EA:REF"*>

<xsd:attribute name=*"TYPE"* type=*"EA:FUNCTION-TYPE--SUBTYPES-ENUM"* use=*"required"*/>

</xsd:extension>

</xsd:simpleContent>

</xsd:complexType>

</xsd:element>

2.7.3 XML instance example

existing XML instance example

<FUNCTION-BEHAVIOR>

…

<FUNCTION-REF DEST="FUNCTIONAL-DEVICE">*/EARoot/functionDevice1*</FUNCTION-REF>

</FUNCTION-BEHAVIOR>

proposed XML instance example

<FUNCTION-BEHAVIOR>

…

<FUNCTION-REF TYPE="FUNCTIONAL-DEVICE">*/EARoot/functionDevice1*</FUNCTION-REF>

</FUNCTION-BEHAVIOR>

# 3. EAXML instance example

Existing EAXML example: xsd-test1.eaxml

Proposed EAXML example: xsd-test1-proposed.eaxml