

XML Schema for WSML identifiers

Table of Contents

- [Schema Document Properties](#)
- [Global Schema Components](#)
 - Simple Type: [wsmlIRI](#)
 - Complex Type: [wsmlAnyValue](#)
 - Complex Type: [wsmlID](#)
 - Complex Type: [wsmlVariable](#)

[top](#)

Schema Document Properties

Target Namespace	http://www.wsmo.org/wsml/wsml-syntax#
Element and Attribute Namespaces	<ul style="list-style-type: none">• Global element and attribute declarations belong to this schema's target namespace.• By default, local element declarations belong to this schema's target namespace.• By default, local attribute declarations have no namespace.
Documentation	version: \$Revision: 1.17 \$ date: \$Date: 2005/09/13 08:50:22 \$ author: Jos de Bruijn this schema is a module, which belongs to the WSML/XML schema specification. This schema provides the necessary definitions for the identifiers in WSML/XML.

Declared Namespaces

Prefix	Namespace
Default namespace	http://www.wsmo.org/wsml/wsml-syntax#
xml	http://www.w3.org/XML/1998/namespace
xs	http://www.w3.org/2001/XMLSchema

Schema Component Representation

```
<xs:schema targetNamespace="http://www.wsmo.org/wsml/wsml-syntax#"
  elementFormDefault="qualified" attributeFormDefault="unqualified">
  ...
</xs:schema>
```

[top](#)

Global Schema Components

Simple Type: **wsmlIRI**

<i>Super-types:</i>	None
<i>Sub-types:</i>	None

Name	wsmlIRI
Content	<ul style="list-style-type: none">• Union of following types:<ul style="list-style-type: none">◦ xs:anyURI

Documentation

The basic kind of identifier in WSMML: an IRI. Note that when translating the standard WSMML syntax to XML, Qualified names are resolved to full IRIs.

Schema Component Representation

```
<xs:simpleType name="wsmmlIRI">
  <xs:union memberTypes=" xs:anyURI"/>
</xs:simpleType>
```

[top](#)

Complex Type: **wsmmlAnyValue**

Super-types: None

Sub-types: None

Name	wsmmlAnyValue
Abstract	no
Documentation	A data value in WSMML. The type is required. Because we allow complex constructed data values, a data value can have multiple arguments. The arguments are strictly ordered. The content is mixed; it is thus possible to use both regular text and tags which represent the arguments. Regular text is interpreted as the first argument.

XML Instance Representation

```
<...
  type=" wsmmlIRI [1]">
  <!-- Mixed content -->
  <argument> xs:string </argument> [0..*]
</...>
```

Schema Component Representation

```
<xs:complexType name="wsmmlAnyValue" mixed="true">
  <xs:sequence>
    <xs:element name="argument" type=" xs:string " minOccurs="0"
      maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="type" type=" wsmmlIRI " use="required"/>
</xs:complexType>
```

[top](#)

Complex Type: **wsmmlID**

Super-types: [xs:string](#) < **wsmmlID** (by extension)

Sub-types: None

Name	wsmIID
Abstract	no
Documentation	wsmIID corresponds with either an IRI, a string, an integer, a decimal, a variable or an anonymous identifier. The type of the identifier is indicated in the attribute 'type'. The IRIs identifying the types IRI, string, integer and decimal correspond with the IRIs of the datatypes as defined in Appendix C of the WSMML specification. The type variable is identified with the IRI http://www.wsmo.org/2004/wsmml#variable , the type anonymousID is identified with http://www.wsmo.org/2004/wsmml#anonymousID .

XML Instance Representation

```
<...
  type=" wsmIID [1]">
  xs:string
</...>
```

Schema Component Representation

```
<xs:complexType name="wsmIID">
  <xs:simpleContent>
    <xs:extension base=" xs:string ">
      <xs:attribute name="type" type=" wsmIID " use="required"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

[top](#)

Complex Type: **wsmVariable**

<i>Super-types:</i>	None
<i>Sub-types:</i>	None

Name	wsmVariable
Abstract	no
Documentation	A variable in a WSMML logical expression.

XML Instance Representation

```
<...
  name=" xs:string [1]" />
```

Schema Component Representation

```
<xs:complexType name="wsmVariable">
  <!--new version of variable corresponds to how all other are treated.-->
  <xs:attribute name="name" type=" xs:string " use="required"/>
  <!--xs:simpleContent> <xs:extension base="xs:string"> <xs:attribute
name="name" use="required"> <xs:simpleType> <xs:restriction
base="xs:string"> <xs:pattern value="'?'\([a-z][A-Z][0-9][0x4E00 -
0x9FA5][0x3007][0x3021 - 0x3029])+' ' /> </xs:restriction>
</xs:simpleType> </xs:attribute> </xs:extension> </xs:simpleContent-->
  <!-- old version <xs:restriction base="xs:string"> <xs:pattern
value="'?'\([a-z][A-Z][0-9][0x4E00 - 0x9FA5][0x3007][0x3021 -
0x3029])+' ' /> </xs:restriction> -->
</xs:complexType>
```

[top](#)

