XML Schema Documentation

Table of Contents

- Schema Document Properties
- Global Declarations
  - Element: **quotableFxSingleLeg**
  - Element: **quotableProduct**
- Global Definitions
  - Complex Type: **AcceptQuote**
  - Complex Type: **QuotableFxLeg**
  - Complex Type: **QuotableFxRate**
  - Complex Type: **QuotablePayment**
  - Complex Type: **QuotableProduct**
  - Complex Type: **QuoteAcceptanceConfirmed**
  - Complex Type: **QuoteAlreadyExpired**
  - Complex Type: **QuoteUpdated**
  - Complex Type: **RequestQuote**
  - Complex Type: **RequestQuoteResponse**
- Legend
- Glossary

top

Schema Document Properties

Target Namespace	**http://www.fpml.org/2007/FpML-4-4**
Version	$Revision: 2350 $
Element and Attribute Namespaces	• Global element and attribute declarations belong to this schema's target namespace. • By default, local element declarations belong to this schema's target namespace. • By default, local attribute declarations have no namespace.
Schema Composition	• This schema includes components from the following schema document(s): ○ fpml-msg-4-4.xsd

Declared Namespaces

Prefix	Namespace
Default namespace	**http://www.fpml.org/2007/FpML-4-4**
xml	http://www.w3.org/XML/1998/namespace
ecore	http://www.eclipse.org/emf/2002/Ecore
fpml	**http://www.fpml.org/2007/FpML-4-4**
dsig	http://www.w3.org/2000/09/xmldsig#
xsd	http://www.w3.org/2001/XMLSchema

Schema Component Representation

```
<xsd:schema nsPrefix="fpml" package="org.fpml" documentRoot="FpML" targetNamespace="http://
www.fpml.org/2007/FpML-4-4" version="$Revision: 2350 $"
attributeFormDefault="unqualified" elementFormDefault="qualified">
   <xsd:include schemaLocation="fpml-msg-4-4.xsd"/>
   ...
```

```
</xsd:schema>
```

top

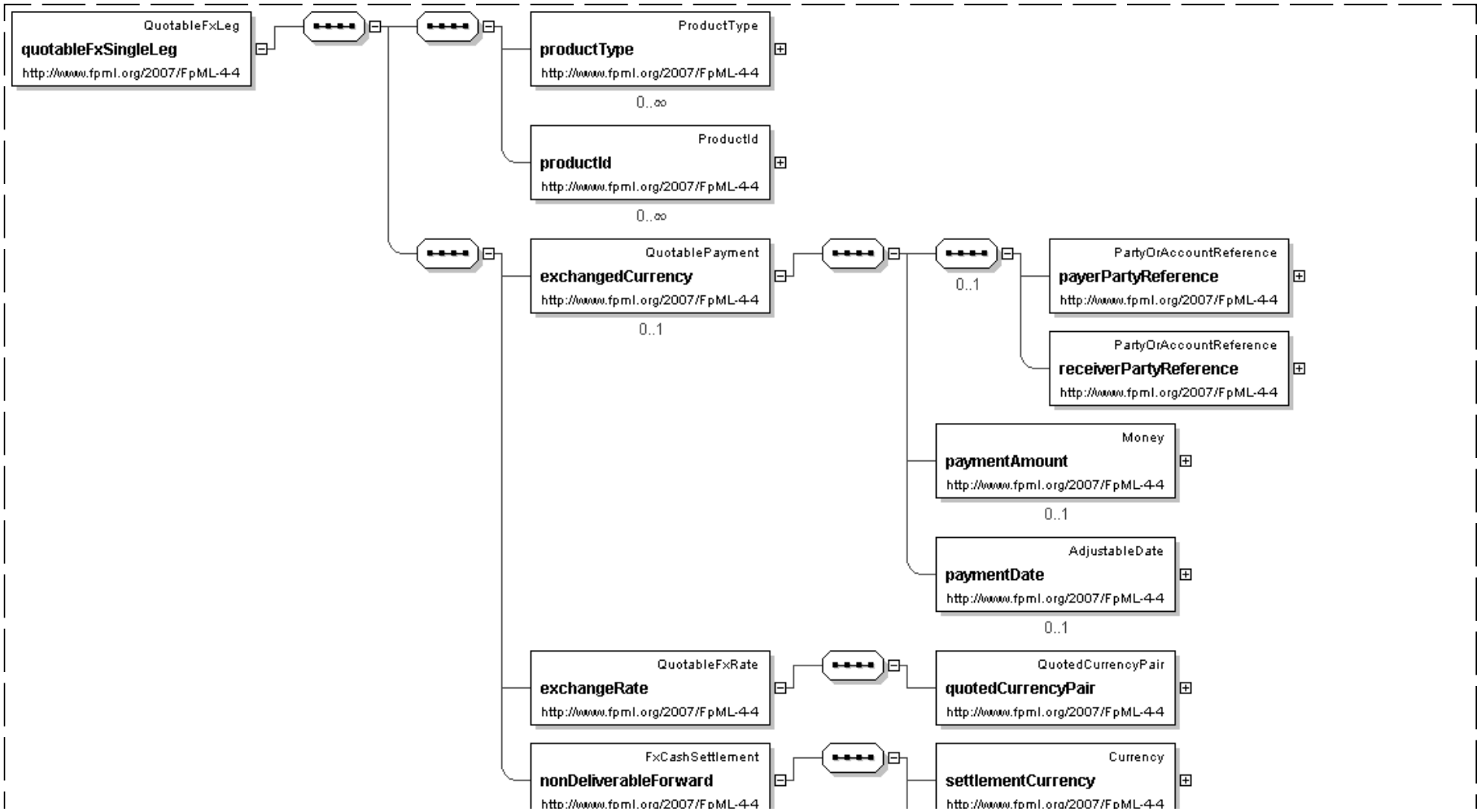
Global Declarations

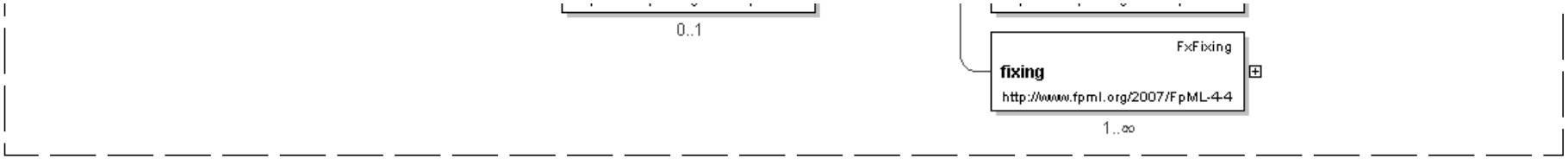
Element: quotableFxSingleLeg

- *This element can be used wherever the following element is referenced:*
  - quotableProduct

Name	quotableFxSingleLeg
Type	QuotableFxLeg
Nillable	no
Abstract	no

Logical Diagram

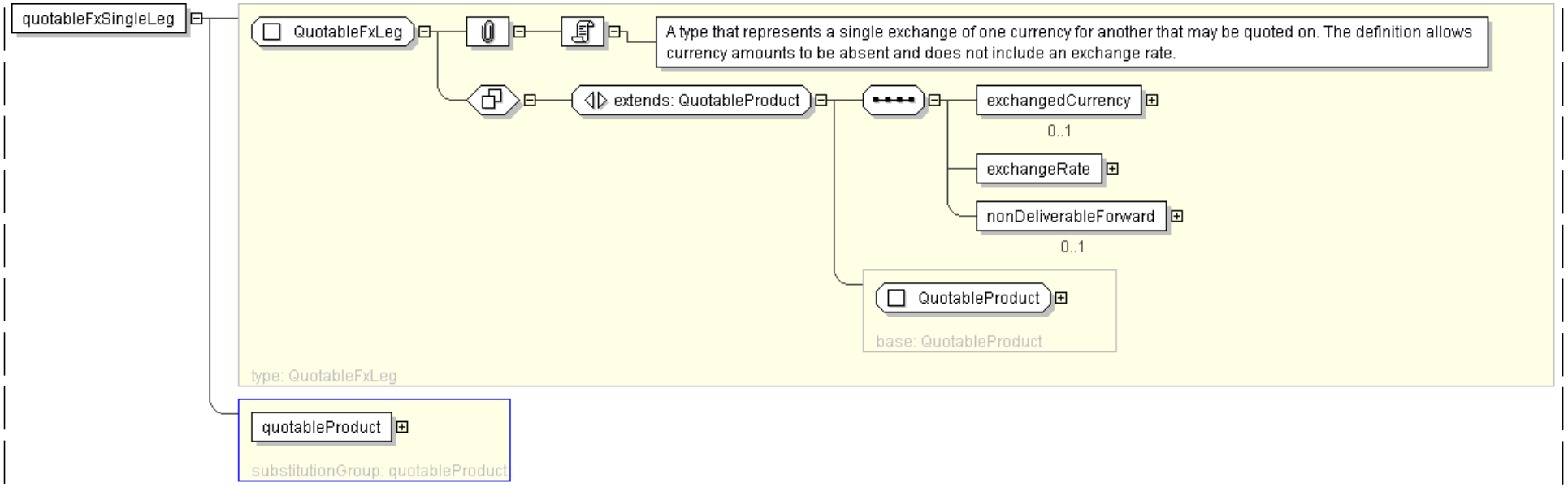




XML Instance Representation

```
<quotableFxSingleLeg>
   <productType> ProductType </productType> [0..*]
   'A classification of the type of product. FpML defines a simple product categorization using
   a coding scheme.'

   <productId> ProductId </productId> [0..*]
   'A product reference identifier allocated by a party. FpML does not define the domain
   values associated with this element. Note that the domain values for this element are
   not strictly an enumerated list.'

   <exchangedCurrency> QuotablePayment </exchangedCurrency> [0..1]
   <exchangeRate> QuotableFxRate </exchangeRate> [1]
   <nonDeliverableForward> FxCashSettlement </nonDeliverableForward> [0..1]
   'Used to describe a particular type of FX forward transaction that is settled in a
   single currency.'

</quotableFxSingleLeg>
```

Diagram



Schema Component Representation

```
<xsd:element name="quotableFxSingleLeg" type=" QuotableFxLeg
" substitutionGroup="quotableProduct"/>
```

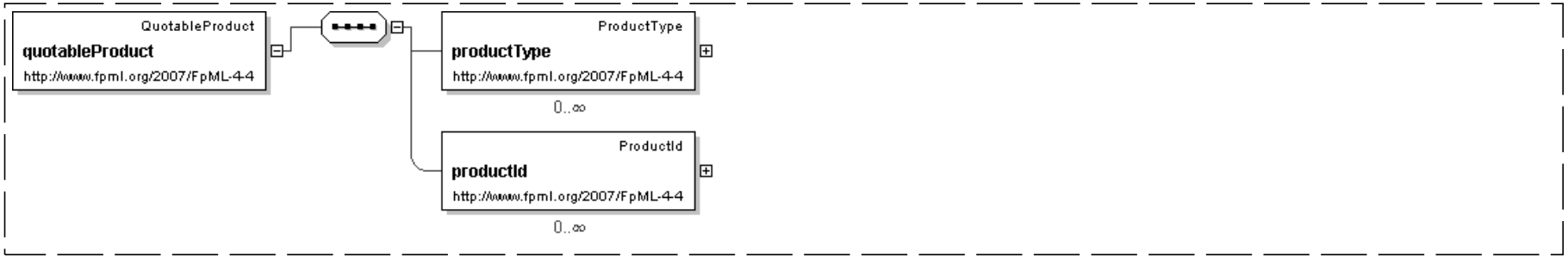
top

Element: quotableProduct

- *The following elements can be used wherever this element is referenced:*
  - quotableFxSingleLeg

Name	quotableProduct
Used by (from the same schema document)	Complex Type **QuoteUpdated** , Complex Type **RequestQuote**
Type	QuotableProduct
Nillable	no
Abstract	yes

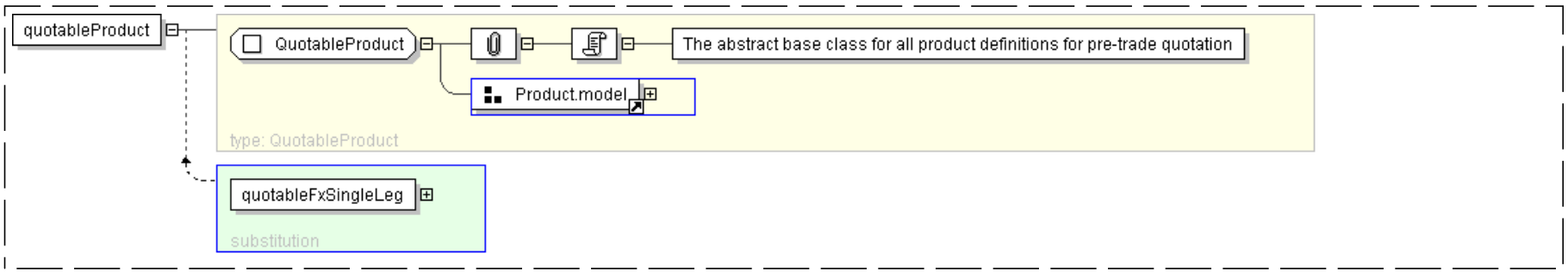
Logical Diagram



XML Instance Representation

```
<quotableProduct>
   <productType> ProductType </productType> [0..*]
   'A classification of the type of product. FpML defines a simple product categorization using
   a coding scheme.'

   <productId> ProductId </productId> [0..*]
   'A product reference identifier allocated by a party. FpML does not define the domain
   values associated with this element. Note that the domain values for this element are
   not strictly an enumerated list.'

</quotableProduct>
```

Diagram



Schema Component Representation

```
<xsd:element name="quotableProduct" type=" QuotableProduct " abstract="true"/>
```

top

Global Definitions

Complex Type: AcceptQuote

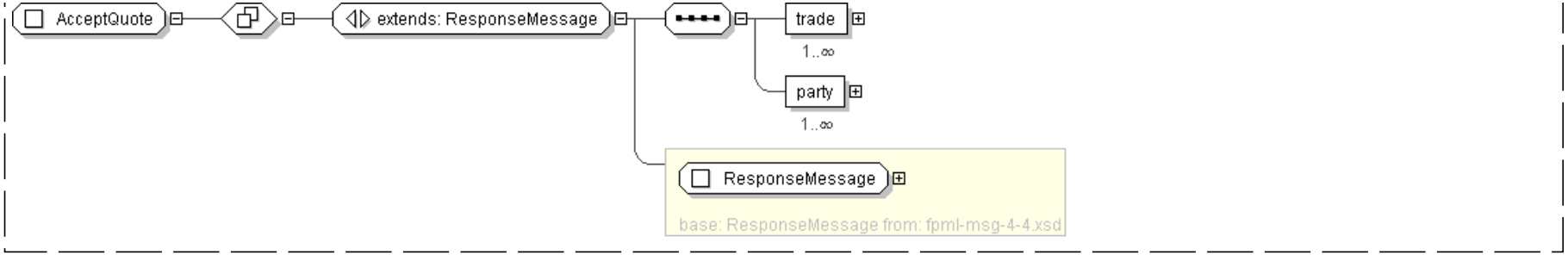
Super-types:	ResponseMessage < **AcceptQuote** (by extension)
Sub-types:	None

Name	AcceptQuote
Abstract	no

XML Instance Representation

```
<...
version=" xsd:token (value comes from list: {'4-0'|'4-1'|'4-2'|'4-3'|'4-4'}) [1]
'Indicate which version of the FpML Schema an FpML message adheres to.'

"
expectedBuild=" xsd:positiveInteger [0..1]
'This optional attribute can be supplied by a message creator in an FpML instance to
specify which build number of the schema was used to define the message when it was generated.'

"
actualBuild="2 [0..1]
'The specific build number of this schema version. This attribute is not included in
an instance document. Instead, it is supplied by the XML parser when the document is
validated against the FpML schema and indicates the build number of the schema file. Every
time FpML publishes a change to the schema, validation rules, or examples within a version
(e.g., version 4.2) the actual build number is incremented. If no changes have been
made between releases within a version (i.e. from Trial Recommendation to Recommendation)
the actual build number stays the same.'

">
   <header> ResponseMessageHeader </header> [1]
   <validation> Validation </validation> [0..*]
   <trade> Trade </trade> [1..*]
   <party> Party </party> [1..*]
</...>
```

Diagram



Schema Component Representation

```
<xsd:complexType name="AcceptQuote">
   <xsd:complexContent>
      <xsd:extension base=" ResponseMessage ">
         <xsd:sequence>
            <xsd:element name="trade" type=" Trade " maxOccurs="unbounded"/>
            <xsd:element name="party" type=" Party " maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
```

top

Complex Type: QuotableFxLeg

Super-types:	QuotableProduct < **QuotableFxLeg** (by extension)
Sub-types:	None

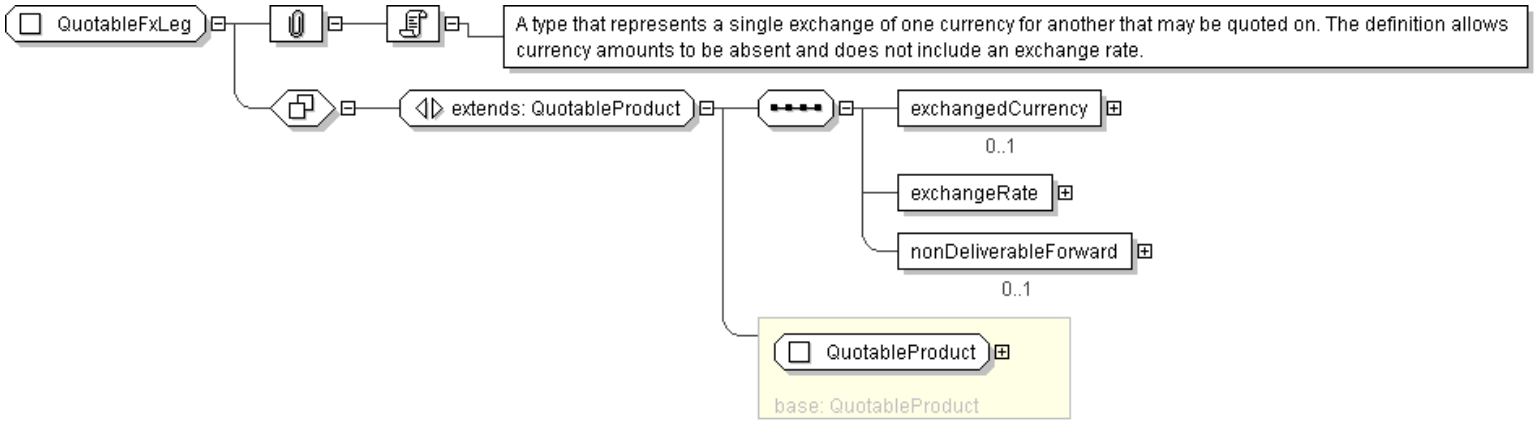
Name	QuotableFxLeg
Used by (from the same schema document)	Element **quotableFxSingleLeg**
Abstract	no
Documentation	A type that represents a single exchange of one currency for another that may be quoted on. The definition allows currency amounts to be absent and does not include an exchange rate.

XML Instance Representation

```
<...>
   <productType> ProductType </productType> [0..*]
   'A classification of the type of product. FpML defines a simple product categorization using
   a coding scheme.'

   <productId> ProductId </productId> [0..*]
   'A product reference identifier allocated by a party. FpML does not define the domain
   values associated with this element. Note that the domain values for this element are
   not strictly an enumerated list.'

   <exchangedCurrency> QuotablePayment </exchangedCurrency> [0..1]
   <exchangeRate> QuotableFxRate </exchangeRate> [1]
   <nonDeliverableForward> FxCashSettlement </nonDeliverableForward> [0..1]
   'Used to describe a particular type of FX forward transaction that is settled in a
   single currency.'
```

```
</...>
```

Diagram



Schema Component Representation

```
<xsd:complexType name="QuotableFxLeg">
   <xsd:complexContent>
      <xsd:extension base=" QuotableProduct ">
         <xsd:sequence>
            <xsd:element name="exchangedCurrency" type=" QuotablePayment " minOccurs="0"/>
            <xsd:element name="exchangeRate" type=" QuotableFxRate "/>
            <xsd:element name="nonDeliverableForward" type=" FxCashSettlement " minOccurs="0"/>
         </xsd:sequence>
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
```

top

Complex Type: QuotableFxRate

Super-types:	None
Sub-types:	None

Name	QuotableFxRate
Used by (from the same schema document)	Complex Type **QuotableFxLeg**
Abstract	no

XML Instance Representation

```
<...>
   <quotedCurrencyPair> QuotedCurrencyPair </quotedCurrencyPair> [1]
</...>
```

Diagram

Schema Component Representation

```
<xsd:complexType name="QuotableFxRate">
   <xsd:sequence>
      <xsd:element name="quotedCurrencyPair" type=" QuotedCurrencyPair "/>
   </xsd:sequence>
</xsd:complexType>
```

top

Complex Type: QuotablePayment

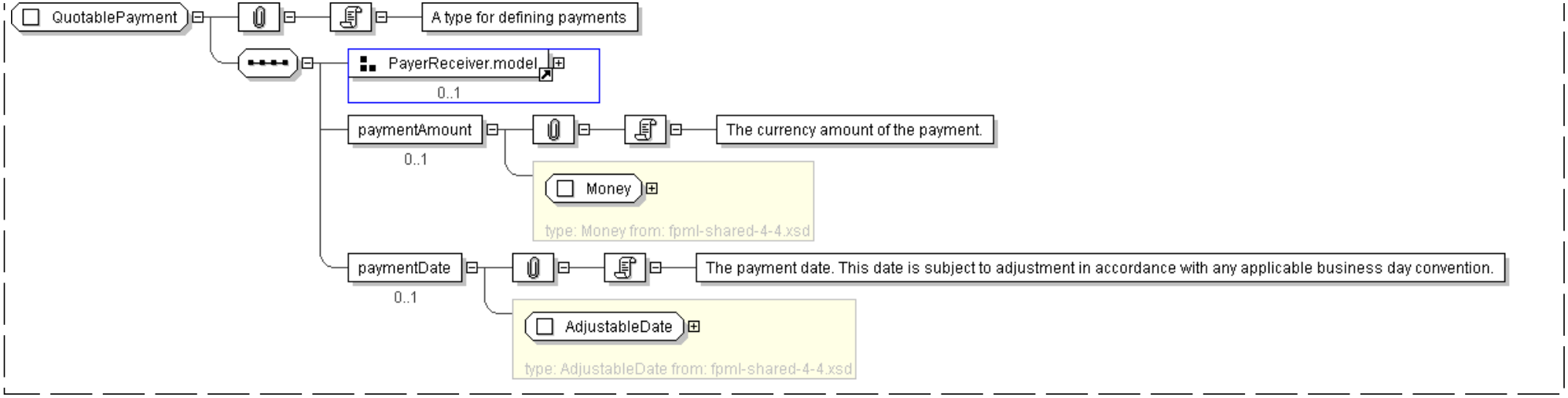
Super-types:	None
Sub-types:	None

Name	QuotablePayment
Used by (from the same schema document)	Complex Type **QuotableFxLeg**
Abstract	no
Documentation	A type for defining payments

XML Instance Representation

```
<...>
Start Group: PayerReceiver.model [0..1]
      <payerPartyReference> PartyOrAccountReference </payerPartyReference> [1]
      'A reference to the party responsible for making the payments defined by this structure.'

      <receiverPartyReference> PartyOrAccountReference </receiverPartyReference> [1]
      'A reference to the party that receives the payments corresponding to this structure.'

End Group: PayerReceiver.model
   <paymentAmount> Money </paymentAmount> [0..1]
   'The currency amount of the payment.'

   <paymentDate> AdjustableDate </paymentDate> [0..1]
   'The payment date. This date is subject to adjustment in accordance with any
   applicable business day convention.'

</...>
```

Diagram



Schema Component Representation

```
<xsd:complexType name="QuotablePayment">
   <xsd:sequence>
      <xsd:group ref=" PayerReceiver.model " minOccurs="0"/>
      <xsd:element name="paymentAmount" type=" Money " minOccurs="0"/>
      <xsd:element name="paymentDate" type=" AdjustableDate " minOccurs="0"/>
   </xsd:sequence>
</xsd:complexType>
```

top

Complex Type: QuotableProduct

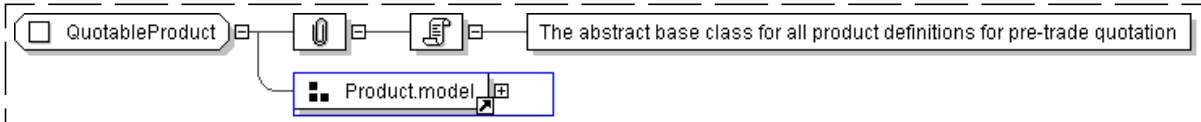
Super-types:	None
Sub-types:	• QuotableFxLeg (by extension)

Name	QuotableProduct
Used by (from the same schema document)	Element **quotableProduct**
Abstract	yes
Documentation	The abstract base class for all product definitions for pre-trade quotation

XML Instance Representation

```
<...>
   <productType> ProductType </productType> [0..*]
   'A classification of the type of product. FpML defines a simple product categorization using
   a coding scheme.'

   <productId> ProductId </productId> [0..*]
   'A product reference identifier allocated by a party. FpML does not define the domain
   values associated with this element. Note that the domain values for this element are
   not strictly an enumerated list.'

</...>
```

Diagram



Schema Component Representation

```
<xsd:complexType name="QuotableProduct" abstract="true">
   <xsd:group ref=" Product.model "/>
</xsd:complexType>
```

top

Complex Type: QuoteAcceptanceConfirmed

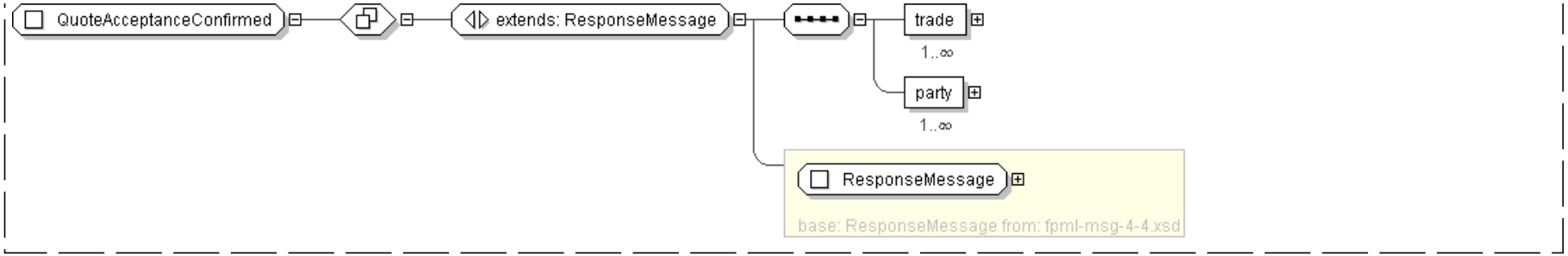
Super-types:	ResponseMessage < **QuoteAcceptanceConfirmed** (by extension)
Sub-types:	None

Name	QuoteAcceptanceConfirmed
Abstract	no

XML Instance Representation

```
<...
version=" xsd:token (value comes from list: {'4-0'|'4-1'|'4-2'|'4-3'|'4-4'}) [1]
'Indicate which version of the FpML Schema an FpML message adheres to.'

"
expectedBuild=" xsd:positiveInteger [0..1]
'This optional attribute can be supplied by a message creator in an FpML instance to
specify which build number of the schema was used to define the message when it was generated.'

"
actualBuild="2 [0..1]
'The specific build number of this schema version. This attribute is not included in
an instance document. Instead, it is supplied by the XML parser when the document is
validated against the FpML schema and indicates the build number of the schema file. Every
time FpML publishes a change to the schema, validation rules, or examples within a version
(e.g., version 4.2) the actual build number is incremented. If no changes have been
made between releases within a version (i.e. from Trial Recommendation to Recommendation)
the actual build number stays the same.'

">
   <header> ResponseMessageHeader </header> [1]
   <validation> Validation </validation> [0..*]
   <trade> Trade </trade> [1..*]
   <party> Party </party> [1..*]
</...>
```

Diagram



Schema Component Representation

```
<xsd:complexType name="QuoteAcceptanceConfirmed">
   <xsd:complexContent>
      <xsd:extension base=" ResponseMessage ">
         <xsd:sequence>
            <xsd:element name="trade" type=" Trade " maxOccurs="unbounded"/>
            <xsd:element name="party" type=" Party " maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
```

top

Complex Type: QuoteAlreadyExpired

Super-types:	ResponseMessage < **QuoteAlreadyExpired** (by extension)
Sub-types:	None

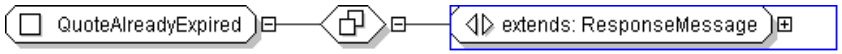
Name	QuoteAlreadyExpired
Abstract	no

XML Instance Representation

```
<...
version=" xsd:token (value comes from list: {'4-0'|'4-1'|'4-2'|'4-3'|'4-4'}) [1]
'Indicate which version of the FpML Schema an FpML message adheres to.'

"
expectedBuild=" xsd:positiveInteger [0..1]
'This optional attribute can be supplied by a message creator in an FpML instance to
specify which build number of the schema was used to define the message when it was generated.'

"
actualBuild="2 [0..1]
'The specific build number of this schema version. This attribute is not included in
an instance document. Instead, it is supplied by the XML parser when the document is
validated against the FpML schema and indicates the build number of the schema file. Every
time FpML publishes a change to the schema, validation rules, or examples within a version
(e.g., version 4.2) the actual build number is incremented. If no changes have been
made between releases within a version (i.e. from Trial Recommendation to Recommendation)
the actual build number stays the same.'
```

```
">
   <header> ResponseMessageHeader </header> [1]
   <validation> Validation </validation> [0..*]
</...>
```

Diagram



Schema Component Representation

```
<xsd:complexType name="QuoteAlreadyExpired">
   <xsd:complexContent>
      <xsd:extension base=" ResponseMessage "/>
   </xsd:complexContent>
</xsd:complexType>
```

top

Complex Type: QuoteUpdated

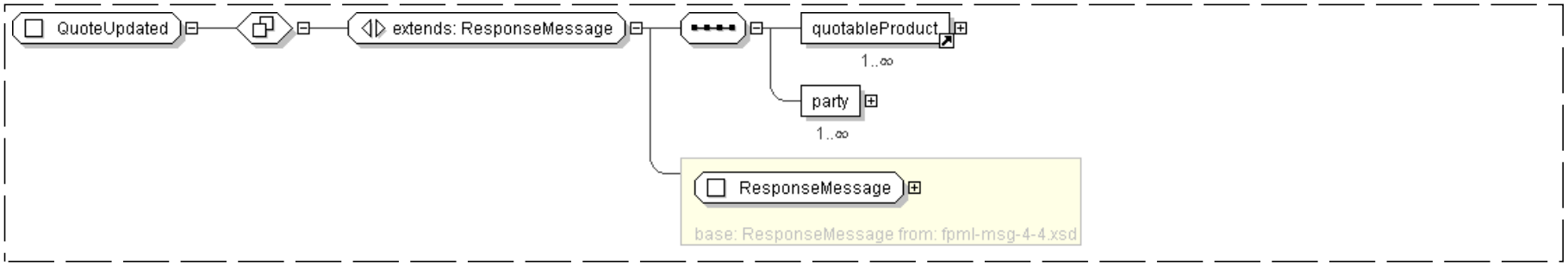
Super-types:	ResponseMessage < **QuoteUpdated** (by extension)
Sub-types:	None

Name	QuoteUpdated
Abstract	no

XML Instance Representation

```
<...
version=" xsd:token (value comes from list: {'4-0'|'4-1'|'4-2'|'4-3'|'4-4'}) [1]
'Indicate which version of the FpML Schema an FpML message adheres to.'

"
expectedBuild=" xsd:positiveInteger [0..1]
'This optional attribute can be supplied by a message creator in an FpML instance to
specify which build number of the schema was used to define the message when it was generated.'

"
actualBuild="2 [0..1]
'The specific build number of this schema version. This attribute is not included in
an instance document. Instead, it is supplied by the XML parser when the document is
validated against the FpML schema and indicates the build number of the schema file. Every
time FpML publishes a change to the schema, validation rules, or examples within a version
(e.g., version 4.2) the actual build number is incremented. If no changes have been
made between releases within a version (i.e. from Trial Recommendation to Recommendation)
the actual build number stays the same.'

">
   <header> ResponseMessageHeader </header> [1]
   <validation> Validation </validation> [0..*]
   <quotableProduct> ... </quotableProduct> [1..*]
   <party> Party </party> [1..*]
</...>
```

Diagram



Schema Component Representation

```
<xsd:complexType name="QuoteUpdated">
   <xsd:complexContent>
      <xsd:extension base=" ResponseMessage ">
         <xsd:sequence>
            <xsd:element ref=" quotableProduct " maxOccurs="unbounded"/>
            <xsd:element name="party" type=" Party " maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
```

top

Complex Type: RequestQuote

Super-types:	RequestMessage < **RequestQuote** (by extension)
Sub-types:	None

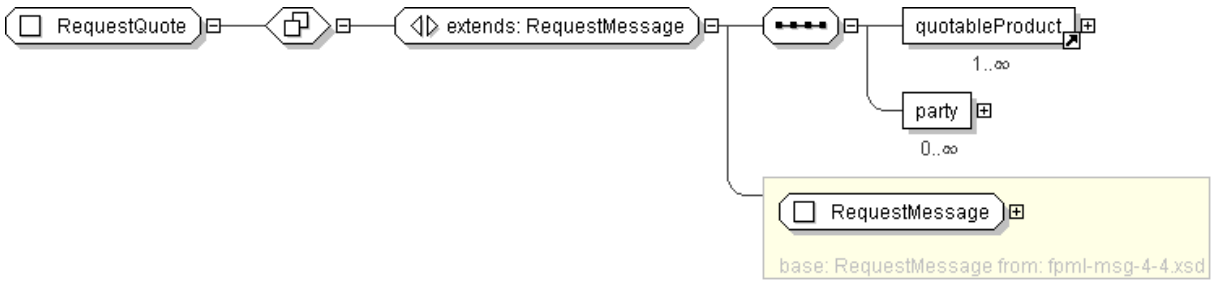
Name	RequestQuote
Abstract	no

XML Instance Representation

```
<...
version=" xsd:token (value comes from list: {'4-0'|'4-1'|'4-2'|'4-3'|'4-4'}) [1]
'Indicate which version of the FpML Schema an FpML message adheres to.'

"
expectedBuild=" xsd:positiveInteger [0..1]
'This optional attribute can be supplied by a message creator in an FpML instance to
specify which build number of the schema was used to define the message when it was generated.'

"
actualBuild="2 [0..1]
'The specific build number of this schema version. This attribute is not included in
an instance document. Instead, it is supplied by the XML parser when the document is
validated against the FpML schema and indicates the build number of the schema file. Every
time FpML publishes a change to the schema, validation rules, or examples within a version
(e.g., version 4.2) the actual build number is incremented. If no changes have been
made between releases within a version (i.e. from Trial Recommendation to Recommendation)
```

```
the actual build number stays the same.'

">
   <header> RequestMessageHeader </header> [1]
   <validation> Validation </validation> [0..*]
   <quotableProduct> ... </quotableProduct> [1..*]
   <party> Party </party> [0..*]
</...>
```

Diagram



Schema Component Representation

```
<xsd:complexType name="RequestQuote">
   <xsd:complexContent>
      <xsd:extension base=" RequestMessage ">
         <xsd:sequence>
            <xsd:element ref=" quotableProduct " maxOccurs="unbounded"/>
            <xsd:element name="party" type=" Party " minOccurs="0" maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
```

top

Complex Type: RequestQuoteResponse

Super-types:	ResponseMessage < **RequestQuoteResponse** (by extension)
Sub-types:	None

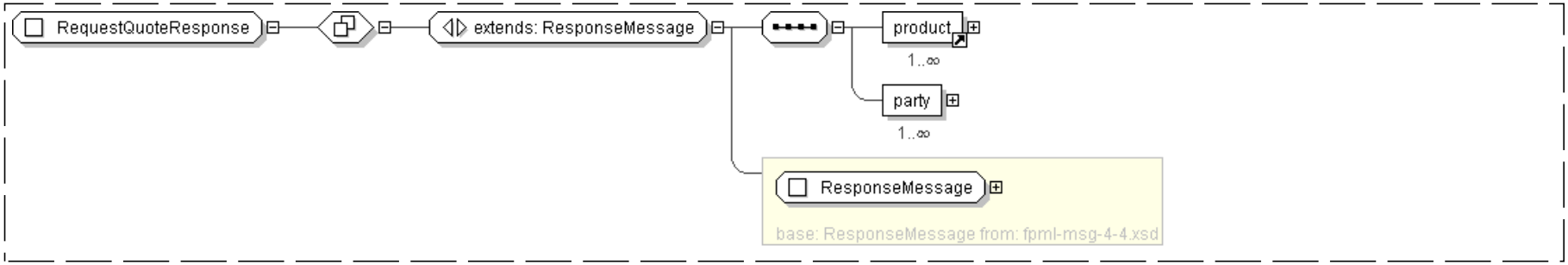
Name	RequestQuoteResponse
Abstract	no

XML Instance Representation

```
<...
version=" xsd:token (value comes from list: {'4-0'|'4-1'|'4-2'|'4-3'|'4-4'}) [1]
'Indicate which version of the FpML Schema an FpML message adheres to.'

"
expectedBuild=" xsd:positiveInteger [0..1]
'This optional attribute can be supplied by a message creator in an FpML instance to
specify which build number of the schema was used to define the message when it was generated.'
```

```
"
actualBuild="2 [0..1]
'The specific build number of this schema version. This attribute is not included in
an instance document. Instead, it is supplied by the XML parser when the document is
validated against the FpML schema and indicates the build number of the schema file. Every
time FpML publishes a change to the schema, validation rules, or examples within a version
(e.g., version 4.2) the actual build number is incremented. If no changes have been
made between releases within a version (i.e. from Trial Recommendation to Recommendation)
the actual build number stays the same.'

">
   <header> ResponseMessageHeader </header> [1]
   <validation> Validation </validation> [0..*]
   <product> ... </product> [1..*]
   <party> Party </party> [1..*]
</...>
```

Diagram



Schema Component Representation

```
<xsd:complexType name="RequestQuoteResponse">
   <xsd:complexContent>
      <xsd:extension base=" ResponseMessage ">
         <xsd:sequence>
            <xsd:element ref=" product " maxOccurs="unbounded"/>
            <xsd:element name="party" type=" Party " maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
```

top

Legend

Complex Type: Schema Component Type

AusAddress Schema Component Name

Super-types:	Address < AusAddress (by extension)
Sub-types:	• QLDAddress (by restriction)

If this schema component is a type definition, its type hierarchy is shown in a gray-bordered box.

Name	AusAddress
Abstract	no

The table above displays the properties of this schema component.

XML Instance Representation

```
<... country="Australia" >
<unitNo> string </unitNo> [0..1]
<houseNo> string </houseNo> [1]
<street> string </street> [1]
Start Choice [1]
<city> string </city> [1]
<town> string </town> [1]
End Choice
<state> AusStates </state> [1]
<postcode> string <<pattern = [1-9][0-9]{3}>> </postcode> [1]
</...>
```

The XML Instance Representation table above shows the schema component's content as an XML instance.

- The minimum and maximum occurrence of elements and attributes are provided in square brackets, e.g. [0..1].
- Model group information are shown in gray, e.g. Start Choice ... End Choice.
- For type derivations, the elements and attributes that have been added to or changed from the base type's content are shown in **bold**.
- If an element/attribute has a fixed value, the fixed value is shown in green, e.g. country="Australia".
- Otherwise, the type of the element/attribute is displayed.
  - If the element/attribute's type is in the schema, a link is provided to it.
  - For local simple type definitions, the constraints are displayed in angle brackets, e.g. <<*pattern* = [1-9][0-9]{3}>>.

Schema Component Representation

```
<complexType name="AusAddress">
<complexContent>
<extension base=" Address ">
<sequence>
<element name="state" type=" AusStates "/>
<element name="postcode">
<simpleType>
<restriction base=" string ">
<pattern value="[1-9][0-9]{3}"/>
</restriction>
</simpleType>
</element>
</sequence>
<attribute name="country" type=" string " fixed="Australia"/>
</extension>
</complexContent>
</complexType>
```

The Schema Component Representation table above displays the underlying XML representation of the schema component. (Annotations are not shown.)

top

Glossary

Abstract (Applies to complex type definitions and element declarations). An abstract element or complex type cannot used to validate an element instance. If there is a reference to an abstract element, only element declarations that can substitute the abstract element can be used to validate the instance. For references to abstract type definitions, only derived types can be used.

All Model Group Child elements can be provided *in any order* in instances. See: http://www.w3.org/TR/xmlschema-1/#element-all.

Choice Model Group *Only one* from the list of child elements and model groups can be provided in instances. See: http://www.w3.org/TR/xmlschema-1/#element-choice.

Collapse Whitespace Policy Replace tab, line feed, and carriage return characters with space character (Unicode character 32). Then, collapse contiguous sequences of space characters into single space character, and remove leading and trailing space characters.

Disallowed Substitutions (Applies to element declarations). If *substitution* is specified, then substitution group members cannot be used in place of the given element declaration to validate element instances. If *derivation methods*, e.g. extension, restriction, are specified, then the given element declaration will not validate element instances that have types derived from the element declaration's type using the specified derivation methods. Normally, element instances can override their declaration's type by specifying an `xsi:type` attribute.

Key Constraint Like Uniqueness Constraint, but additionally requires that the specified value(s) must be provided. See: http://www.w3.org/TR/xmlschema-1/#cIdentity-constraint_Definitions.

Key Reference Constraint Ensures that the specified value(s) must match value(s) from a Key Constraint or Uniqueness Constraint. See: http://www.w3.org/TR/xmlschema-1/#cIdentity-constraint_Definitions.

Model Group Groups together element content, specifying the order in which the element content can occur and the number of times the group of element content may be repeated. See: http://www.w3.org/TR/xmlschema-1/#Model_Groups.

Nillable (Applies to element declarations). If an element declaration is nillable, instances can use the `xsi:nil` attribute. The `xsi:nil` attribute is the boolean attribute, *nil*, from the *http://www.w3.org/2001/XMLSchema-instance* namespace. If an element instance has an `xsi:nil` attribute set to true, it can be left empty, even though its element declaration may have required content.

Notation A notation is used to identify the format of a piece of data. Values of elements and attributes that are of type, NOTATION, must come from the names of declared notations. See: http://www.w3.org/TR/xmlschema-1/#cNotation_Declarations.

Preserve Whitespace Policy Preserve whitespaces exactly as they appear in instances.

Prohibited Derivations (Applies to type definitions). Derivation methods that cannot be used to create sub-types from a given type definition.

Prohibited Substitutions (Applies to complex type definitions). Prevents sub-types that have been derived using the specified derivation methods from validating element instances in place of the given type definition.

Replace Whitespace Policy Replace tab, line feed, and carriage return characters with space character (Unicode character 32).

Sequence Model Group Child elements and model groups must be provided *in the specified order* in instances. See: http://www.w3.org/TR/xmlschema-1/#element-sequence.

Substitution Group Elements that are *members* of a substitution group can be used wherever the *head* element of the substitution group is referenced.

Substitution Group Exclusions (Applies to element declarations). Prohibits element declarations from nominating themselves as being able to substitute a given element declaration, if they have types that are derived from the original element's type using the specified derivation methods.

Target Namespace The target namespace identifies the namespace that components in this schema belongs to. If no target namespace is provided, then the schema components do not belong to any namespace.

Uniqueness Constraint Ensures uniqueness of an element/attribute value, or a combination of values, within a specified scope. See: http://www.w3.

org/TR/xmlschema-1/#cIdentity-constraint_Definitions.

top

Generated by <oXygen/> XML Editor using a modified version of xs3p that adds schema diagrams and chunking support.