Schema documentation for event_schema.xsd

august 31, 2016

Table of Contents

Namespace: "" ... 2
- Schema(s) ... 2
  - Main schema event_schema.xsd ... 2
  - Included schema common_elements.xsd ... 2
  - Included schema common_types.xsd ... 2
- Element(s) ... 2
  - Element events ... 2
  - Element event ... 3
  - Element comment ... 4
  - Element args ... 4
  - Element external_arg_define / arg ... 5
  - Element enum ... 6
  - Element item ... 6
  - Element arg_define / arg ... 7
  - Element return ... 8
- Simple Type(s) ... 9
  - Simple Type id_define ... 9
  - Simple Type severity_define ... 10
  - Simple Type base_code_define ... 10
  - Simple Type full_items_define ... 10
  - Simple Type pass_by_define ... 11
  - Simple Type component_role_define ... 11
  - Simple Type channel_update_define ... 12
  - Simple Type command_kind_define ... 12
  - Simple Type component_types_define ... 12
  - Simple Type port_types_define ... 13
  - Simple Type id_or_system_var_define ... 13
  - Simple Type system_var_define ... 14
  - Simple Type positive_integer_define ... 14
  - Simple Type int8_t_define ... 14
  - Simple Type uint8_t_define ... 15
  - Simple Type int16_t_define ... 15
  - Simple Type uint16_t_define ... 15
  - Simple Type int32_t_define ... 16
  - Simple Type uint32_t_define ... 16
  - Simple Type int64_t_define ... 16
  - Simple Type uint64_t_define ... 17
  - Simple Type not_user_cpp_type_define ... 17
  - Simple Type NATIVE_INT_TYPE_define ... 17
  - Simple Type NATIVE_UINT_TYPE_define ... 18
  - Simple Type I8_define ... 18
  - Simple Type U8_define ... 18
  - Simple Type BYTE_define ... 19
  - Simple Type I16_define ... 19
  - Simple Type U16_define ... 19
  - Simple Type I32_define ... 20
  - Simple Type U32_define ... 20
  - Simple Type I64_define ... 21
  - Simple Type U64_define ... 21
  - Simple Type F32_define ... 21
  - Simple Type F64_define ... 21
- Attribute(s) ... 22
  - Attribute item / @name ... 22
  - Attribute item / @value ... 22
  - Attribute item / @comment ... 22
  - Attribute enum / @name ... 22
  - Attribute external_arg_define / arg / @name ... 22
  - Attribute external_arg_define / arg / @comment ... 23
  - Attribute type_size_choice_define / @data_type ... 23
  - Attribute type_size_choice_define / @type ... 23

Attribute type_size_choice_define / @size 24
Attribute event / @name 24
Attribute event / @id 24
Attribute event / @severity 24
Attribute event / @format_string 25
Attribute event / @throttle 25
Attribute events / @event_base 25
Attribute arg_define / arg / @name 25
Attribute arg_define / arg / @pass_by 25
Attribute arg_define / arg / @comment 26
Attribute return / @name 26
Attribute return / @pass_by 26
Attribute return / @comment 26
Element Group(s) 27
Element Group external_arg_define 27
Element Group type_size_choice_define 27
Element Group arg_define 27
Attribute Group(s) 28
Attribute Group type_size_choice_define 28

Namespace: ""

Schema(s)

Main schema event_schema.xsd

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

Included schema common_elements.xsd

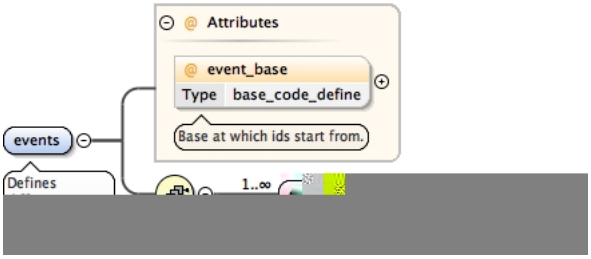
Namespace	No namespace
Properties	attribute form default: unqualified element form default: qualified

Included schema common_types.xsd

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

Element(s)

Element events

Namespace	No namespace
Annotations	Defines different events for a component.
Diagram	
Properties	content: complex
Model	event+

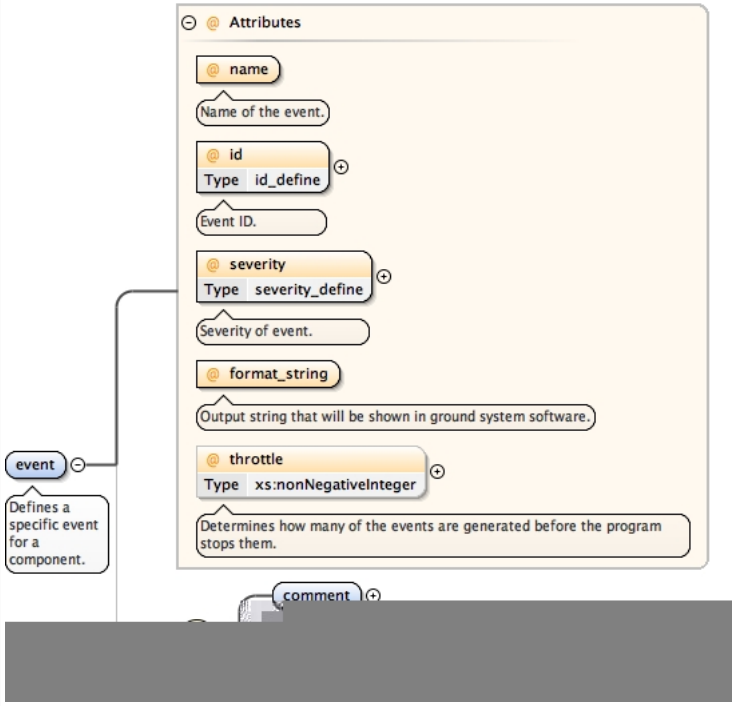
Children	event

Instance	`<events event_base="">`
	`  <event format_string="" id="" name="" severity="" throttle="">{1,unbounded}</event>`
	`</events>`

Attributes	**QName** / **Type** / **Use** **event_base** / base_code_define / optional Base at which ids start from.

Source	`<xs:element name="events">`
	`  <xs:annotation>`
	`  <xs:documentation>Defines different events for a component.</xs:documentation>`
	`  </xs:annotation>`
	`  <xs:complexType>`
	`  <xs:sequence>`
	`  <xs:element maxOccurs="unbounded" ref="event"/>`
	`  </xs:sequence>`
	`  <xs:attribute name="event_base" type="base_code_define">`
	`  <xs:annotation>`
	`  <xs:documentation>Base at which ids start from.</xs:documentation>`
	`  </xs:annotation>`
	`  </xs:attribute>`
	`  </xs:complexType>`
	`</xs:element>`

Element event

Namespace	No namespace
Annotations	Defines a specific event for a component.
Diagram	
Properties	content: complex
Used by	Element events
Model	comment | args
Children	args, comment

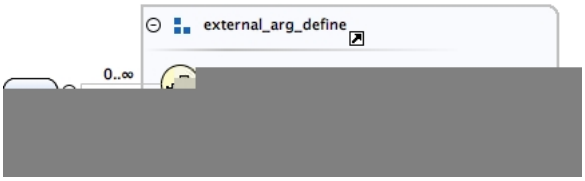
Instance	`<event format_string="" id="" name="" severity="" throttle="">`
	`  <comment>{1,1}</comment>`
	`  <args>{1,1}</args>`
	`</event>`

Attributes	**QName** / **Type** / **Use** **format_string** / / required Output string that will be shown in ground system software.

QName	Type	Use	
id	id_define	required	
	Event ID.		
name		required	
	Name of the event.		
severity	severity_define	required	
	Severity of event.		
throttle	xs:nonNegativeInteger	optional	
	Determines how many of the events are generated before the program stops them.		

Source

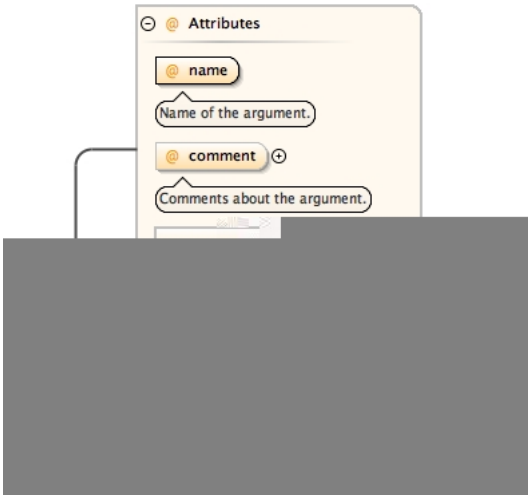
```
<xs:element name="event">
  <xs:annotation>
    <xs:documentation>Defines a specific event for a component.</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:choice minOccurs="0" maxOccurs="unbounded">
      <xs:element ref="comment"/>
      <xs:element ref="args"/>
    </xs:choice>
    <xs:attribute name="name" use="required"
```

Diagram	
Properties	content: `complex`
Used by	Element event
Model	arg
Children	arg

Instance	`<args>`
	`  <arg comment="" data_type="" name="" size="" type="">{1,1}</arg>`
	`</args>`

Source	`<xs:element name="args">`
	`  <xs:complexType>`
	`  <xs:group minOccurs="0" maxOccurs="unbounded" ref="external_arg_define"/>`
	`  </xs:complexType>`
	`</xs:element>`

Element `external_arg_define / arg`

Namespace	No namespace
Diagram	
Properties	content: `complex`
Model	(enum{0,1}) | comment
Children	comment, enum

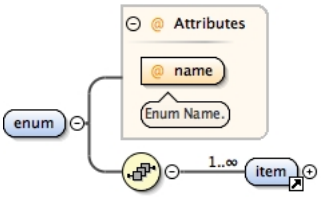
Instance	`<arg comment="" data_type="" name="" size="" type="">`
	`  <enum name="">{0,1}</enum>`
	`  <comment>{1,1}</comment>`
	`</arg>`

Attributes

QName	Type	Use	
comment	xs:string	optional	
	`Comments about the argument.`		
data_type	union of(xs:string, restriction of xs:token, restriction of xs:token)	optional	
name		required	
	`Name of the argument.`		
size	xs:nonNegativeInteger	optional	
	`The size of the argument.`		
type	union of(xs:string, restriction of xs:token, restriction of xs:token)	optional	

Source	<xs:element name="arg"> <xs:complexType> <xs:choice minOccurs="0" maxOccurs="unbounded"> <xs:group ref="type_size_choice_define"/> <xs:element ref="comment"/> </xs:choice> <xs:attribute name="name" use="required"> <xs:annotation> <xs:documentation>Name of the argument.</xs:documentation> </xs:annotation> </xs:attribute> <xs:attribute name="comment" type="xs:string"> <xs:annotation> <xs:documentation>Comments about the argument.</xs:documentation> </xs:annotation> </xs:attribute> <xs:attributeGroup ref="type_size_choice_define"/> </xs:complexType> </xs:element>

Element enum

Namespace	No namespace
Diagram	
Properties	content: complex
Used by	Element Group type_size_choice_define
Model	item+
Children	item
Instance	<enum name=""> <item comment="" name="" value="">{1,unbounded}</item> </enum>
Attributes	**QName** / **Type** / **Use** **name** / / required Enum Name.
Source	<xs:element name="enum"> <xs:complexType> <xs:sequence> <xs:element maxOccurs="unbounded" ref="item"/> </xs:sequence> <xs:attribute name="name" use="required"> <xs:annotation> <xs:documentation>Enum Name.</xs:documentation> </xs:annotation> </xs:attribute> </xs:complexType> </xs:element>

Element item

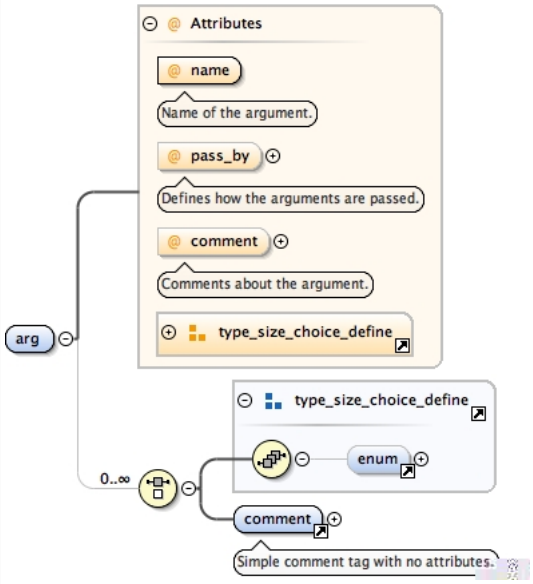
Namespace	No namespace
Diagram	

Properties	content:	complex		
Used by	Element	enum		
Attributes	**QName**	**Type**	**Use**	
	comment		optional	
		Comment about the enum item.		
	name		required	
		Name of the enum item.		
	value		optional	
		The value being sent through the enum item.		

Source

```
<xs:element name="item">
  <xs:complexType>
    <xs:attribute name="name" use="required">
      <xs:annotation>
        <xs:documentation>Name of the enum item.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="value">
      <xs:annotation>
        <xs:documentation>The value being sent through the enum item.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="comment">
      <xs:annotation>
        <xs:documentation>Comment about the enum item.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
</xs:element>
```

Element arg_define / arg

Namespace	No namespace
Diagram	
Properties	content: complex
Model	(enum{0,1}) | comment
Children	comment, enum

Instance

```
<arg comment="" data_type="" name="" pass_by="" size="" type="">
  <enum name="">{0,1}</enum>
  <comment>{1,1}</comment>
</arg>
```

Attributes	**QName**	**Type**	**Use**	
	comment	xs:string	optional	
		Comments about the argument.		

	QName	Type	Use	
	data_type	union of(xs:string, restriction of xs:token, restriction of xs:token)	optional	
	name		required	
		`Name of the argument.`		
	pass_by	pass_by_define	optional	
		`Defines how the arguments are passed.`		
	size	xs:nonNegativeInteger	optional	
		`The size of the argument.`		
	type	union of(xs:string, restriction of xs:token, restriction of xs:token)	optional	

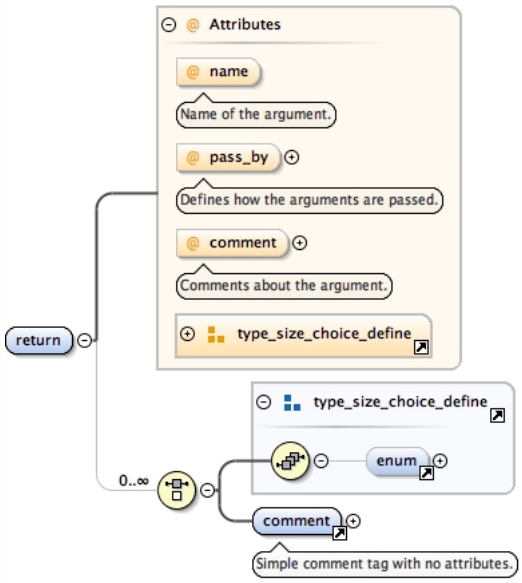
Source

```
<xs:element name="arg">
  <xs:complexType>
    <xs:choice minOccurs="0" maxOccurs="unbounded">
      <xs:group ref="type_size_choice_define"/>
      <xs:element ref="comment"/>
    </xs:choice>
    <xs:attribute name="name" use="required">
      <xs:annotation>
        <xs:documentation>Name of the argument.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="pass_by" type="pass_by_define">
      <xs:annotation>
        <xs:documentation>Defines how the arguments are passed.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="comment" type="xs:string">
      <xs:annotation>
        <xs:documentation>Comments about the argument.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attributeGroup ref="type_size_choice_define"/>
  </xs:complexType>
</xs:element>
```

Element `return`

Namespace: No namespace

Diagram



Properties	content: `complex`
Model	(enum{0,1}) | comment
Children	comment, enum
Instance	`<return comment="" data_type="" name="" pass_by="" size="" type="">`

```
  <enum name="">{0,1}</enum>
  <comment>{1,1}</comment>
</return>
```

Attributes

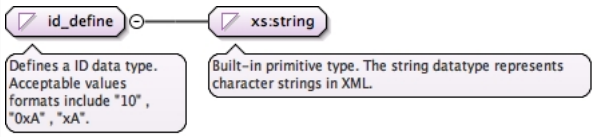
QName	Type	Use	
comment	xs:string	optional	
	Comments about the argument.		
data_type	union of(xs:string, restriction of xs:token, restriction of xs:token)	optional	
name		optional	
	Name of the argument.		
pass_by	pass_by_define	optional	
	Defines how the arguments are passed.		
size	xs:nonNegativeInteger	optional	
	The size of the argument.		
type	union of(xs:string, restriction of xs:token, restriction of xs:token)	optional	

Source

```
<xs:element name="return">
  <xs:complexType>
    <xs:choice minOccurs="0" maxOccurs="unbounded">
      <xs:group ref="type_size_choice_define"/>
      <xs:element ref="comment"/>
    </xs:choice>
    <xs:attribute name="name">
      <xs:annotation>
        <xs:documentation>Name of the argument.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="pass_by" type="pass_by_define">
      <xs:annotation>
        <xs:documentation>Defines how the arguments are passed.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="comment" type="xs:string">
      <xs:annotation>
        <xs:documentation>Comments about the argument.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attributeGroup ref="type_size_choice_define"/>
  </xs:complexType>
</xs:element>
```

Simple Type(s)

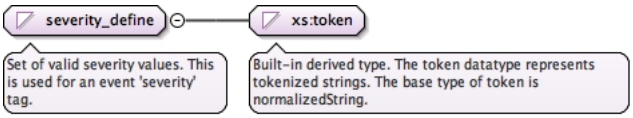
Simple Type `id_define`

Namespace	No namespace
Annotations	Defines a ID data type. Acceptable values formats include "10" , "0xA" , "xA".
Diagram	
Type	restriction of xs:string
Facets	pattern ((0?x\d+)|\d+)
Used by	Attribute event/@id

Source

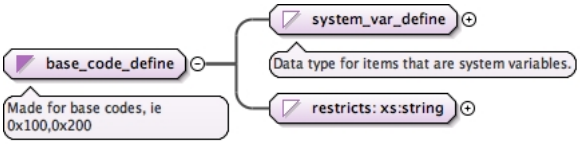
```
<xs:simpleType name="id_define">
  <xs:annotation>
    <xs:documentation>Defines a ID data type. Acceptable values formats include "10" , "0xA" ,
 "xA".</xs:documentation>
  </xs:annotation>
  <xs:restriction base="xs:string">
    <xs:pattern value="((0?x\d+)|\d+)"/>
  </xs:restriction>
</xs:simpleType>
```

Simple Type `severity_define`

Namespace	No namespace
Annotations	`Set of valid severity values. This is used for an event 'severity' tag.`
Diagram	
Type	restriction of xs:token
Facets	enumeration `COMMAND` enumeration `ACTIVITY_LO` enumeration `ACTIVITY_HI` enumeration `WARNING_LO` enumeration `WARNING_HI` enumeration `DIAGNOSTIC` enumeration `FATAL`
Used by	Attribute event/@severity
Source	

```
<xs:simpleType name="severity_define">
  <xs:annotation>
    <xs:documentation>Set of valid severity values. This is used for an event 'severity' tag.</
xs:documentation>
  </xs:annotation>
  <xs:restriction base="xs:token">
    <xs:enumeration value="COMMAND"/>
    <xs:enumeration value="ACTIVITY_LO"/>
    <xs:enumeration value="ACTIVITY_HI"/>
    <xs:enumeration value="WARNING_LO"/>
    <xs:enumeration value="WARNING_HI"/>
    <xs:enumeration value="DIAGNOSTIC"/>
    <xs:enumeration value="FATAL"/>
  </xs:restriction>
</xs:simpleType>
```

Simple Type `base_code_define`

Namespace	No namespace
Annotations	`Made for base codes, ie 0x100,0x200`
Diagram	
Type	union of(system_var_define, restriction of xs:string)
Used by	Attribute events/@event_base
Source	

```
<xs:simpleType name="base_code_define">
  <xs:annotation>
    <xs:documentation>Made for base codes, ie 0x100,0x200</xs:documentation>
  </xs:annotation>
  <xs:union memberTypes="system_var_define">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:pattern value="(((0?x\d+)|\d+)(,?))+"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:union>
</xs:simpleType>
```

Simple Type `full_items_define`

Namespace	No namespace
Annotations	`Valid values for the full tag.`

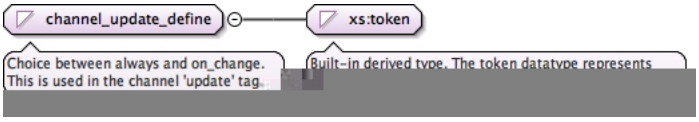
Diagram	
Type	restriction of xs:token
Facets	

	enumeration	Cmd
	enumeration	CmdResponse

Source

```
<xs:simpleType name="component_role_define">
  <xs:annotation>
    <xs:documentation>Choice for component roles.</xs:documentation>
  </xs:annotation>
  <xs:restriction base="xs:token">
    <xs:enumeration value="LogEvent"/>
    <xs:enumeration value="LogTextEvent"/>
    <xs:enumeration value="TimeGet"/>
    <xs:enumeration value="ParamSet"/>
    <xs:enumeration value="ParamGet"/>
    <xs:enumeration value="Telemetry"/>
    <xs:enumeration value="CmdRegistration"/>
    <xs:enumeration value="Cmd"/>
    <xs:enumeration value="CmdResponse"/>
  </xs:restriction>
</xs:simpleType>
```

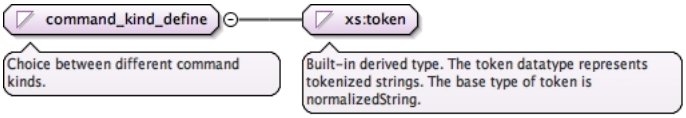
Simple Type channel_update_define

Namespace	No namespace
Annotations	Choice between always and on_change. This is used in the channel 'update' tag.
Diagram	
Type	restriction of xs:token
Facets	enumeration: always enumeration: on_change

Source

```
<xs:simpleType name="channel_update_define">
  <xs:annotation>
    <xs:documentation>Choice between always and on_change. This is used in the channel 'update'
 tag.</xs:documentation>
  </xs:annotation>
  <xs:restriction base="xs:token">
    <xs:enumeration value="always"/>
    <xs:enumeration value="on_change"/>
  </xs:restriction>
</xs:simpleType>
```

Simple Type command_kind_define

Namespace	No namespace
Annotations	Choice between different command kinds.
Diagram	
Type	restriction of xs:token
Facets	enumeration: async enumeration: sync enumeration: guarded

Source

```
<xs:simpleType name="command_kind_define">
  <xs:annotation>
    <xs:documentation>Choice between different command kinds.</xs:documentation>
  </xs:annotation>
  <xs:restriction base="xs:token">
    <xs:enumeration value="async"/>
    <xs:enumeration value="sync"/>
    <xs:enumeration value="guarded"/>
  </xs:restriction>
</xs:simpleType>
```

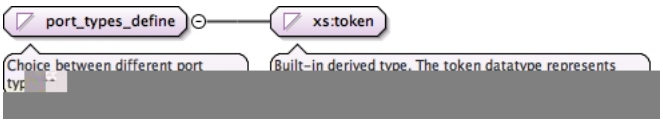
Simple Type component_types_define

Namespace	No namespace

Annotations	Choice between active, passive, or queued.
Diagram	
Type	restriction of xs:token
Facets	enumeration: active enumeration: passive enumeration: queued

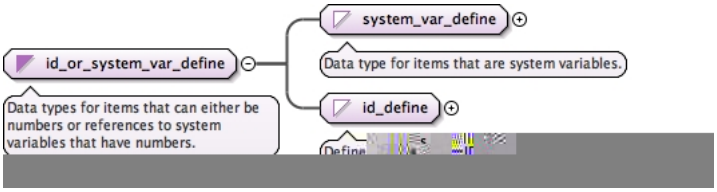
Source	`<xs:simpleType name="component_types_define">`
	`  <xs:annotation>`
	`  <xs:documentation>Choice between active, passive, or queued.</xs:documentation>`
	`  </xs:annotation>`
	`  <xs:restriction base="xs:token">`
	`  <xs:enumeration value="active"/>`
	`  <xs:enumeration value="passive"/>`
	`  <xs:enumeration value="queued"/>`
	`  </xs:restriction>`
	`</xs:simpleType>`

Simple Type `port_types_define`

Namespace	No namespace
Annotations	Choice between different port types.
Diagram	
Type	restriction of xs:token
Facets	enumeration: input enumeration: sync_input enumeration: guarded_input enumeration: async_input enumeration: model_input enumeration: output

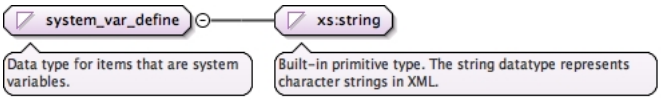
Source	`<xs:simpleType name="port_types_define">`
	`  <xs:annotation>`
	`  <xs:documentation>Choice between different port types.</xs:documentation>`
	`  </xs:annotation>`
	`  <xs:restriction base="xs:token">`
	`  <xs:enumeration value="input"/>`
	`  <xs:enumeration value="sync_input"/>`
	`  <xs:enumeration value="guarded_input"/>`
	`  <xs:enumeration value="async_input"/>`
	`  <xs:enumeration value="model_input"/>`
	`  <xs:enumeration value="output"/>`
	`  </xs:restriction>`
	`</xs:simpleType>`

Simple Type `id_or_system_var_define`

Namespace	No namespace
Annotations	Data types for items that can either be numbers or references to system variables that have numbers.
Diagram	
Type	union of(system_var_define, id_define)
Source	`<xs:simpleType name="id_or_system_var_define">`

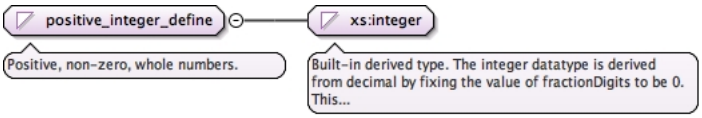
```
  <xs:annotation>
    <xs:documentation>Data types for items that can either be numbers or references to system
 variables that have numbers.</xs:documentation>
  </xs:annotation>
  <xs:union memberTypes="system_var_define id_define"/>
</xs:simpleType>
```

Simple Type `system_var_define`

Namespace	No namespace
Annotations	`Data type for items that are system variables.`
Diagram	
Type	restriction of xs:string
Facets	pattern `$[\w|_|\-]+`

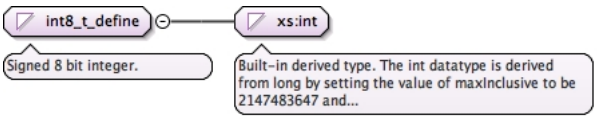
Source	`<xs:simpleType name="system_var_define">`
	`  <xs:annotation>`
	`  <xs:documentation>Data type for items that are system variables.</xs:documentation>`
	`  </xs:annotation>`
	`  <xs:restriction base="xs:string">`
	`  <xs:pattern value="$[\w|_|\-]+"/>`
	`  </xs:restriction>`
	`</xs:simpleType>`

Simple Type `positive_integer_define`

Namespace	No namespace
Annotations	`Positive, non-zero, whole numbers.`
Diagram	
Type	restriction of xs:integer
Facets	minInclusive `1`

Source	`<xs:simpleType name="positive_integer_define">`
	`  <xs:annotation>`
	`  <xs:documentation>Positive, non-zero, whole numbers.</xs:documentation>`
	`  </xs:annotation>`
	`  <xs:restriction base="xs:integer">`
	`  <xs:minInclusive value="1"/>`
	`  </xs:restriction>`
	`</xs:simpleType>`

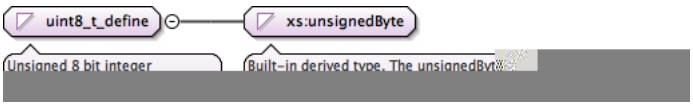
Simple Type `int8_t_define`

Namespace	No namespace
Annotations	`Signed 8 bit integer.`
Diagram	
Type	restriction of xs:int
Facets	maxInclusive `127` minInclusive `-128`
Used by	Simple Type I8_define

Source	`<xs:simpleType name="int8_t_define">`
	`  <xs:annotation>`
	`  <xs:documentation>Signed 8 bit integer.</xs:documentation>`
	`  </xs:annotation>`
	`  <xs:restriction base="xs:int">`
	`  <xs:minInclusive value="-128"/>`
	`  <xs:maxInclusive value="127"/>`

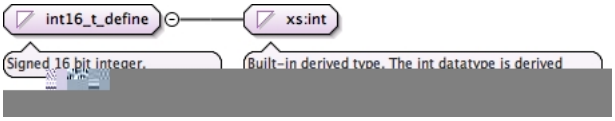
```
  </xs:restriction>
</xs:simpleType>
```

Simple Type `uint8_t_define`

Namespace	No namespace
Annotations	`Unsigned 8 bit integer`
Diagram	
Type	restriction of xs:unsignedByte
Facets	maxInclusive `255` minInclusive `0`
Used by	Simple Type U8_define

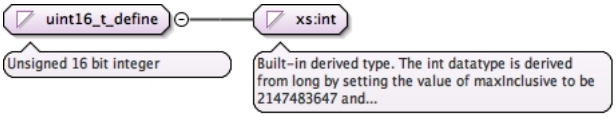
Source	`<xs:simpleType name="uint8_t_define">`
	`  <xs:annotation>`
	`  <xs:documentation>Unsigned 8 bit integer</xs:documentation>`
	`  </xs:annotation>`
	`  <xs:restriction base="xs:unsignedByte">`
	`  <xs:minInclusive value="0"/>`
	`  <xs:maxInclusive value="255"/>`
	`  </xs:restriction>`
	`</xs:simpleType>`

Simple Type `int16_t_define`

Namespace	No namespace
Annotations	`Signed 16 bit integer.`
Diagram	
Type	restriction of xs:int
Facets	maxInclusive `32767` minInclusive `-32768`
Used by	Simple Type I16_define

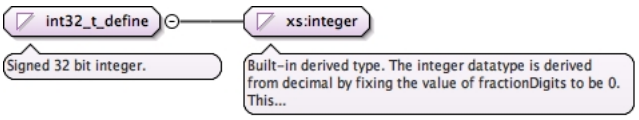
Source	`<xs:simpleType name="int16_t_define">`
	`  <xs:annotation>`
	`  <xs:documentation>Signed 16 bit integer.</xs:documentation>`
	`  </xs:annotation>`
	`  <xs:restriction base="xs:int">`
	`  <xs:minInclusive value="-32768"/>`
	`  <xs:maxInclusive value="32767"/>`
	`  </xs:restriction>`
	`</xs:simpleType>`

Simple Type `uint16_t_define`

Namespace	No namespace
Annotations	`Unsigned 16 bit integer`
Diagram	
Type	restriction of xs:int
Facets	maxInclusive `65535` minInclusive `0`
Used by	Simple Type U16_define
Source	`<xs:simpleType name="uint16_t_define">`

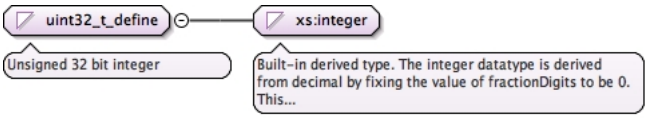
```
  <xs:annotation>
    <xs:documentation>Unsigned 16 bit integer</xs:documentation>
  </xs:annotation>
  <xs:restriction base="xs:int">
    <xs:minInclusive value="0"/>
    <xs:maxInclusive value="65535"/>
  </xs:restriction>
</xs:simpleType>
```

Simple Type `int32_t_define`

Namespace	No namespace
Annotations	`Signed 32 bit integer.`
Diagram	
Type	restriction of xs:integer
Facets	maxInclusive `2147483647` minInclusive `-2147483648`
Used by	Simple Types I32_define, NATIVE_INT_TYPE_define

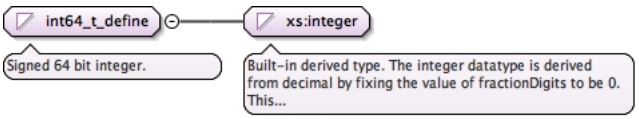
Source	`<xs:simpleType name="int32_t_define">`
	`  <xs:annotation>`
	`  <xs:documentation>Signed 32 bit integer.</xs:documentation>`
	`  </xs:annotation>`
	`  <xs:restriction base="xs:integer">`
	`  <xs:minInclusive value="-2147483648"/>`
	`  <xs:maxInclusive value="2147483647"/>`
	`  </xs:restriction>`
	`</xs:simpleType>`

Simple Type `uint32_t_define`

Namespace	No namespace
Annotations	`Unsigned 32 bit integer`
Diagram	
Type	restriction of xs:integer
Facets	maxInclusive `4294967295` minInclusive `0`
Used by	Simple Types NATIVE_UINT_TYPE_define, U32_define

Source	`<xs:simpleType name="uint32_t_define">`
	`  <xs:annotation>`
	`  <xs:documentation>Unsigned 32 bit integer</xs:documentation>`
	`  </xs:annotation>`
	`  <xs:restriction base="xs:integer">`
	`  <xs:minInclusive value="0"/>`
	`  <xs:maxInclusive value="4294967295"/>`
	`  </xs:restriction>`
	`</xs:simpleType>`

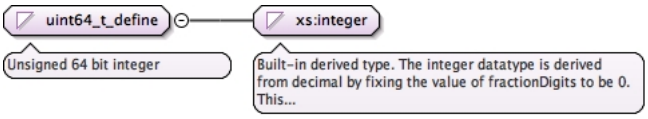
Simple Type `int64_t_define`

Namespace	No namespace
Annotations	`Signed 64 bit integer.`
Diagram	
Type	restriction of xs:integer

Facets	maxInclusive	9223372036854775807
	minInclusive	-9223372036854775808
Used by	Simple Type	I64_define
Source		

```
<xs:simpleType name="int64_t_define">
  <xs:annotation>
    <xs:documentation>Signed 64 bit integer.</xs:documentation>
  </xs:annotation>
  <xs:restriction base="xs:integer">
    <xs:minInclusive value="-9223372036854775808"/>
    <xs:maxInclusive value="9223372036854775807"/>
  </xs:restriction>
</xs:simpleType>
```

Simple Type `uint64_t_define`

Namespace	No namespace
Annotations	Unsigned 64 bit integer
Diagram	
Type	restriction of xs:integer
Facets	maxInclusive: 18446744073709551615 minInclusive: 0
Used by	Simple Type: U64_define
Source	

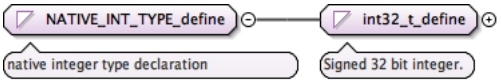
```
<xs:simpleType name="uint64_t_define">
  <xs:annotation>
    <xs:documentation>Unsigned 64 bit integer</xs:documentation>
  </xs:annotation>
  <xs:restriction base="xs:integer">
    <xs:minInclusive value="0"/>
    <xs:maxInclusive value="18446744073709551615"/>
  </xs:restriction>
</xs:simpleType>
```

Simple Type `not_user_cpp_type_define`

Namespace	No namespace
Annotations	Ensures data is not of the names of any other user defined C++ name.
Diagram	
Type	xs:string
Source	

```
<xs:simpleType name="not_user_cpp_type_define">
  <xs:annotation>
    <xs:documentation>Ensures data is not of the names of any other user defined C++ name.</
xs:documentation>
  </xs:annotation>
  <xs:restriction base="xs:string"/>
</xs:simpleType>
```

Simple Type `NATIVE_INT_TYPE_define`

Namespace	No namespace
Annotations	native integer type declaration
Diagram	
Type	int32_t_define
Type hierarchy	• xs:integer • int32_t_define

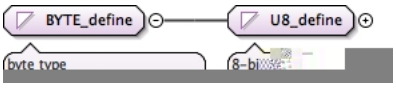
Type	uint8_t_define

Type hierarchy	• xs:unsignedByte
	  • uint8_t_define
	  • U8_define

Facets	maxInclusive `255` minInclusive `0`
Used by	Simple Type BYTE_define

Source	`<xs:simpleType name="U8_define">`
	`  <xs:annotation>`
	`  <xs:documentation>8-bit unsigned integer</xs:documentation>`
	`  </xs:annotation>`
	`  <xs:restriction base="uint8_t_define"/>`
	`</xs:simpleType>`

Simple Type `BYTE_define`

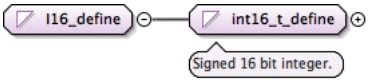
Namespace	No namespace
Annotations	`byte type`
Diagram	
Type	U8_define

Type hierarchy	• xs:unsignedByte
	  • uint8_t_define
	  • U8_define
	  • BYTE_define

Facets	maxInclusive `255` minInclusive `0`

Source	`<xs:simpleType name="BYTE_define">`
	`  <xs:annotation>`
	`  <xs:documentation>byte type</xs:documentation>`
	`  </xs:annotation>`
	`  <xs:restriction base="U8_define"/>`
	`</xs:simpleType>`

Simple Type `I16_define`

Namespace	No namespace
Diagram	
Type	int16_t_define

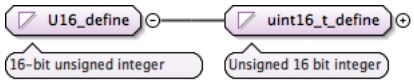
Type hierarchy	• xs:int
	  • int16_t_define
	  • I16_define

Facets	maxInclusive `32767` minInclusive `-32768`

Source	`<xs:simpleType name="I16_define">`
	`  <xs:restriction base="int16_t_define"/>`
	`</xs:simpleType>`

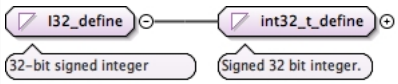
Simple Type `U16_define`

Namespace	No namespace

Annotations	16-bit unsigned integer
Diagram	
Type	uint16_t_define
Type hierarchy	• xs:int • uint16_t_define • U16_define
Facets	maxInclusive 65535 minInclusive 0

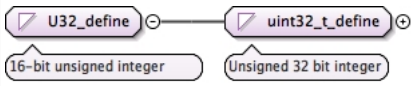
Source	`<xs:simpleType name="U16_define">`
	`  <xs:annotation>`
	`  <xs:documentation>16-bit unsigned integer</xs:documentation>`
	`  </xs:annotation>`
	`  <xs:restriction base="uint16_t_define"/>`
	`</xs:simpleType>`

Simple Type I32_define

Namespace	No namespace
Annotations	32-bit signed integer
Diagram	
Type	int32_t_define
Type hierarchy	• xs:integer • int32_t_define • I32_define
Facets	maxInclusive 2147483647 minInclusive -2147483648

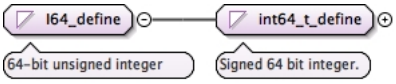
Source	`<xs:simpleType name="I32_define">`
	`  <xs:annotation>`
	`  <xs:documentation>32-bit signed integer</xs:documentation>`
	`  </xs:annotation>`
	`  <xs:restriction base="int32_t_define"/>`
	`</xs:simpleType>`

Simple Type U32_define

Namespace	No namespace
Annotations	16-bit unsigned integer
Diagram	
Type	uint32_t_define
Type hierarchy	• xs:integer • uint32_t_define • U32_define
Facets	maxInclusive 4294967295 minInclusive 0

Source	`<xs:simpleType name="U32_define">`
	`  <xs:annotation>`
	`  <xs:documentation>16-bit unsigned integer</xs:documentation>`
	`  </xs:annotation>`
	`  <xs:restriction base="uint32_t_define"/>`
	`</xs:simpleType>`

Simple Type `I64_define`

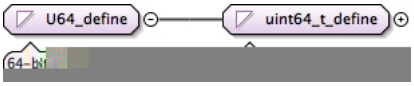
Namespace	No namespace
Annotations	`64-bit unsigned integer`
Diagram	
Type	int64_t_define

Type hierarchy	• xs:integer
	  • int64_t_define
	  • I64_define

Facets	maxInclusive `9223372036854775807` minInclusive `-9223372036854775808`

Source	`<xs:simpleType name="I64_define">`
	`  <xs:annotation>`
	`  <xs:documentation>64-bit unsigned integer</xs:documentation>`
	`  </xs:annotation>`
	`  <xs:restriction base="int64_t_define"/>`
	`</xs:simpleType>`

Simple Type `U64_define`

Namespace	No namespace
Annotations	`64-bit unsigned integer`
Diagram	
Type	uint64_t_define

Type hierarchy	• xs:integer
	  • uint64_t_define
	  • U64_define

Facets	maxInclusive `18446744073709551615` minInclusive `0`

Source	`<xs:simpleType name="U64_define">`
	`  <xs:annotation>`
	`  <xs:documentation>64-bit unsigned integer</xs:documentation>`
	`  </xs:annotation>`
	`  <xs:restriction base="uint64_t_define"/>`
	`</xs:simpleType>`

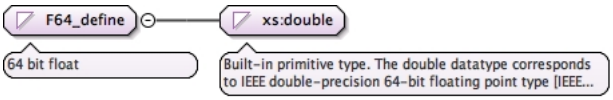
Simple Type `F32_define`

Namespace	No namespace
Annotations	`32 bit float`
Diagram	
Type	xs:float

Source	`<xs:simpleType name="F32_define">`
	`  <xs:annotation>`
	`  <xs:documentation>32 bit float</xs:documentation>`
	`  </xs:annotation>`
	`  <xs:restriction base="xs:float"/>`
	`</xs:simpleType>`

Simple Type `F64_define`

Namespace	No namespace

Annotations	64 bit float
Diagram	
Type	xs:double

Source	`<xs:simpleType name="F64_define">`
	`  <xs:annotation>`
	`  <xs:documentation>64 bit float</xs:documentation>`
	`  </xs:annotation>`
	`  <xs:restriction base="xs:double"/>`
	`</xs:simpleType>`

Attribute(s)

Attribute `item / @name`

Namespace	No namespace
Annotations	Name of the enum item.
Properties	use: required
Used by	Element item

Source	`<xs:attribute name="name" use="required">`
	`  <xs:annotation>`
	`  <xs:documentation>Name of the enum item.</xs:documentation>`
	`  </xs:annotation>`
	`</xs:attribute>`

Attribute `item / @value`

Namespace	No namespace
Annotations	The value being sent through the enum item.
Used by	Element item

Source	`<xs:attribute name="value">`
	`  <xs:annotation>`
	`  <xs:documentation>The value being sent through the enum item.</xs:documentation>`
	`  </xs:annotation>`
	`</xs:attribute>`

Attribute `item / @comment`

Namespace	No namespace
Annotations	Comment about the enum item.
Used by	Element item

Source	`<xs:attribute name="comment">`
	`  <xs:annotation>`
	`  <xs:documentation>Comment about the enum item.</xs:documentation>`
	`  </xs:annotation>`
	`</xs:attribute>`

Attribute `enum / @name`

Namespace	No namespace
Annotations	Enum Name.
Properties	use: required
Used by	Element enum

Source	`<xs:attribute name="name" use="required">`
	`  <xs:annotation>`
	`  <xs:documentation>Enum Name.</xs:documentation>`
	`  </xs:annotation>`
	`</xs:attribute>`

Attribute `external_arg_define / arg / @name`

Namespace	No namespace

Annotations	Name of the argument.
Properties	use: required
Used by	Element external_arg_define/arg

Source	`<xs:attribute name="name" use="required">`
	`  <xs:annotation>`
	`  <xs:documentation>Name of the argument.</xs:documentation>`
	`  </xs:annotation>`
	`</xs:attribute>`

Attribute `external_arg_define / arg / @comment`

Namespace	No namespace
Annotations	Comments about the argument.
Type	xs:string
Properties	content: simple
Used by	Element external_arg_define/arg

Source	`<xs:attribute name="comment" type="xs:string">`
	`  <xs:annotation>`
	`  <xs:documentation>Comments about the argument.</xs:documentation>`
	`  </xs:annotation>`
	`</xs:attribute>`

Attribute `type_size_choice_define / @data_type`

Namespace	No namespace
Type	union of(xs:string, restriction of xs:token, restriction of xs:token)
Properties	content: simple
Used by	Attribute Group type_size_choice_define

Source	`<xs:attribute name="data_type">`
	`  <xs:simpleType>`
	`  <xs:union memberTypes="xs:string">`
	`  <xs:simpleType>`
	`  <xs:restriction base="xs:token">`
	`  <xs:enumeration value="string"/>`
	`  </xs:restriction>`
	`  </xs:simpleType>`
	`  <xs:simpleType>`
	`  <xs:restriction base="xs:token">`
	`  <xs:enumeration value="ENUM"/>`
	`  </xs:restriction>`
	`  </xs:simpleType>`
	`  </xs:union>`
	`  </xs:simpleType>`
	`</xs:attribute>`

Attribute `type_size_choice_define / @type`

Namespace	No namespace
Type	union of(xs:string, restriction of xs:token, restriction of xs:token)
Properties	content: simple
Used by	Attribute Group type_size_choice_define

Source	`<xs:attribute name="type">`
	`  <xs:simpleType>`
	`  <xs:union memberTypes="xs:string">`
	`  <xs:simpleType>`
	`  <xs:restriction base="xs:token">`
	`  <xs:enumeration value="string"/>`
	`  </xs:restriction>`
	`  </xs:simpleType>`
	`  <xs:simpleType>`
	`  <xs:restriction base="xs:token">`
	`  <xs:enumeration value="ENUM"/>`
	`  </xs:restriction>`
	`  </xs:simpleType>`
	`  </xs:union>`

```
  </xs:simpleType>
</xs:attribute>
```

Attribute type_size_choice_define / @size

Namespace	No namespace
Annotations	The size of the argument.
Type	xs:nonNegativeInteger
Properties	content: simple
Used by	Attribute Group type_size_choice_define

Source	`<xs:attribute name="size" type="xs:nonNegativeInteger">`
	`  <xs:annotation>`
	`  <xs:documentation>The size of the argument.</xs:documentation>`
	`  </xs:annotation>`
	`</xs:attribute>`

Attribute event / @name

Namespace	No namespace
Annotations	Name of the event.
Properties	use: required
Used by	Element event

Source	`<xs:attribute name="name" use="required">`
	`  <xs:annotation>`
	`  <xs:documentation>Name of the event.</xs:documentation>`
	`  </xs:annotation>`
	`</xs:attribute>`

Attribute event / @id

Namespace	No namespace
Annotations	Event ID.
Type	id_define
Properties	use: required
Facets	pattern ((0?x\d+)|\d+)
Used by	Element event

Source	`<xs:attribute name="id" use="required" type="id_define">`
	`  <xs:annotation>`
	`  <xs:documentation>Event ID.</xs:documentation>`
	`  </xs:annotation>`
	`</xs:attribute>`

Attribute event / @severity

Namespace	No namespace
Annotations	Severity of event.
Type	severity_define
Properties	use: required
Facets	enumeration COMMAND
	enumeration ACTIVITY_LO
	enumeration ACTIVITY_HI
	enumeration WARNING_LO
	enumeration WARNING_HI
	enumeration DIAGNOSTIC
	enumeration FATAL
Used by	Element event

Source	`<xs:attribute name="severity" use="required" type="severity_define">`
	`  <xs:annotation>`
	`  <xs:documentation>Severity of event.</xs:documentation>`
	`  </xs:annotation>`
	`</xs:attribute>`

Attribute `event / @format_string`

Namespace	No namespace
Annotations	`Output string that will be shown in ground system software.`
Properties	use: `required`
Used by	Element event

Source	`<xs:attribute name="format_string" use="required">`
	`  <xs:annotation>`
	`  <xs:documentation>Output string that will be shown in ground system software.</xs:documentation>`
	`  </xs:annotation>`
	`</xs:attribute>`

Attribute `event / @throttle`

Namespace	No namespace
Annotations	`Determines how many of the events are generated before the program stops them.`
Type	xs:nonNegativeInteger
Properties	content: `simple`
Used by	Element event

Source	`<xs:attribute name="throttle" type="xs:nonNegativeInteger">`
	`  <xs:annotation>`
	`  <xs:documentation>Determines how many of the events are generated before the program stops`
	` them.</xs:documentation>`
	`  </xs:annotation>`
	`</xs:attribute>`

Attribute `events / @event_base`

Namespace	No namespace
Annotations	`Base at which ids start from.`
Type	base_code_define
Properties	content: `simple`
Used by	Element events

Source	`<xs:attribute name="event_base" type="base_code_define">`
	`  <xs:annotation>`
	`  <xs:documentation>Base at which ids start from.</xs:documentation>`
	`  </xs:annotation>`
	`</xs:attribute>`

Attribute `arg_define / arg / @name`

Namespace	No namespace
Annotations	`Name of the argument.`
Properties	use: `required`
Used by	Element arg_define/arg

Source	`<xs:attribute name="name" use="required">`
	`  <xs:annotation>`
	`  <xs:documentation>Name of the argument.</xs:documentation>`
	`  </xs:annotation>`
	`</xs:attribute>`

Attribute `arg_define / arg / @pass_by`

Namespace	No namespace
Annotations	`Defines how the arguments are passed.`

Type	pass_by_define
Properties	content: simple
Facets	enumeration reference enumeration value enumeration pointer
Used by	Element arg_define/arg

Source	`<xs:attribute name="pass_by" type="pass_by_define">`
	`  <xs:annotation>`
	`  <xs:documentation>Defines how the arguments are passed.</xs:documentation>`
	`  </xs:annotation>`
	`</xs:attribute>`

Attribute `arg_define / arg / @comment`

Namespace	No namespace
Annotations	Comments about the argument.
Type	xs:string
Properties	content: simple
Used by	Element arg_define/arg

Source	`<xs:attribute name="comment" type="xs:string">`
	`  <xs:annotation>`
	`  <xs:documentation>Comments about the argument.</xs:documentation>`
	`  </xs:annotation>`
	`</xs:attribute>`

Attribute `return / @name`

Namespace	No namespace
Annotations	Name of the argument.
Used by	Element return

Source	`<xs:attribute name="name">`
	`  <xs:annotation>`
	`  <xs:documentation>Name of the argument.</xs:documentation>`
	`  </xs:annotation>`
	`</xs:attribute>`

Attribute `return / @pass_by`

Namespace	No namespace
Annotations	Defines how the arguments are passed.
Type	pass_by_define
Properties	content: simple
Facets	enumeration reference enumeration value enumeration pointer
Used by	Element return

Source	`<xs:attribute name="pass_by" type="pass_by_define">`
	`  <xs:annotation>`
	`  <xs:documentation>Defines how the arguments are passed.</xs:documentation>`
	`  </xs:annotation>`
	`</xs:attribute>`

Attribute `return / @comment`

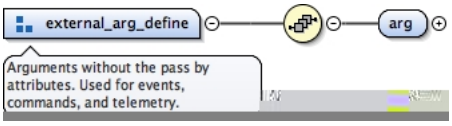
Namespace	No namespace
Annotations	Comments about the argument.
Type	xs:string

Properties	content: simple
Used by	Element return

Source	`<xs:attribute name="comment" type="xs:string">`
	`  <xs:annotation>`
	`  <xs:documentation>Comments about the argument.</xs:documentation>`
	`  </xs:annotation>`
	`</xs:attribute>`

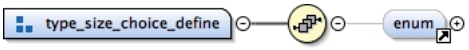
Element Group(s)

Element Group `external_arg_define`

Namespace	No namespace
Annotations	Arguments without the pass by attributes. Used for events, commands, and telemetry.
Diagram	
Used by	Element args
Model	arg
Children	arg

Source	`<xs:group name="external_arg_define">`
	`  <xs:annotation>`
	`  <xs:documentation>Arguments without the pass by attributes. Used for events, commands, and telemetry.</xs:documentation>`
	`  </xs:annotation>`
	`  <xs:sequence>`
	`  <xs:element name="arg">`
	`  <xs:complexType>`
	`  <xs:choice minOccurs="0" maxOccurs="unbounded">`
	`  <xs:group ref="type_size_choice_define"/>`
	`  <xs:element ref="comment"/>`
	`  </xs:choice>`
	`  <xs:attribute name="name" use="required">`
	`  <xs:annotation>`
	`  <xs:documentation>Name of the argument.</xs:documentation>`
	`  </xs:annotation>`
	`  </xs:attribute>`
	`  <xs:attribute name="comment" type="xs:string">`
	`  <xs:annotation>`
	`  <xs:documentation>Comments about the argument.</xs:documentation>`
	`  </xs:annotation>`
	`  </xs:attribute>`
	`  <xs:attributeGroup ref="type_size_choice_define"/>`
	`  </xs:complexType>`
	`  </xs:element>`
	`  </xs:sequence>`
	`</xs:group>`

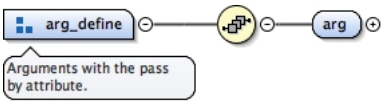
Element Group `type_size_choice_define`

Namespace	No namespace
Diagram	
Used by	Elements arg_define/arg, external_arg_define/arg, return
Model	enum{0,1}
Children	enum

Source	`<xs:group name="type_size_choice_define">`
	`  <xs:sequence>`
	`  <xs:element minOccurs="0" ref="enum"/>`
	`  </xs:sequence>`
	`</xs:group>`

Element Group `arg_define`

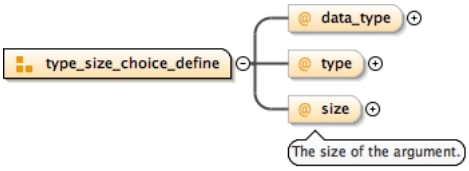
Namespace	No namespace

Annotations	`Arguments with the pass by attribute.`
Diagram	
Model	arg
Children	arg

Source	`<xs:group name="arg_define">`
	`  <xs:annotation>`
	`  <xs:documentation>Arguments with the pass by attribute.</xs:documentation>`
	`  </xs:annotation>`
	`  <xs:sequence>`
	`  <xs:element name="arg">`
	`  <xs:complexType>`
	`  <xs:choice minOccurs="0" maxOccurs="unbounded">`
	`  <xs:group ref="type_size_choice_define"/>`
	`  <xs:element ref="comment"/>`
	`  </xs:choice>`
	`  <xs:attribute name="name" use="required">`
	`  <xs:annotation>`
	`  <xs:documentation>Name of the argument.</xs:documentation>`
	`  </xs:annotation>`
	`  </xs:attribute>`
	`  <xs:attribute name="pass_by" type="pass_by_define">`
	`  <xs:annotation>`
	`  <xs:documentation>Defines how the arguments are passed.</xs:documentation>`
	`  </xs:annotation>`
	`  </xs:attribute>`
	`  <xs:attribute name="comment" type="xs:string">`
	`  <xs:annotation>`
	`  <xs:documentation>Comments about the argument.</xs:documentation>`
	`  </xs:annotation>`
	`  </xs:attribute>`
	`  <xs:attributeGroup ref="type_size_choice_define"/>`
	`  </xs:complexType>`
	`  </xs:element>`
	`  </xs:sequence>`
	`</xs:group>`

Attribute Group(s)

Attribute Group `type_size_choice_define`

Namespace	No namespace
Diagram	
Used by	Elements arg_define/arg, external_arg_define/arg, return
Attributes	

QName	Type	Use	
data_type	union of(xs:string, restriction of xs:token, restriction of xs:token)	optional	
size	xs:nonNegativeInteger	optional	
	`The size of the argument.`		
type	union of(xs:string, restriction of xs:token, restriction of xs:token)	optional	

Source

```
<xs:attributeGroup name="type_size_choice_define">
  <xs:attribute name="data_type">
    <xs:simpleType>
      <xs:union memberTypes="xs:string">
        <xs:simpleType>
          <xs:restriction base="xs:token">
            <xs:enumeration value="string"/>
          </xs:restriction>
        </xs:simpleType>
```

```
        <xs:simpleType>
          <xs:restriction base="xs:token">
            <xs:enumeration value="ENUM"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:union>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="type">
    <xs:simpleType>
      <xs:union memberTypes="xs:string">
        <xs:simpleType>
          <xs:restriction base="xs:token">
            <xs:enumeration value="string"/>
          </xs:restriction>
        </xs:simpleType>
        <xs:simpleType>
          <xs:restriction base="xs:token">
            <xs:enumeration value="ENUM"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:union>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="size" type="xs:nonNegativeInteger">
    <xs:annotation>
      <xs:documentation>The size of the argument.</xs:documentation>
    </xs:annotation>
  </xs:attribute>
</xs:attributeGroup>
```