### 8.1.1 Head elements of substitution groups

This clause is invoked if the global XSD *element* information item being translated is referenced by the *substitutionGroup* attribute of one or more other global *element* information item(s) in the set of schemas being translated (i.e. it is the head of an element substitution group) and the user has requested to generate TTCN-3 code allowing using element substitution (see clause 8).

Substitution group head elements shall be translated to TTCN-3 **union** types. The name of the **union** type shall be the result of applying clause 5.2.2 to the name composed of the header element's name and the postfix "\_group".

One alternative shall be added for the head element itself and one for each member of the substitution group. The first alternative (field) of the **union** type shall correspond to the head element. The alternatives corresponding to the member elements shall be added in an ordered manner, first alphabetically ordering the elements according to their target namespaces (elements with no target namespace first) and subsequently alphabetically ordering the elements with the same namespace based on their names. For each alternative the field name shall be the name applying clause 5.2.2 to the name of the XSD *element* corresponding to the given alternative. The type of the alternative shall be:

* the TTCN-3 type resulted by applying clause 7.3 to the head element, in the case of the head element;
* the TTCN-3 type resulted by applying clause 8.1.2 to the member element, in the case of the member elements (i.e. it shall reference the TTCN-3 type generated for the given global XSD *element* information item).

NOTE 1: In XSD, substitution group membership is transitive, i.e. the members of a substitution group (ESG1) whose head is a member of another substitution group (ESG2) are all also members of the second substitution group (ESG2).

If the value of the head element's *abstract* attribute is "*true*", the "abstract" encoding instruction has to be attached to the field corresponding to the head element (i.e. to the first field).

NOTE 2: If the value of a member element's *abstract* attribute is "true", the "abstract" encoding instruction is attached to the TTCN-3 type generated for that element, according to clause 7.1.9.

If the head element's effective block value (see clause 7.1.10) is "*#all*" or "*substitution*", the "block" encoding instruction shall be attached to all fields of the **union** type except the field corresponding to the head element (the first field).

If the head element's effective block value (see clause 7.1.10) is "*restriction*" or "*extension*" the "block" encoding instruction shall be attached to all fields, generated for group member elements with a type, which has been derived from the type of the head element by *restriction* or by *extension*, respectively, at any step along the derivation path.

NOTE 3: The TTCN-3 syntax allows to attach the same attribute to several fields of the same structured type in one with attribute.

Finally, the **union** type shall be appended with the "untagged" encoding instruction.

When translating XSD references to the head element to TTCN-3, the TTCN-3 **union** type generated according to this clause shall be used.

EXAMPLE 1: Substitution group:

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

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

 targetNamespace="http://www.example.org/SimpleCase"

 xmlns:tns="http://www.example.org/SimpleCase" >

 <!-- THE HEAD ELEMENT -->

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

 <!-- SUBSTITUTION ELEMENT OF THE SAME TYPE AS THE HEAD -->

 <xsd:element name="member1" substitutionGroup="tns:head"/>

 <!-- SUBSTITUTION ELEMENT OF A TYPE RESTRICTING THE TYPE OF THE HEAD -->

 <xsd:simpleType name="stringEnum">

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

 <xsd:enumeration value="something"/>

 <xsd:enumeration value="else"/>

 </xsd:restriction>

 </xsd:simpleType>

 <xsd:element name="member2" type="tns:stringEnum" substitutionGroup="tns:head"/>

 <!-- SUBSTITUTION ELEMENT OF A TYPE EXTENDING THE TYPE OF THE HEAD -->

 <xsd:complexType name="complexEnum">

 <xsd:simpleContent>

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

 <xsd:attribute name="foo" type="xsd:float"/>

 <xsd:attribute name="bar" type="xsd:integer"/>

 </xsd:extension>

 </xsd:simpleContent>

 </xsd:complexType>

 <xsd:element name="member3" type="tns:complexEnum" substitutionGroup="tns:head"/>

 <!-- TOP LEVEL ELEMENT TO DEMONSTRATE SUBSTITUTION -->

 <xsd:element name="ize">

 <xsd:complexType>

 <xsd:sequence>

 <xsd:element ref="tns:head" minOccurs="0" maxOccurs="unbounded"/>

 </xsd:sequence>

 </xsd:complexType>

 </xsd:element>

</xsd:schema>

//Is translated to TTCN-3 as:

**module** http\_www\_example\_org\_SimpleCase {

 /\* SUBSTITUTION ELEMENT OF THE SAME TYPE AS THE HEAD \*/

 **type** XSD.String Member1

 **with** {

 **variant** "name as uncapitalized";

 **variant** "element";

 };

 /\* SUBSTITUTION ELEMENT OF A TYPE RESTRICTING THE TYPE OF THE HEAD \*/

 **type** **enumerated** StringEnum { something, else }

 **with** {

 **variant** "name as uncapitalized";

 };

 **type** StringEnum Member2

 **with** {

 **variant** "name as uncapitalized";

 **variant** "element";

 };

 /\* SUBSTITUTION ELEMENT OF A TYPE EXTENDING THE TYPE OF THE HEAD \*/

 **type** **record** ComplexEnum

 {

 XSD.Integer bar **optional**,

 XSD.Float foo **optional**,

 XSD.String base

 }

 **with** {

 **variant** "name as uncapitalized";

 **variant** (bar) "attribute";

 **variant** (foo) "attribute";

 **variant** (base) "untagged";

 };

 **type** ComplexEnum Member3

 **with** {

 **variant** "name as uncapitalized";

 **variant** "element";

 };

 /\* THE HEAD ELEMENT \*/

 **type union** Head\_group {

 XSD.String head,

 Member1 member1,

 Member2 member2,

 Member3 member3

 }

 **with** {

 **variant** "untagged"

}

 /\* TOP LEVEL ELEMENT TO DEMONSTRATE SUBSTITUTION \*/

 **type record** Ize

 {

 **record of** Head\_group head\_list

 }

 **with** {

 **variant** "name as uncapitalized";

 **variant** "element";

 **variant** (head\_list) "untagged";

 }

} **with** {

 **encode** "XML";

 **variant** "namespace as 'http://www.example.org/SimpleCase' prefix 'tns'";

 **variant** "controlNamespace 'http://www.w3.org/2001/XMLSchema-instance' prefix 'xsi'";

}

//and the template

**template** Ize t\_Ize := {

  head\_list := {

 { head := "anything" },

 { member1 := "any thing" },

 { member2 := something },

 { member3 := { bar:= 5, foo := **omit**, base := "anything else" }

 }

}

//will be encoded in XML as:

<?xml version="1.0" encoding="UTF-8"?>

<tns:ize

 xmlns:tns="http://www.example.org/SimpleCase"

 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

 xsi:schemaLocation="http://www.example.org/SimpleCase SimpleCase.xsd">

 <tns:head>anything</tns:head>

 <tns:member1>any thing</tns:member1>

 <tns:member2>something</tns:member2>

 <tns:member3>akarmi</tns:member3>

 <tns:member3 bar="5" >anything else</tns:member3>

</tns:ize>

EXAMPLE 2: Effect of the block and abstract attributes on element substitution:

<?xml version="1.0" encoding="UTF-8"?>

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

 targetNamespace="http://www.example.org/BlockRestriction">

 <!-- THE HEAD ELEMENT -->

 <xsd:element name=*"head"* type=*"xsd:string"* block=*"restriction"* abstract=*"true"*/>

<!-- Substitution group members member1, member2, member3, their types and element *"ize"* are the same as in example 1 above, hence not repeated here -->

</xsd:schema>

//Is translated to TTCN-3 as:

// TTCN-3 type definitions Member1, StringEnum, Member2, ComplexEnum, Member3 and Ize

// are the same as in example 1 above, hence not repeated here

**module** http\_www\_example\_org\_BlockRestriction {

 /\* THE HEAD ELEMENT \*/

 **type union** Head\_group {

 XSD.String head,

 Member1 member1,

 Member2 member2,

 Member3 member3

 }

 **with** {

 **variant** "untagged";

 **variant** (head) "abstract";

 **variant** (member2) "block"

}

 /\* Substitution group members member1, member2, member3, their types and element "ize" are
 the same as in example 1 above, hence not repeated here \*/

} **with** {

 **encode** "XML";

 **variant** "namespace as 'http://www.example.org/BlockRestriction' prefix 'tns'";

 **variant** "controlNamespace 'http://www.w3.org/2001/XMLSchema-instance' prefix 'xsi'";

}

//and the template

**template** Ize t\_Ize := {

  head\_list := {

 { head := "anything" },

 { member1 := "any thing" },

 { member2 := something },

 { member3 := { bar:= 5, foo := **omit**, base := "anything else" }

 }

}

//will be encoded in XML as:

<?xml version="1.0" encoding="UTF-8"?>

<tns:ize

 xmlns:tns="http://www.example.org/*BlockRestriction* "

 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

 xsi:schemaLocation="http://www.example.org/*BlockRestriction* *BlockRestriction*.xsd">

<!-- allowed to send but causes a decoding failure if present in the received XML document
 ( the head element is abstract) -->

 <tns:head>anything</tns:head>

<!-- OK to send and receive -->

 <tns:member1>any thing</tns:member1>

<!-- allowed to send but causes a decoding failure if present in the received XML document
 ( the type of member2 is derived by restriction in XSD) -->

 <tns:member2>something</tns:member2>

<!-- OK to send and receive (the type of member3 is derived by extension in XSD) -->

 <tns:member3>akarmi</tns:member3>

 <tns:member3 bar="5" >anything else</tns:member3>

</tns:ize>

EXAMPLE 3: Blocking substitution:

<?xml version="1.0" encoding="UTF-8"?>

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

 targetNamespace="http://www.example.org/BlockAll"

 xmlns:tns="http://www.example.org/BlockAll">

 <!-- THE HEAD ELEMENT -->

 <xsd:element name="headNoSubstition" type="xsd:string" block="#all"/>

 <xsd:element name="groupMember1" type="xsd:string" substitutionGroup="tns:headNoSubstition"/>

 <xsd:element name="groupMember2" type="xsd:string" substitutionGroup="tns:headNoSubstition"/>

 <!-- TOP LEVEL ELEMENT TO DEMONSTRATE SUBSTITUTION -->

 <xsd:element name="ize2">

 <xsd:complexType>

 <xsd:sequence>

 <xsd:element ref="tns:headNoSubstition" minOccurs="0" maxOccurs="unbounded"/>

 </xsd:sequence>

 </xsd:complexType>

 </xsd:element>

</xsd:schema>

//Is translated to TTCN-3 as:

**module** http\_www\_example\_org\_BlockAll {

 **type** XSD.String GroupMember1

 **with** {

 **variant** "name as uncapitalized";

 **variant** "element";

 };

 **type** XSD.String GroupMember2

 **with** {

 **variant** "name as uncapitalized";

 **variant** "element";

 };

 /\* THE HEAD ELEMENT \*/

 **type union** HeadNoSubstition\_group {

 XSD.String headNoSubstition,

 GroupMember1 groupMember1,

 GroupMember2 groupMember2

 }

 **with** {

 **variant** "untagged";

 **variant** (groupMember1, groupMember2) "block"

}

 /\* TOP LEVEL ELEMENT TO DEMONSTRATE SUBSTITUTION \*/

 **type record** Ize2

 {

 **record of** HeadNoSubstition\_group head\_list

 }

 **with** {

 **variant** "name as uncapitalized";

 **variant** "element";

 **variant** (head\_list) "untagged";

 };

} **with** {

 **encode** "XML";

 **variant** "namespace as 'http://www.example.org/BlockAll' prefix 'tns'";

 **variant** "controlNamespace 'http://www.w3.org/2001/XMLSchema-instance' prefix 'xsi'";

}

//and the template

**template** Ize2 t\_Ize2 := {

  head\_list := {

 { headNoSubstition := "anything" },

 { *groupMember1* := "any thing" },

 { *groupMember2* := "something" }

 }

}

//will be encoded in XML as:

<?xml version="1.0" encoding="UTF-8"?>

<tns:ize

 xmlns:tns="http://www.example.org/*BlockAll* "

 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

 xsi:schemaLocation="http://www.example.org/*BlockAll* *BlockAll*.xsd">

<!-- OK to send and receive -->

 <tns:headNoSubstition>anything</tns:headNoSubstition>

<!-- allowed to send but causes a decoding failure if present in the received XML document
 (all substitutions are disallowed) -->

 <tns:groupMember1>any thing</tns:groupMember1>

<!-- allowed to send but causes a decoding failure if present in the received XML document
 (all substitutions are disallowed) -->

 <tns:groupMember2>something</tns:groupMember2>

</tns:ize>