

# XML Schema for WSML

## Table of Contents

- [Schema Document Properties](#)
- [Global Schema Components](#)
  - [Element: wsml](#)
  - [Element: importsOntology](#)
  - [Element: usesMediator](#)
  - [Element: sharedVariables](#)
  - [Element: precondition](#)
  - [Element: assumption](#)
  - [Element: postcondition](#)
  - [Element: effect](#)
  - [Element: choreography](#)
  - [Element: orchestration](#)
  - [Element: goal](#)
  - [Element: nonFunctionalProperties](#)
  - [Element: webService](#)
  - [Element: capability](#)
  - [Element: interface](#)
  - [Element: ooMediator](#)
  - [Element: ggMediator](#)
  - [Element: wgMediator](#)
  - [Element: wwMediator](#)
  - [Element: ontology](#)
  - [Element: axiom](#)
  - [Complex Type: axiomType](#)
  - [Element: concept](#)
  - [Element: attribute](#)
  - [Element: relation](#)
  - [Complex Type: relationType](#)
  - [Element: parameters](#)
  - [Element: instance](#)
  - [Element: attributeValue](#)
  - [Element: relationInstance](#)
  - [Element: parameterValue](#)

[top](#)

## Schema Document Properties

<b>Target Namespace</b>	<a href="http://www.wsmo.org/wsml/wsml-syntax#">http://www.wsmo.org/wsml/wsml-syntax#</a>
<b>Element and Attribute Namespaces</b>	<ul style="list-style-type: none"><li>• Global element and attribute declarations belong to this schema's target namespace.</li><li>• By default, local element declarations belong to this schema's target namespace.</li><li>• By default, local attribute declarations have no namespace.</li></ul>
<b>Schema Composition</b>	<ul style="list-style-type: none"><li>• This schema imports schema(s) from the following namespace(s):<ul style="list-style-type: none"><li>◦ <a href="http://purl.org/dc/elements/1.1/">http://purl.org/dc/elements/1.1/</a> (at <a href="http://dublincore.org/schemas/xmls/qdc/2003/04/02/dc.xsd">http://dublincore.org/schemas/xmls/qdc/2003/04/02/dc.xsd</a>)</li></ul></li><li>• This schema includes components from the following schema document(s):<ul style="list-style-type: none"><li>◦ <a href="http://www.wsmo.org/TR/d16/d16.1/v0.2/xml-syntax/wsml-expr.xsd">http://www.wsmo.org/TR/d16/d16.1/v0.2/xml-syntax/wsml-expr.xsd</a></li><li>◦ <a href="http://www.wsmo.org/TR/d16/d16.1/v0.2/xml-syntax/wsml-identifiers.xsd">http://www.wsmo.org/TR/d16/d16.1/v0.2/xml-syntax/wsml-identifiers.xsd</a></li></ul></li></ul>

<b>Documentation</b>	version: \$Revision: 1.21 \$ date: \$Date: 2005/03/12 18:03:59 \$ author: Jos de Bruijn, Re Krummenacher this document provides an XML Schema for the XML version of the WSML syntax The syntax presented in this document can be used for all WSML variant: The WSML variant is identified through the 'variant' attribute of the root element 'wsml'. / modular approach has been chosen when creating this schema. This schema contains 1 conceptual syntax. Separate schemas have been developed for the WSML identifiers ar for the WSML logical expressions. These schemas are therefore included in this schem:
----------------------	--

## Declared Namespaces

Prefix	Namespace
Default namespace	<a href="http://www.wsmo.org/wsml/wsml-syntax#">http://www.wsmo.org/wsml/wsml-syntax#</a>
xml	http://www.w3.org/XML/1998/namespace
dc	http://purl.org/dc/elements/1.1/
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:include
    schemaLocation="http://www.wsmo.org/TR/d16/d16.1/v0.2/xml-syntax/wsml-expr.xsd" />
  <xs:include
    schemaLocation="http://www.wsmo.org/TR/d16/d16.1/v0.2/xml-syntax/wsml-identifiers.xsd" />
  <xs:import namespace="http://purl.org/dc/elements/1.1/"
    schemaLocation="http://dublincore.org/schemas/xmls/qdc/2003/04/02/dc.xsd" />
  ...
</xs:schema>
```

[top](#)

## Global Schema Components

### Element: **wsml**

<b>Name</b>	wsml
<b>Type</b>	Locally-defined complex type
<b><u>Nilable</u></b>	no
<b><u>Abstract</u></b>	no
<b>Documentation</b>	The element 'wsml' is the preferred root element for any WSML specification. It allows the user to explicitly specify, through the 'variant' attribute, which WSML variant is used. If the WSML variant has not been specified, the application has to "guess" the variant and in the worst case, WSML-Full will be assumed.

### XML Instance Representation

```
<wsml
  variant=" wsmlIRI [0..1]">
  Start Sequence [0..*]
    Allow any elements from any namespace (strict validation). [1]
  End Sequence
</wsml>
```

## Schema Component Representation

```
<xs:element name="wsm1" >
  <xs:complexType>
    <xs:sequence minOccurs="0" maxOccurs="unbounded" >
      <xs:any/>
    </xs:sequence>
    <xs:attribute name="variant" type=" wsm1IRI "/>
  </xs:complexType>
</xs:element>
```

[top](#)

## Element: **importsOntology**

<b>Name</b>	importsOntology
<b>Type</b>	<a href="#">wsm1IRI</a>
<b>Nilable</b>	no
<b>Abstract</b>	no
<b>Documentation</b>	This element is used for indicating which ontologies to import. Importing ontologies is the most basic (and inflexible) modularization mechanism. The result of the import is merely the union of the axioms in the importing and the imported ontology.

### XML Instance Representation

```
<importsOntology> wsm1IRI </importsOntology>
```

### Schema Component Representation

```
<xs:element name="importsOntology" type=" wsm1IRI "/>
```

[top](#)

## Element: **usesMediator**

<b>Name</b>	usesMediator
<b>Type</b>	<a href="#">wsm1IRI</a>
<b>Nilable</b>	no
<b>Abstract</b>	no
<b>Documentation</b>	Mediators can be used to mediate between any top-level WSMO elements, i.e. ontologies, goals, and web services. The most frequent use of mediators is to resolve heterogeneity between ontologies. Thus, mediators provide a more flexible way for modularization of ontologies.

### XML Instance Representation

```
<usesMediator> wsm1IRI </usesMediator>
```

### Schema Component Representation

```
<xs:element name="usesMediator" type=" wsm1IRI "/>
```

[top](#)

---

## Element: **sharedVariables**

<b>Name</b>	sharedVariables
<b>Type</b>	Locally-defined complex type
<b><u>Nilable</u></b>	no
<b><u>Abstract</u></b>	no
<b>Documentation</b>	A list of shared variables for the capability

### XML Instance Representation

```
<sharedVariables>
  <variable> wsmlVariable </variable> [1..*]
</sharedVariables>
```

### Schema Component Representation

```
<xs:element name="sharedVariables">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="variable" type=" wsmlVariable "
        minOccurs="1" maxOccurs="unbounded" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

[top](#)

---

## Element: **precondition**

<b>Name</b>	precondition
<b>Type</b>	<a href="#">axiomType</a>
<b><u>Nilable</u></b>	no
<b><u>Abstract</u></b>	no
<b>Documentation</b>	A pre-condition is an axiom describing the state of the information space before executing the web service.

### XML Instance Representation

```
<precondition
  name=" wsmlIRI [0..1]">
  Start Sequence [0..1]
  <nonFunctionalProperties> ... </nonFunctionalProperties> [0..1]
  <definedBy> logicalExpressionType </definedBy> [1]
  End Sequence
</precondition>
```

### Schema Component Representation

```
<xs:element name="precondition" type=" axiomType "/>
```

[top](#)

---

## Element: **assumption**

<b>Name</b>	assumption
<b>Type</b>	<a href="#">axiomType</a>
<b>Nilable</b>	no
<b>Abstract</b>	no
<b>Documentation</b>	An assumption is an axiom describing the state of the world before executing the web service.

#### XML Instance Representation

```

<assumption
  name=" wsmlIRI [0..1]">
  Start Sequence [0..1]
    <nonFunctionalProperties> ... </nonFunctionalProperties> [0..1]
    <definedBy> logicalExpressionType </definedBy> [1]
  End Sequence
</assumption>

```

#### Schema Component Representation

```

<xs:element name="assumption" type=" axiomType "/>

```

[top](#)

### Element: **postcondition**

<b>Name</b>	postcondition
<b>Type</b>	<a href="#">axiomType</a>
<b>Nilable</b>	no
<b>Abstract</b>	no
<b>Documentation</b>	A post-condition is an axiom describing the state of the information space after executing the web service.

#### XML Instance Representation

```

<postcondition
  name=" wsmlIRI [0..1]">
  Start Sequence [0..1]
    <nonFunctionalProperties> ... </nonFunctionalProperties> [0..1]
    <definedBy> logicalExpressionType </definedBy> [1]
  End Sequence
</postcondition>

```

#### Schema Component Representation

```

<xs:element name="postcondition" type=" axiomType "/>

```

[top](#)

### Element: **effect**

<b>Name</b>	effect
<b>Type</b>	<a href="#">axiomType</a>
<b><u>Nilable</u></b>	no
<b><u>Abstract</u></b>	no
<b>Documentation</b>	An effect is an axiom describing the state of the world after executing the web service.

#### XML Instance Representation

```
<effect
  name=" wsmlIRI [0..1]">
  Start Sequence [0..1]
    <nonFunctionalProperties> ... </nonFunctionalProperties> [0..1]
    <definedBy> logicalExpressionType </definedBy> [1]
  End Sequence
</effect>
```

#### Schema Component Representation

```
<xs:element name="effect" type=" axiomType "/>
```

[top](#)

### Element: choreography

<b>Name</b>	choreography
<b>Type</b>	<a href="#">wsmlIRI</a>
<b><u>Nilable</u></b>	no
<b><u>Abstract</u></b>	no
<b>Documentation</b>	A reference to a choreography description.

#### XML Instance Representation

```
<choreography> wsmlIRI </choreography>
```

#### Schema Component Representation

```
<xs:element name="choreography" type=" wsmlIRI "/>
```

[top](#)

### Element: orchestration

<b>Name</b>	orchestration
<b>Type</b>	<a href="#">wsmlIRI</a>
<b><u>Nilable</u></b>	no
<b><u>Abstract</u></b>	no
<b>Documentation</b>	A reference to an orchestration description.

#### XML Instance Representation

```
<orchestration> wsmlIRI </orchestration>
```

## Schema Component Representation

```
<xs:element name="orchestration" type=" wsmlIRI "/>
```

[top](#)

## Element: goal

<b>Name</b>	goal
<b>Type</b>	Locally-defined complex type
<b><u>Nilable</u></b>	no
<b><u>Abstract</u></b>	no
<b>Documentation</b>	A goal specification.

## XML Instance Representation

```
<goal  
  name=" wsmlIRI [0..1]">  
  <nonFunctionalProperties> ... </nonFunctionalProperties> [0..1]  
  <importsOntology> ... </importsOntology> [0..*]  
  <usesMediator> ... </usesMediator> [0..*]  
  <postcondition> ... </postcondition> [0..*]  
  <effect> ... </effect> [0..*]  
</goal>
```

## Schema Component Representation

```
<xs:element name="goal">  
  <xs:complexType>  
    <xs:sequence>  
      <xs:element ref=" nonFunctionalProperties " minOccurs="0"/>  
      <xs:element ref=" importsOntology " minOccurs="0"  
maxOccurs="unbounded"/>  
      <xs:element ref=" usesMediator " minOccurs="0"  
maxOccurs="unbounded"/>  
      <xs:element ref=" postcondition " minOccurs="0"  
maxOccurs="unbounded"/>  
      <xs:element ref=" effect " minOccurs="0"  
maxOccurs="unbounded"/>  
    </xs:sequence>  
    <xs:attribute name="name" type=" wsmlIRI " use="optional"/>  
  </xs:complexType>  
</xs:element>
```

[top](#)

## Element: nonFunctionalProperties

<b>Name</b>	nonFunctionalProperties
<b>Type</b>	Locally-defined complex type
<b><u>Nilable</u></b>	no
<b><u>Abstract</u></b>	no
<b>Documentation</b>	Any element can be used inside the non-functional properties. Non-functional properties function as annotation for the containing element. The recommended set of non-functional properties is the set of elements offered by Dublin Core. WSML further defines one element, named 'version' and a number of web-service specific elements, named 'performance', 'reliability', 'security', 'scalability', 'robustness', 'accuracy', 'transactional', 'trust', 'financial', and 'networkRelatedQoS'.

### XML Instance Representation

```
<nonFunctionalProperties>
  <attributeValue> ... </attributeValue> [0..*]
</nonFunctionalProperties>
```

### Schema Component Representation

```
<xs:element name="nonFunctionalProperties">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="attributeValue" minOccurs="0"
        maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

[top](#)

## Element: **webService**

<b>Name</b>	webService
<b>Type</b>	Locally-defined complex type
<b><u>Nilable</u></b>	no
<b><u>Abstract</u></b>	no
<b>Documentation</b>	A web service specification.

### XML Instance Representation

```
<webService
  name=" wsmlIRI [0..1]">
  <nonFunctionalProperties> ... </nonFunctionalProperties> [0..1]
  <importsOntology> ... </importsOntology> [0..*]
  <ooMediator> ... </ooMediator> [0..*]
  <capability> ... </capability> [1]
  <interface> ... </interface> [0..*]
</webService>
```

### Schema Component Representation

```
<xs:element name="webService">
  <xs:complexType>
    <xs:sequence>
```

```

<xs:element ref=" nonFunctionalProperties " minOccurs="0"/>
<xs:element ref=" importsOntology " minOccurs="0"
maxOccurs="unbounded"/>
<xs:element ref=" ooMediator " minOccurs="0"
maxOccurs="unbounded"/>
<xs:element ref=" capability "/>
<xs:element ref=" interface " minOccurs="0"
maxOccurs="unbounded"/>
</xs:sequence>
<xs:attribute name="name" type=" wsmlIRI " use="optional"/>
</xs:complexType>
</xs:element>

```

[top](#)

## Element: **capability**

<b>Name</b>	capability
<b>Type</b>	Locally-defined complex type
<b>Nilable</b>	no
<b>Abstract</b>	no
<b>Documentation</b>	A web service capability specification.

### XML Instance Representation

```

<capability
  name=" wsmlIRI [0..1]">
  <nonFunctionalProperties> ... </nonFunctionalProperties> [0..1]
  <importsOntology> ... </importsOntology> [0..*]
  <usesMediator> ... </usesMediator> [0..*]
  <sharedVariables> ... </sharedVariables> [0..1]
  <precondition> ... </precondition> [0..*]
  <assumption> ... </assumption> [0..*]
  <postcondition> ... </postcondition> [0..*]
  <effect> ... </effect> [0..*]
</capability>

```

### Schema Component Representation

```

<xs:element name=" capability">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref=" nonFunctionalProperties " minOccurs="0"/>
      <xs:element ref=" importsOntology " minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element ref=" usesMediator " minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element ref=" sharedVariables " minOccurs="0"/>
      <xs:element ref=" precondition " minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element ref=" assumption " minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element ref=" postcondition " minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element ref=" effect " minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="name" type=" wsmlIRI " use="optional"/>
  </xs:complexType>
</xs:element>

```

## Element: **interface**

<b>Name</b>	interface
<b>Type</b>	Locally-defined complex type
<b><u>Nilable</u></b>	no
<b><u>Abstract</u></b>	no
<b>Documentation</b>	A web service interface specification.

### XML Instance Representation

```
<interface
  name=" wsmlIRI [0..1]">
  <nonFunctionalProperties> ... </nonFunctionalProperties> [0..1]
  <importsOntology> ... </importsOntology> [0..*]
  <usesMediator> ... </usesMediator> [0..*]
  <choreography> ... </choreography> [1]
  <orchestration> ... </orchestration> [1]
</interface>
```

### Schema Component Representation

```
<xs:element name="interface">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref=" nonFunctionalProperties " minOccurs="0"/>
      <xs:element ref=" importsOntology " minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element ref=" usesMediator " minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element ref=" choreography "/>
      <xs:element ref=" orchestration "/>
    </xs:sequence>
    <xs:attribute name="name" type=" wsmlIRI " use="optional"/>
  </xs:complexType>
</xs:element>
```

## Element: **ooMediator**

<b>Name</b>	ooMediator
<b>Type</b>	Locally-defined complex type
<b><u>Nilable</u></b>	no
<b><u>Abstract</u></b>	no
<b>Documentation</b>	The element ooMediator is used for both declaration of ooMediators and reference to existing ooMediators. In case no elements are nested inside the ooMediator, the element is interpreted as a reference to the ooMediator indicated with the attribute 'name'. If elements are nested inside the ooMediator, it is interpreted as a specification of a new ooMediator whose identifier is specified in the attribute 'name'.

### XML Instance Representation

```

<ooMediator
  name=" wsmlIRI [0..1]">
  <nonFunctionalProperties> ... </nonFunctionalProperties> [0..1]
  <importsOntology> ... </importsOntology> [0..*]
  <source> wsmlIRI </source> [1..*]
  <target> wsmlIRI </target> [0..1]
  <usesService> wsmlIRI </usesService> [0..1]
</ooMediator>

```

### Schema Component Representation

```

<xs:element name="ooMediator">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref=" nonFunctionalProperties " minOccurs="0"/>
      <xs:element ref=" importsOntology " minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element name="source" type=" wsmlIRI " minOccurs="1"
maxOccurs="unbounded"/>
      <xs:element name="target" type=" wsmlIRI " minOccurs="0"/>
      <xs:element name="usesService" type=" wsmlIRI "
minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="name" type=" wsmlIRI " use="optional"/>
  </xs:complexType>
</xs:element>

```

[top](#)

## Element: ggMediator

<b>Name</b>	ggMediator
<b>Type</b>	Locally-defined complex type
<b>Nilable</b>	no
<b>Abstract</b>	no
<b>Documentation</b>	A ggMediator specification.

### XML Instance Representation

```

<ggMediator
  name=" wsmlIRI [0..1]">
  <nonFunctionalProperties>... </nonFunctionalProperties> [0..1]
  <importsOntology> ... </importsOntology> [0..*]
  <usesMediator> ... </usesMediator> [0..*]
  <source> wsmlIRI </source> [1..*]
  <target> wsmlIRI </target> [0..1]
  <usesService> wsmlIRI </usesService> [0..1]
</ggMediator>

```

### Schema Component Representation

```

<xs:element name="ggMediator">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref=" nonFunctionalProperties " minOccurs="0"/>
      <xs:element ref=" importsOntology " minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element ref=" usesMediator " minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element name="source" type=" wsmlIRI " minOccurs="1"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

```

```

    <xs:element name="target" type=" wsmlIRI " minOccurs="0"/>
    <xs:element name="usesService" type=" wsmlIRI "
      minOccurs="0"/>
  </xs:sequence>
  <xs:attribute name="name" type=" wsmlIRI " use="optional"/>
</xs:complexType>
</xs:element>

```

[top](#)

## Element: **wgMediator**

<b>Name</b>	wgMediator
<b>Type</b>	Locally-defined complex type
<b>Nilable</b>	no
<b>Abstract</b>	no
<b>Documentation</b>	A wgMediator specification.

### XML Instance Representation

```

<wgMediator
  name=" wsmlIRI [0..1]">
  Start Sequence [0..1]
    <nonFunctionalProperties> ... </nonFunctionalProperties> [0..1]
    <importsOntology> ... </importsOntology> [0..*]
    <usesMediator> ... </usesMediator> [0..*]
    <source> wsmlIRI </source> [0..1]
    <target> wsmlIRI </target> [0..1]
    <usesService> wsmlIRI </usesService> [0..1]
  End Sequence
</wgMediator>

```

### Schema Component Representation

```

<xs:element name="wgMediator">
  <xs:complexType>
    <xs:sequence minOccurs="0">
      <xs:element ref=" nonFunctionalProperties " minOccurs="0"/>
      <xs:element ref=" importsOntology " minOccurs="0"
        maxOccurs="unbounded"/>
      <xs:element ref=" usesMediator " minOccurs="0"
        maxOccurs="unbounded"/>
      <xs:element name="source" type=" wsmlIRI " minOccurs="0"/>
      <xs:element name="target" type=" wsmlIRI " minOccurs="0"/>
      <xs:element name="usesService" type=" wsmlIRI "
        minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="name" type=" wsmlIRI " use="optional"/>
  </xs:complexType>
</xs:element>

```

[top](#)

## Element: **wwMediator**

<b>Name</b>	wwMediator
<b>Type</b>	Locally-defined complex type
<b><u>Nilable</u></b>	no
<b><u>Abstract</u></b>	no
<b>Documentation</b>	A wwMediator specification.

### XML Instance Representation

```

<wwMediator
  name=" wsmlIRI [0..1]">
  <nonFunctionalProperties> ... </nonFunctionalProperties> [0..1]
  <importsOntology> ... </importsOntology> [0..*]
  <usesMediator> ... </usesMediator> [0..*]
  <source> wsmlIRI </source> [0..1]
  <target> wsmlIRI </target> [0..1]
  <usesService> wsmlIRI </usesService> [0..1]
</wwMediator>

```

### Schema Component Representation

```

<xs:element name="wwMediator">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref=" nonFunctionalProperties " minOccurs="0"/>
      <xs:element ref=" importsOntology " minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element ref=" usesMediator " minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element name="source" type=" wsmlIRI " minOccurs="0"/>
      <xs:element name="target" type=" wsmlIRI " minOccurs="0"/>
      <xs:element name="usesService" type=" wsmlIRI "
minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="name" type=" wsmlIRI " use="optional"/>
  </xs:complexType>
</xs:element>

```

[top](#)

## Element: **ontology**

<b>Name</b>	ontology
<b>Type</b>	Locally-defined complex type
<b><u>Nilable</u></b>	no
<b><u>Abstract</u></b>	no
<b>Documentation</b>	An ontology specification.

### XML Instance Representation

```

<ontology
  name=" wsmlIRI [0..1]">
  Start Choice [0..*]
  <nonFunctionalProperties> ... </nonFunctionalProperties> [1]
  <importsOntology> ... </importsOntology> [1]
  <usesMediator> ... </usesMediator> [1]
  <concept> ... </concept> [1]
  <relation> ... </relation> [1]
  <instance> ... </instance> [1]

```

```

    <relationInstance> ... </relationInstance> [1]
    <axiom> ... </axiom> [1]
  End Choice
</ontology>

```

### Schema Component Representation

```

<xs:element name="ontology">
  <xs:complexType>
    <xs:sequence>
      <xs:choice minOccurs="0" maxOccurs="unbounded">
        <xs:element ref="nonFunctionalProperties" />
        <xs:element ref="importsOntology" />
        <xs:element ref="usesMediator" />
        <xs:element ref="concept" />
        <xs:element ref="relation" />
        <xs:element ref="instance" />
        <xs:element ref="relationInstance" />
        <xs:element ref="axiom" />
      </xs:choice>
    </xs:sequence>
    <xs:attribute name="name" type="wsmlIRI" use="optional" />
  </xs:complexType>
</xs:element>

```

[top](#)

### Element: axiom

<b>Name</b>	axiom
<b>Type</b>	<a href="#">axiomType</a>
<b>Nilable</b>	no
<b>Abstract</b>	no

### XML Instance Representation

```

<axiom
  name=" wsmlIRI [0..1]">
  Start Sequence [0..1]
    <nonFunctionalProperties> ... </nonFunctionalProperties> [0..1]
    <definedBy> logicalExpressionType </definedBy> [1]
  End Sequence
</axiom>

```

### Schema Component Representation

```

<xs:element name="axiom" type="axiomType" />

```

[top](#)

### Complex Type: axiomType

Super-types: None

Sub-types: None

<b>Name</b>	axiomType
<b>Abstract</b>	no
<b>Documentation</b>	An axiom reference or specification. In case no elements are nested inside the axiom, it is merely a reference to an axiom definition. Otherwise, it is an axiom specification, consisting of a number of annotations in the form of non-functional properties and the actual logical expression, contained in the <code>definedBy</code> keyword.

### XML Instance Representation

```
<...
  name=" wsmlIRI [0..1]">
  Start Sequence [0..1]
    <nonFunctionalProperties> ... </nonFunctionalProperties> [0..1]
    <definedBy> logicalExpressionType </definedBy> [1]
  End Sequence
</...>
```

### Schema Component Representation

```
<xs:complexType name="axiomType">
  <xs:sequence minOccurs="0">
    <xs:element ref=" nonFunctionalProperties " minOccurs="0"/>
    <xs:element name="definedBy" type=" logicalExpressionType "/>
  </xs:sequence>
  <xs:attribute name="name" type=" wsmlIRI " use="optional"/>
</xs:complexType>
```

[top](#)

## Element: **concept**

<b>Name</b>	concept
<b>Type</b>	Locally-defined complex type
<b>Nilable</b>	no
<b>Abstract</b>	no
<b>Documentation</b>	A concept specification or reference. In case no elements are nested inside the concept, it is a reference to the concept identified by the name. Otherwise, it is a concept specification.

### XML Instance Representation

```
<concept
  name=" wsmlIRI [1]">
  <superConcept> wsmlIRI </superConcept> [0..*]
  <nonFunctionalProperties> ... </nonFunctionalProperties> [0..1]
  <attribute> ... </attribute> [0..*]
</concept>
```

### Schema Component Representation

```
<xs:element name="concept">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="superConcept" type=" wsmlIRI " minOccurs="0"
        maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

```

<xs:element ref=" nonFunctionalProperties " minOccurs="0"/>
<xs:element ref=" attribute " minOccurs="0"
maxOccurs="unbounded"/>
</xs:sequence>
<xs:attribute name="name" type=" wsmlIRI " use="required"/>
</xs:complexType>
</xs:element>

```

[top](#)

## Element: **attribute**

<b>Name</b>	attribute
<b>Type</b>	Locally-defined complex type
<b>Nilable</b>	no
<b>Abstract</b>	no
<b>Documentation</b>	A concept can have zero or more attributes. The type of the attribute can be inferring or constraining. A constraining attribute (corresponding to 'ofType') is used to check the type of parameter values; an inferring parameter (corresponding to 'impliesType') is used to derive the type of attribute values. By default an attribute is constraining. Furthermore, an attribute can be symmetric, transitive, reflexive or the inverse of another attribute. Finally, it is possible to specify the minimal and maximal cardinality of an attribute. By default the minimal cardinality is 0 and the maximal cardinality is n (i.e. not restricted).

### XML Instance Representation

```

<attribute
  name=" wsmlIRI [1]"
  type=" xs:string (value comes from list:
  {'constraining'|'inferring'}) [0..1]">
  <range> wsmlIRI </range> [0..1]
  <symmetric> ... </symmetric> [0..1]
  <transitive> ... </transitive> [0..1]
  <reflexive> ... </reflexive> [0..1]
  <inverseOf> wsmlID </inverseOf> [0..1]
  <minCardinality> xs:integer </minCardinality> [0..1]
  <maxCardinality> xs:integer </maxCardinality> [0..1]
  <nonFunctionalProperties>... </nonFunctionalProperties> [0..1]
</attribute>

```

### Schema Component Representation

```

<xs:element name="attribute">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="range" type=" wsmlIRI " minOccurs="0"/>
      <xs:element name="symmetric" minOccurs="0"/>
      <xs:element name="transitive" minOccurs="0"/>
      <xs:element name="reflexive" minOccurs="0"/>
      <xs:element name="inverseOf" type=" wsmlID " minOccurs="0"/>
      <xs:element name="minCardinality" type=" xs:integer "
minOccurs="0"/>
      <xs:element name="maxCardinality" type=" xs:integer "
minOccurs="0"/>
      <xs:element ref=" nonFunctionalProperties " minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="name" type=" wsmlIRI " use="required"/>
  </xs:complexType>
</xs:element>

```

```

<xs:attribute name="type" default="constraining">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:enumeration value="constraining"/>
      <xs:enumeration value="inferring"/>
    </xs:restriction>
  </xs:simpleType>
</xs:attribute>
</xs:complexType>
</xs:element>

```

[top](#)

## Element: **relation**

<b>Name</b>	relation
<b>Type</b>	<a href="#">relationType</a>
<b>Nilable</b>	no
<b>Abstract</b>	no

### XML Instance Representation

```

<relation
  name=" wsmlIRI [1]"
  arity=" xs:integer [0..1]">
  <superRelation> wsmlIRI </superRelation> [0..*]
  <parameters> ... </parameters> [0..1]
  <nonFunctionalProperties> ... </nonFunctionalProperties> [0..1]
</relation>

```

### Schema Component Representation

```

<xs:element name="relation" type=" relationType "/>

```

[top](#)

## Complex Type: **relationType**

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

<b>Name</b>	relationType
<b>Abstract</b>	no
<b>Documentation</b>	A relation specification or reference. In case no elements are nested inside the relation, it is a reference to the relation identified by the name. Otherwise, it is a relation specification.

### XML Instance Representation

```

<...
  name=" wsmlIRI [1]"
  arity=" xs:integer [0..1]">
  <superRelation> wsmlIRI </superRelation> [0..*]
  <parameters> ... </parameters> [0..1]

```

```

</nonFunctionalProperties> ... </nonFunctionalProperties> [0..1]
</...>

```

### Schema Component Representation

```

<xs:complexType name="relationType">
  <xs:sequence>
    <xs:element name="superRelation" type=" wsmlIRI " minOccurs="0"
      maxOccurs="unbounded"/>
    <xs:element ref=" parameters " minOccurs="0"/>
    <xs:element ref=" nonFunctionalProperties " minOccurs="0"/>
  </xs:sequence>
  <xs:attribute name="name" type=" wsmlIRI " use="required"/>
  <xs:attribute name="arity" type=" xs:integer " use="optional"/>
</xs:complexType>

```

[top](#)

## Element: **parameters**

<b>Name</b>	parameters
<b>Type</b>	Locally-defined complex type
<b>Nilable</b>	no
<b>Abstract</b>	no
<b>Documentation</b>	The ordered list of parameter types

### XML Instance Representation

```

<parameters>
  <parameter
    type=" xs:string (value comes from list:
    {'constraining'|'inferring'}) [0..1]"> [0..*]
    <range> wsmlIRI </range> [1..*]
  </parameter>
</parameters>

```

### Schema Component Representation

```

<xs:element name="parameters">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="parameter" minOccurs="0"
        maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="range" type=" wsmlIRI "
              maxOccurs="unbounded"/>
          </xs:sequence>
          <xs:attribute name="type" default="constraining">
            <xs:simpleType>
              <xs:restriction base=" xs:string ">
                <xs:enumeration value="constraining"/>
                <xs:enumeration value="inferring"/>
              </xs:restriction>
            </xs:simpleType>
          </xs:attribute>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>

```

## Element: **instance**

<b>Name</b>	instance
<b>Type</b>	Locally-defined complex type
<b><u>Nilable</u></b>	no
<b><u>Abstract</u></b>	no
<b>Documentation</b>	An instance can be member of zero or more concepts. Furthermore, zero or more attribute values can be associated with this particular instance.

### XML Instance Representation

```
<instance
  name=" wsmlIRI [0..1]">
  <memberOf> wsmlIRI </memberOf> [0..*]
  <nonFunctionalProperties> ... </nonFunctionalProperties> [0..1]
  <attributeValue> ... </attributeValue> [0..*]
</instance>
```

### Schema Component Representation

```
<xs:element name="instance">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="memberOf" type=" wsmlIRI " minOccurs="0"
        maxOccurs="unbounded"/>
      <xs:element ref=" nonFunctionalProperties " minOccurs="0"/>
      <xs:element ref=" attributeValue " minOccurs="0"
        maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="name" type=" wsmlIRI " use="optional"/>
  </xs:complexType>
</xs:element>
```

## Element: **attributeValue**

<b>Name</b>	attributeValue
<b>Type</b>	Locally-defined complex type
<b><u>Nilable</u></b>	no
<b><u>Abstract</u></b>	no
<b>Documentation</b>	The value of an attribute is either the identifier of another WSML element or a literal.

### XML Instance Representation

```
<attributeValue
  name=" wsmlIRI [1]">
  <value> wsmlAnyValue </value> [1..*]
</attributeValue>
```

### Schema Component Representation

```

<xs:element name="attributeValue">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="value" type=" wsmlAnyValue "
        maxOccurs="unbounded"/>
      <!-- can be instance, ID or literal -->
    </xs:sequence>
    <xs:attribute name="name" type=" wsmlIRI " use="required"/>
  </xs:complexType>
</xs:element>

```

[top](#)

## Element: **relationInstance**

<b>Name</b>	relationInstance
<b>Type</b>	Locally-defined complex type
<b><u>Nilable</u></b>	no
<b><u>Abstract</u></b>	no

### XML Instance Representation

```

<relationInstance
  name=" wsmlIRI [0..1]">
  <memberOf> wsmlIRI </memberOf> [0..*]
  <nonFunctionalProperties> ... </nonFunctionalProperties> [0..1]
  <parameterValue> ... </parameterValue> [0..*]
</relationInstance>

```

### Schema Component Representation

```

<xs:element name="relationInstance">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="memberOf" type=" wsmlIRI " minOccurs="0"
        maxOccurs="unbounded"/>
      <xs:element ref=" nonFunctionalProperties " minOccurs="0"/>
      <xs:element ref=" parameterValue " minOccurs="0"
        maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="name" type=" wsmlIRI " use="optional"/>
  </xs:complexType>
</xs:element>

```

[top](#)

## Element: **parameterValue**

<b>Name</b>	parameterValue
<b>Type</b>	Locally-defined complex type
<b><u>Nilable</u></b>	no
<b><u>Abstract</u></b>	no

### XML Instance Representation

```

<parameterValue>
  Start Choice [1]

```

```
<value> wsmlAnyValue </value> [1..*]  
End Choice  
</parameterValue>
```

### Schema Component Representation

```
<xs:element name="parameterValue">  
  <xs:complexType>  
    <xs:choice>  
      <xs:element name="value" type=" wsmlAnyValue "  
        maxOccurs="unbounded"/>  
    </xs:choice>  
  </xs:complexType>  
</xs:element>
```

[top](#)

Generated by [xs3p](#).