

Schema documentation for config.xsd

june 27, 2011

Table of Contents

Namespace: ""	1
Schema(s)	1
Main schema config.xsd	1
Element(s)	1
Element jmxpoller	1
Element formatter	2
Element cluster	2
Element mbean	3
Element operation	4
Element parameter	5
Element attribute	6
Element jmxserver	6
Attribute(s)	8
Attribute formatter / @className	8
Attribute parameter / @value	8
Attribute parameter / @type	9
Attribute operation / @name	9
Attribute operation / @outputname	9
Attribute attribute / @name	10
Attribute attribute / @outputname	10
Attribute mbean / @domain	10
Attribute mbean / @properties	10
Attribute jmxserver / @host	11
Attribute jmxserver / @jmxpass	11
Attribute jmxserver / @jmxport	11
Attribute jmxserver / @jmxuser	11
Attribute jmxserver / @jvmDescription	12
Attribute jmxserver / @pid	12
Attribute jmxserver / @pidFile	12
Attribute jmxserver / @pidCommand	12
Attribute cluster / @name	13
Attribute cluster / @description	13

Namespace: ""

Schema(s)

Main schema config.xsd

Namespace	No namespace
Properties	attribute form default: unqualified element form default: qualified

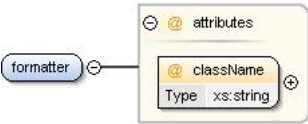
Element(s)

Element jmxpoller

Namespace	No namespace
Annotations	Root element of the configuration file. This configuration file is where you specify local and remote JMX servers to connect to across your enterprise and extract whatever MBean attributes you have declared to query. The result will then be written to STDOUT for SPLUNK indexing.
Diagram	<pre> graph LR jmxpoller --> formatter jmxpoller --> cluster jmxpoller --> jmxserver style jmxpoller fill:#d9e1f2,stroke:#333,stroke-width:1px style formatter fill:#d9e1f2,stroke:#333,stroke-width:1px style cluster fill:#d9e1f2,stroke:#333,stroke-width:1px style jmxserver fill:#d9e1f2,stroke:#333,stroke-width:1px </pre>

Properties	content: complex
Model	formatter{0,1} , cluster* , jmxserver*
Children	cluster, formatter, jmxserver
Instance	<pre><jmxpoller> <formatter className="">{0,1}</formatter> <cluster description="" name="">{0,unbounded}</cluster> <jmxserver host="" jmxpass="" jmxport="" jmxuser="" jvmDescription="" pid="" pidCommand="" pidFile=""> jmxserver> </jmxpoller></pre>
Source	<pre><xs:element name="jmxpoller"> <xs:annotation> <xs:documentation>Root element of the configuration file. This configuration file is where you specify local and remote JMX servers to connect to across your enterprise and extract whatever MBean attributes you have declared to query. The result will then be written to STDOUT for SPLUNK indexing.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element minOccurs="0" maxOccurs="1" ref="formatter"/> <xs:element minOccurs="0" maxOccurs="unbounded" ref="cluster"/> <xs:element minOccurs="0" maxOccurs="unbounded" ref="jmxserver"/> </xs:sequence> </xs:complexType> </xs:element></pre>

Element formatter

Namespace	No namespace				
Annotations	Custom formatter declaration allows you to override the default STDOUT format				
Diagram					
Properties	content: complex				
Used by	Element	jmxpoller			
Attributes	QName	Type	Fixed	Default	Use
	className	xs:string			required
	Fully qualified Java class name of the formatter implementation, implements the com.dtdsoftware.splunk.formatter.Formatter interface				
Source	<pre><xs:element name="formatter"> <xs:annotation> <xs:documentation>Custom formatter declaration allows you to override the default STDOUT format</xs:documentation> </xs:annotation> <xs:complexType> <xs:attribute name="className" type="xs:string" use="required"> <xs:annotation> <xs:documentation>Fully qualified Java class name of the formatter implementation, implements the com.dtdsoftware.splunk.formatter.Formatter interface</xs:documentation> </xs:annotation> </xs:attribute> </xs:complexType> </xs:element></pre>				

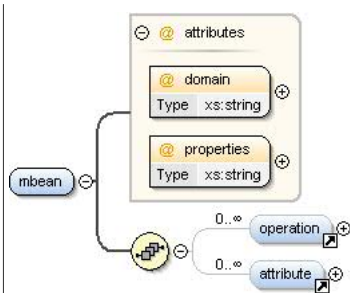
Element cluster

Namespace	No namespace
Annotations	For JVMs with the same MBeans, you can group them under this element so you only have to declare the common beans to query once. You can still declare additional mbeans specific to each jmxserver within the jmxserver elements.

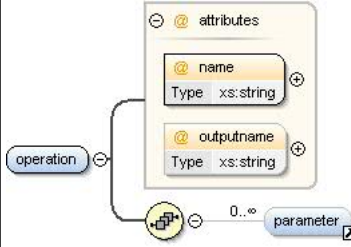
Diagram					
Properties	content:	complex			
Used by	Element	jmxpoller			
Model	mbean+ , jmxserver+				
Children	jmxserver, mbean				
Instance	<pre><cluster description="" name=""> <mbean domain="" properties="">{1,unbounded}</mbean> <jmxserver host="" jmxpass="" jmxport="" jmxuser="" jvmDescription="" pid="" pidCommand="" jmxserver> </cluster></pre>				
Attributes	QName	Type	Fixed	Default	Use
	description	xs:string			optional
		Description of this cluster			
	name	xs:string			optional
		Name for this cluster			
Source	<pre><xs:element name="cluster"> <xs:annotation> <xs:documentation>For JVMs with the same MBeans, you can group them under this element so you only have to declare the common beans to query once. You can still declare additional mbeans specfic to each jmxserver within the jmxserver elements.</ xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element maxOccurs="unbounded" ref="mbean"/> <xs:element maxOccurs="unbounded" ref="jmxserver"/> </xs:sequence> <xs:attribute name="name" type="xs:string"> <xs:annotation> <xs:documentation>Name for this cluster</xs:documentation> </xs:annotation> </xs:attribute> <xs:attribute name="description" type="xs:string"> <xs:annotation> <xs:documentation>Description of this cluster</xs:documentation> </xs:annotation> </xs:attribute> </xs:complexType> </xs:element></pre>				

Element mbean

Namespace	No namespace
Annotations	<p>An MBean to query</p> <p>Standard JMX object name wildcard patterns * and ? are supported</p> <p>If no values are specified for the "domain" and "properties" attributes , the value will default to the * wildcard</p>

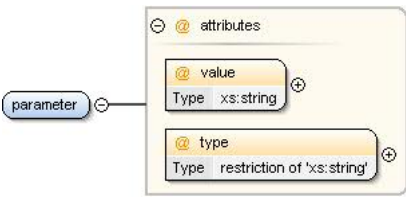
Diagram					
Properties	content:	complex			
Used by	Elements	cluster, jmxserver			
Model	operation*, attribute*				
Children	attribute, operation				
Instance	<pre><mbean domain=" " properties=" "> <operation name=" " outputname=" ">{0,unbounded}</operation> <attribute name=" " outputname=" ">{0,unbounded}</attribute> </mbean></pre>				
Attributes	QName	Type	Fixed	Default	Use
	domain	xs:string			required
		The MBean domain			
	properties	xs:string			required
		The MBean properties string in "key=value,key2=value2" format			
Source	<pre><xs:element name="mbean"> <xs:annotation> <xs:documentation>An MBean to query Standard JMX object name wildcard patterns * and ? are supported If no values are specified for the "domain" and "properties" attributes , the value will default to the * wildcard</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element minOccurs="0" maxOccurs="unbounded" ref="operation"/> <xs:element minOccurs="0" maxOccurs="unbounded" ref="attribute"/> </xs:sequence> <xs:attribute name="domain" use="required" type="xs:string"> <xs:annotation> <xs:documentation>The MBean domain</xs:documentation> </xs:annotation> </xs:attribute> <xs:attribute name="properties" use="required" type="xs:string"> <xs:annotation> <xs:documentation>The MBean properties string in "key=value,key2=value2" format</ xs:documentation> </xs:annotation> </xs:attribute> </xs:complexType> </xs:element></pre>				

Element operation

Namespace	No namespace
Annotations	An MBean operation
Diagram	
Properties	content: complex
Used by	Element mbean

Model	parameter*				
Children	parameter				
Instance	<pre><operation name=" " outputname=" "> <parameter type=" " value=" ">{0,unbounded}</parameter> </operation></pre>				
Attributes	QName	Type	Fixed	Default	Use
	name	xs:string			required
		The operation name. For overloaded operations, the operation signature is inferred from the paramaters list.			
	outputname	xs:string			optional
		The operation result key that is output to STDOUT for SPLUNK indexing.Optional, some operations may not return values.			
Source	<pre><xs:element name="operation"> <xs:annotation> <xs:documentation>An MBean operation</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element minOccurs="0" maxOccurs="unbounded" ref="parameter"/> </xs:sequence> <xs:attribute name="name" use="required" type="xs:string"> <xs:annotation> <xs:documentation>The operation name. For overloaded operations, the operation signature is inferred from the paramaters list.</xs:documentation> </xs:annotation> </xs:attribute> <xs:attribute name="outputname" type="xs:string"> <xs:annotation> <xs:documentation>The operation result key that is output to STDOUT for SPLUNK indexing.Optional, some operations may not return values.</xs:documentation> </xs:annotation> </xs:attribute> </xs:complexType> </xs:element></pre>				

Element parameter

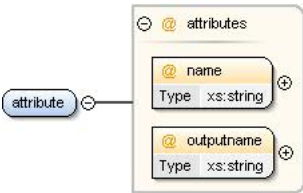
Namespace	No namespace				
Annotations	An MBean operation parameter				
Diagram					
Properties	content:	complex			
Used by	Element	operation			
Attributes	QName	Type	Fixed	Default	Use
	type	restriction of xs:string			required
		The parameter type			
	value	xs:string			required
		The parameter value			
Source	<pre><xs:element name="parameter"> <xs:annotation> <xs:documentation>An MBean operation parameter</xs:documentation> </xs:annotation> <xs:complexType> <xs:attribute name="value" use="required" type="xs:string"> <xs:annotation> <xs:documentation>The parameter value</xs:documentation> </xs:annotation> </xs:attribute> <xs:attribute name="type" use="required"> <xs:annotation></pre>				

```

<xs:documentation>The parameter type</xs:documentation>
</xs:annotation>
<xs:simpleType>
  <xs:restriction base="xs:string">
    <xs:enumeration value="int"/>
    <xs:enumeration value="byte"/>
    <xs:enumeration value="short"/>
    <xs:enumeration value="long"/>
    <xs:enumeration value="float"/>
    <xs:enumeration value="double"/>
    <xs:enumeration value="boolean"/>
    <xs:enumeration value="char"/>
    <xs:enumeration value="string"/>
  </xs:restriction>
</xs:simpleType>
</xs:attribute>
</xs:complexType>
</xs:element>

```

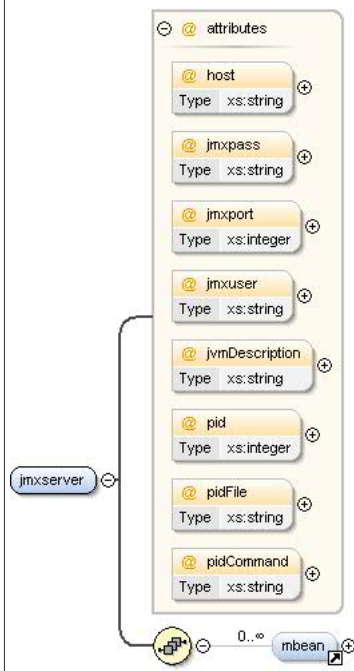
Element attribute

Namespace	No namespace				
Annotations	An MBean attribute				
Diagram					
Properties	content:	complex			
Used by	Element	mbean			
Attributes	QName	Type	Fixed	Default	Use
	name	xs:string			required
		The attribute name For attributes that are multi level ie: composite and tabular attributes , then you can use a ":" delimited notation for specifying the attribute name. ie: foo:goo:myattribute			
	outputname	xs:string			required
		The attribute key that is output to STDOUT for SPLUNK indexing			
Source	<pre><xs:element name="attribute"> <xs:annotation> <xs:documentation>An MBean attribute</xs:documentation> </xs:annotation> <xs:complexType> <xs:attribute name="name" use="required" type="xs:string"> <xs:annotation> <xs:documentation>The attribute name For attributes that are multi level ie: composite and tabular attributes , then you can use a ":" delimited notation for specifying the attribute name. ie: foo:goo:myattribute</xs:documentation> </xs:annotation> </xs:attribute> <xs:attribute name="outputname" use="required" type="xs:string"> <xs:annotation> <xs:documentation>The attribute key that is output to STDOUT for SPLUNK indexing</ xs:documentation> </xs:annotation> </xs:attribute> </xs:complexType> </xs:element></pre>				

Element jmxserver

Namespace	No namespace
Annotations	A local or remote JMX Server to connect to

Diagram



Properties

content: complex

Used by

Elements cluster, jmxpoller

Model

mbean*

Children

mbean

Instance

```

<jmxserver host="" jmxpass="" jmxport="" jmxuser="" jvmDescription="" pid="" pidCommand="" pidFile="">
  <mbean domain="" properties="">{0,unbounded}</mbean>
</jmxserver>
  
```

Attributes

QName	Type	Fixed	Default	Use
host	xs:string			optional
	IP Address, Hostname or DNS Alias.			
jmxpass	xs:string			optional
	JMX Password			
jmxport	xs:integer			optional
	JMX Port			
jmxuser	xs:string			optional
	JMX Username			
jvmDescription	xs:string			optional
	A description of this JVM			
pid	xs:integer			optional
	Process ID for attaching directly to a locally running JVM			
pidCommand	xs:string			optional
	Command/Script that outputs to STDOUT the Process ID for attaching directly to a locally running JVM			
pidFile	xs:string			optional
	File containing the Process ID for attaching directly to a locally running JVM. The only file contents should be the PID on the first line of the file.			

Source

```

<xs:element name="jmxserver">
  <xs:annotation>
    <xs:documentation>A local or remote JMX Server to connect to</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" ref="mbean"/>
    </xs:sequence>
    <xs:attribute name="host" type="xs:string">
  
```

```

<xs:annotation>
  <xs:documentation>IP Address, Hostname or DNS Alias.</xs:documentation>
</xs:annotation>
</xs:attribute>
<xs:attribute name="jmxpass" type="xs:string">
  <xs:annotation>
    <xs:documentation>JMX Password</xs:documentation>
  </xs:annotation>
</xs:attribute>
<xs:attribute name="jmxport" type="xs:integer">
  <xs:annotation>
    <xs:documentation>JMX Port</xs:documentation>
  </xs:annotation>
</xs:attribute>
<xs:attribute name="jmxuser" type="xs:string">
  <xs:annotation>
    <xs:documentation>JMX Username</xs:documentation>
  </xs:annotation>
</xs:attribute>
<xs:attribute name="jvmDescription" type="xs:string">
  <xs:annotation>
    <xs:documentation>A description of this JVM</xs:documentation>
  </xs:annotation>
</xs:attribute>
<xs:attribute name="pid" type="xs:integer">
  <xs:annotation>
    <xs:documentation>Process ID for attaching directly to a locally running JVM</xs:documentation>
  </xs:annotation>
</xs:attribute>
<xs:attribute name="pidFile" type="xs:string">
  <xs:annotation>
    <xs:documentation>File containing the Process ID for attaching directly to a
    locally running JVM.The only file contents should be the PID on the first line of the
    file.</xs:documentation>
  </xs:annotation>
</xs:attribute>
<xs:attribute name="pidCommand" type="xs:string">
  <xs:annotation>
    <xs:documentation>Command/Script that outputs to STDOUT the Process ID for
    attaching directly to a locally running JVM</xs:documentation>
  </xs:annotation>
</xs:attribute>
</xs:complexType>
</xs:element>

```

Attribute(s)

Attribute formatter / @className

Namespace	No namespace
Annotations	Fully qualified Java class name of the formatter implementation, implements the com.dtdsoftware.splunk.formatter.Formatter interface
Type	xs:string
Properties	use: required
Used by	Element formatter
Source	<pre> <xs:attribute name="className" type="xs:string" use="required"> <xs:annotation> <xs:documentation>Fully qualified Java class name of the formatter implementation, implements the com.dtdsoftware.splunk.formatter.Formatter interface</xs:documentation> </xs:annotation> </xs:attribute> </pre>

Attribute parameter / @value

Namespace	No namespace
Annotations	The parameter value
Type	xs:string
Properties	use: required
Used by	Element parameter
Source	<pre> <xs:attribute name="value" use="required" type="xs:string"> </pre>


```

<xs:annotation>
  <xs:documentation>The parameter value</xs:documentation>
</xs:annotation>
</xs:attribute>

```

Attribute parameter / @type

Namespace	No namespace																		
Annotations	The parameter type																		
Type	restriction of xs:string																		
Properties	use: required																		
Facets	<table> <tr><td>enumeration</td><td>int</td></tr> <tr><td>enumeration</td><td>byte</td></tr> <tr><td>enumeration</td><td>short</td></tr> <tr><td>enumeration</td><td>long</td></tr> <tr><td>enumeration</td><td>float</td></tr> <tr><td>enumeration</td><td>double</td></tr> <tr><td>enumeration</td><td>boolean</td></tr> <tr><td>enumeration</td><td>char</td></tr> <tr><td>enumeration</td><td>string</td></tr> </table>	enumeration	int	enumeration	byte	enumeration	short	enumeration	long	enumeration	float	enumeration	double	enumeration	boolean	enumeration	char	enumeration	string
enumeration	int																		
enumeration	byte																		
enumeration	short																		
enumeration	long																		
enumeration	float																		
enumeration	double																		
enumeration	boolean																		
enumeration	char																		
enumeration	string																		
Used by	Element parameter																		
Source	<pre> <xs:attribute name="type" use="required"> <xs:annotation> <xs:documentation>The parameter type</xs:documentation> </xs:annotation> <xs:simpleType> <xs:restriction base="xs:string"> <xs:enumeration value="int"/> <xs:enumeration value="byte"/> <xs:enumeration value="short"/> <xs:enumeration value="long"/> <xs:enumeration value="float"/> <xs:enumeration value="double"/> <xs:enumeration value="boolean"/> <xs:enumeration value="char"/> <xs:enumeration value="string"/> </xs:restriction> </xs:simpleType> </xs:attribute> </pre>																		

Attribute operation / @name

Namespace	No namespace
Annotations	The operation name. For overloaded operations, the operation signature is inferred from the paramaters list.
Type	xs:string
Properties	use: required
Used by	Element operation
Source	<pre> <xs:attribute name="name" use="required" type="xs:string"> <xs:annotation> <xs:documentation>The operation name. For overloaded operations, the operation signature is inferred from the paramaters list.</xs:documentation> </xs:annotation> </xs:attribute> </pre>

Attribute operation / @outputname

Namespace	No namespace
Annotations	The operation result key that is output to STDOUT for SPLUNK indexing.Optional, some operations may not return values.
Type	xs:string
Properties	content: simple

Used by	Element operation
Source	<pre><xs:attribute name="outputname" type="xs:string"> <xs:annotation> <xs:documentation>The operation result key that is output to STDOUT for SPLUNK indexing.Optional, some operations may not return values.</xs:documentation> </xs:annotation> </xs:attribute></pre>

Attribute attribute / @name

Namespace	No namespace
Annotations	<p>The attribute name For attributes that are multi level ie: composite and tabular attributes , then you can use a ":" delimited notation for specifying the attribute name. ie: foo:goo:myattribute</p>
Type	xs:string
Properties	use: required
Used by	Element attribute
Source	<pre><xs:attribute name="name" use="required" type="xs:string"> <xs:annotation> <xs:documentation>The attribute name For attributes that are multi level ie: composite and tabular attributes , then you can use a ":" delimited notation for specifying the attribute name. ie: foo:goo:myattribute</xs:documentation> </xs:annotation> </xs:attribute></pre>

Attribute attribute / @outputname

Namespace	No namespace
Annotations	The attribute key that is output to STDOUT for SPLUNK indexing
Type	xs:string
Properties	use: required
Used by	Element attribute
Source	<pre><xs:attribute name="outputname" use="required" type="xs:string"> <xs:annotation> <xs:documentation>The attribute key that is output to STDOUT for SPLUNK indexing</ </xs:annotation> </xs:attribute></pre>

Attribute mbean / @domain

Namespace	No namespace
Annotations	The MBean domain
Type	xs:string
Properties	use: required
Used by	Element mbean
Source	<pre><xs:attribute name="domain" use="required" type="xs:string"> <xs:annotation> <xs:documentation>The MBean domain</xs:documentation> </xs:annotation> </xs:attribute></pre>

Attribute mbean / @properties

Namespace	No namespace
Annotations	The MBean properties string in "key=value,key2=value2" format
Type	xs:string

Properties	use: required
Used by	Element mbean
Source	<pre><xs:attribute name="properties" use="required" type="xs:string"> <xs:annotation> <xs:documentation>The MBean properties string in "key=value,key2=value2" format</ </xs:annotation> </xs:attribute></pre>

Attribute jmxserver / @host

Namespace	No namespace
Annotations	IP Address, Hostname or DNS Alias.
Type	xs:string
Properties	content: simple
Used by	Element jmxserver
Source	<pre><xs:attribute name="host" type="xs:string"> <xs:annotation> <xs:documentation>IP Address, Hostname or DNS Alias.</xs:documentation> </xs:annotation> </xs:attribute></pre>

Attribute jmxserver / @jmxpass

Namespace	No namespace
Annotations	JMX Password
Type	xs:string
Properties	content: simple
Used by	Element jmxserver
Source	<pre><xs:attribute name="jmxpass" type="xs:string"> <xs:annotation> <xs:documentation>JMX Password</xs:documentation> </xs:annotation> </xs:attribute></pre>

Attribute jmxserver / @jmxport

Namespace	No namespace
Annotations	JMX Port
Type	xs:integer
Properties	content: simple
Used by	Element jmxserver
Source	<pre><xs:attribute name="jmxport" type="xs:integer"> <xs:annotation> <xs:documentation>JMX Port</xs:documentation> </xs:annotation> </xs:attribute></pre>

Attribute jmxserver / @jmxuser

Namespace	No namespace
Annotations	JMX Username
Type	xs:string
Properties	content: simple
Used by	Element jmxserver
Source	<pre><xs:attribute name="jmxuser" type="xs:string"> <xs:annotation> <xs:documentation>JMX Username</xs:documentation> </xs:annotation> </xs:attribute></pre>

	</xs:attribute>
--	-----------------

Attribute jmxserver / @jvmDescription

Namespace	No namespace
Annotations	A description of this JVM
Type	xs:string
Properties	content: simple
Used by	Element jmxserver
Source	<pre><xs:attribute name="jvmDescription" type="xs:string"> <xs:annotation> <xs:documentation>A description of this JVM</xs:documentation> </xs:annotation> </xs:attribute></pre>

Attribute jmxserver / @pid

Namespace	No namespace
Annotations	Process ID for attaching directly to a locally running JVM
Type	xs:integer
Properties	content: simple
Used by	Element jmxserver
Source	<pre><xs:attribute name="pid" type="xs:integer"> <xs:annotation> <xs:documentation>Process ID for attaching directly to a locally running JVM</ xs:documentation> </xs:annotation> </xs:attribute></pre>

Attribute jmxserver / @pidFile

Namespace	No namespace
Annotations	File containing the Process ID for attaching directly to a locally running JVM.The only file contents should be the PID on the first line of the file.
Type	xs:string
Properties	content: simple
Used by	Element jmxserver
Source	<pre><xs:attribute name="pidFile" type="xs:string"> <xs:annotation> <xs:documentation>File containing the Process ID for attaching directly to a locally running JVM.The only file contents should be the PID on the first line of the file.</ xs:documentation> </xs:annotation> </xs:attribute></pre>

Attribute jmxserver / @pidCommand

Namespace	No namespace
Annotations	Command/Script that outputs to STDOUT the Process ID for attaching directly to a locally running JVM
Type	xs:string
Properties	content: simple
Used by	Element jmxserver
Source	<pre><xs:attribute name="pidCommand" type="xs:string"> <xs:annotation> <xs:documentation>Command/Script that outputs to STDOUT the Process ID for attaching directly to a locally running JVM</xs:documentation> </xs:annotation> </xs:attribute></pre>

Attribute cluster / @name

Namespace	No namespace
Annotations	Name for this cluster
Type	xs:string
Properties	content: simple
Used by	Element cluster
Source	<pre><xs:attribute name="name" type="xs:string"> <xs:annotation> <xs:documentation>Name for this cluster</xs:documentation> </xs:annotation> </xs:attribute></pre>

Attribute cluster / @description

Namespace	No namespace
Annotations	Description of this cluster
Type	xs:string
Properties	content: simple
Used by	Element cluster
Source	<pre><xs:attribute name="description" type="xs:string"> <xs:annotation> <xs:documentation>Description of this cluster</xs:documentation> </xs:annotation> </xs:attribute></pre>