XML Schema Documentation

Table of Contents

- Schema Document Properties
- Global Declarations
  - Element: **portfolio**
  - Element: **queryPortfolio**
- Global Definitions
  - Complex Type: **PortfolioValuationItem**
  - Complex Type: **PositionReport**
  - Complex Type: **RequestPositionReport**
  - Complex Type: **RequestValuationReport**
  - Complex Type: **RequestedPositions**
  - Complex Type: **TradeValuationItem**
  - Complex Type: **ValuationReport**
- 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 ○ fpml-valuation-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:include schemaLocation="fpml-valuation-4-4.xsd"/>
   ...
</xsd:schema>
```

top

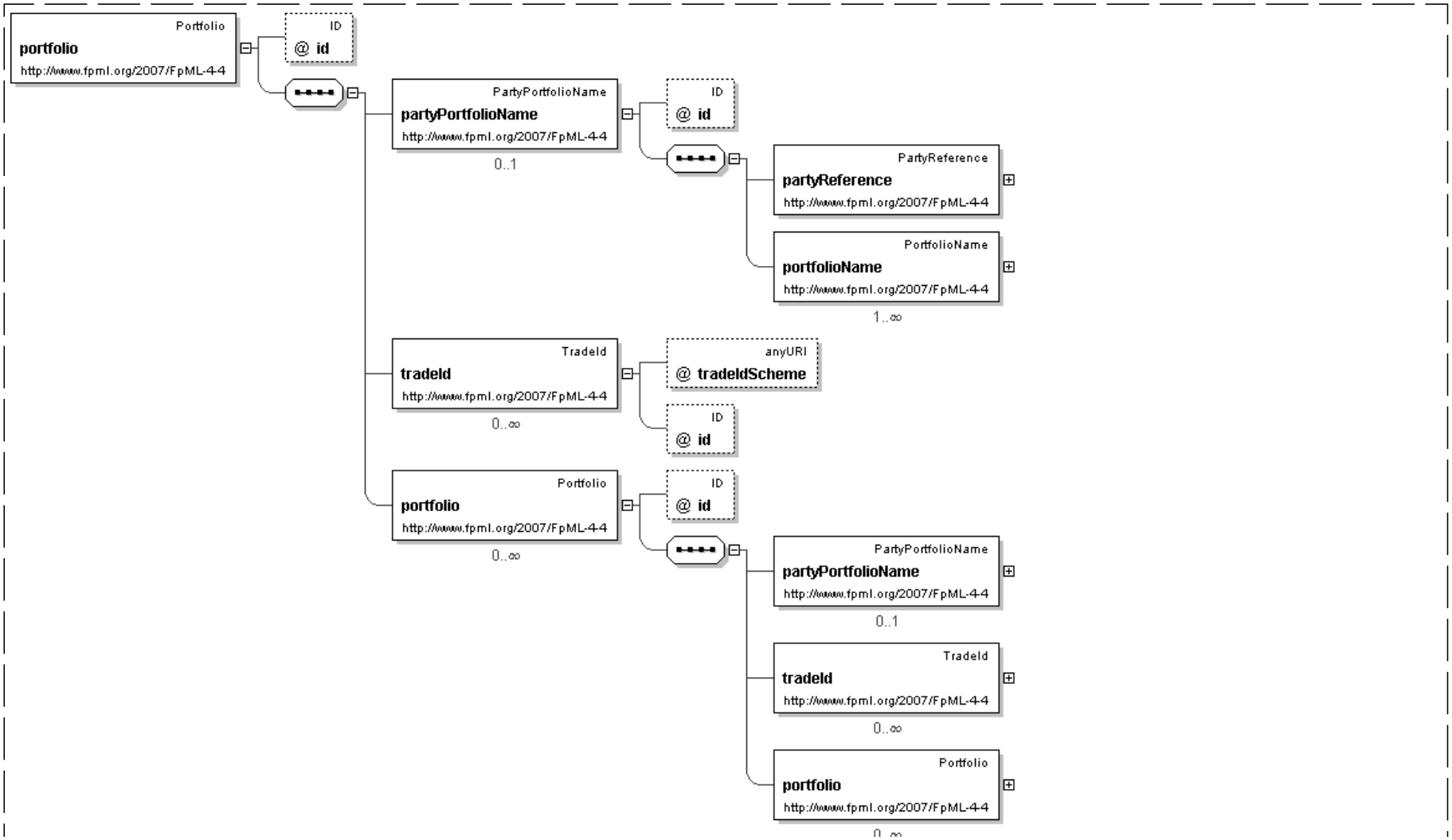
Global Declarations

Element: portfolio

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

Name	portfolio
Used by (from the same schema document)	Complex Type **PortfolioValuationItem**
Type	Portfolio
Nillable	no
Abstract	no
Documentation	Global portfolio element used as a basis for a substitution group.

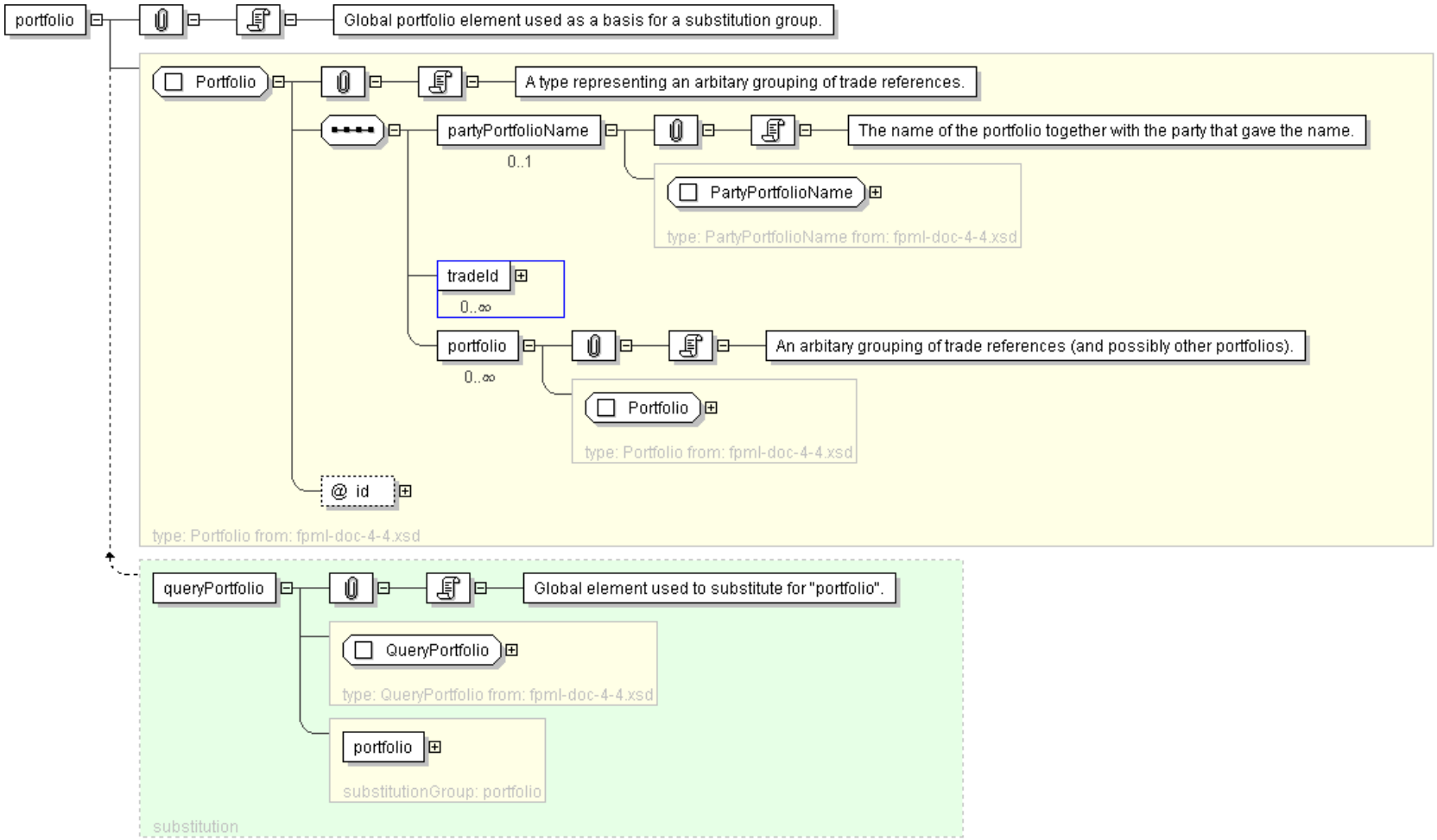
Logical Diagram



XML Instance Representation

```
<portfolio
id=" xsd:ID [0..1]">
   <partyPortfolioName> PartyPortfolioName </partyPortfolioName> [0..1]
   'The name of the portfolio together with the party that gave the name.'

   <tradeId> TradeId </tradeId> [0..*]
   <portfolio> Portfolio </portfolio> [0..*]
   'An arbitary grouping of trade references (and possibly other portfolios).'

</portfolio>
```

Diagram



Schema Component Representation

```
<xsd:element name="portfolio" type=" Portfolio "/>
```

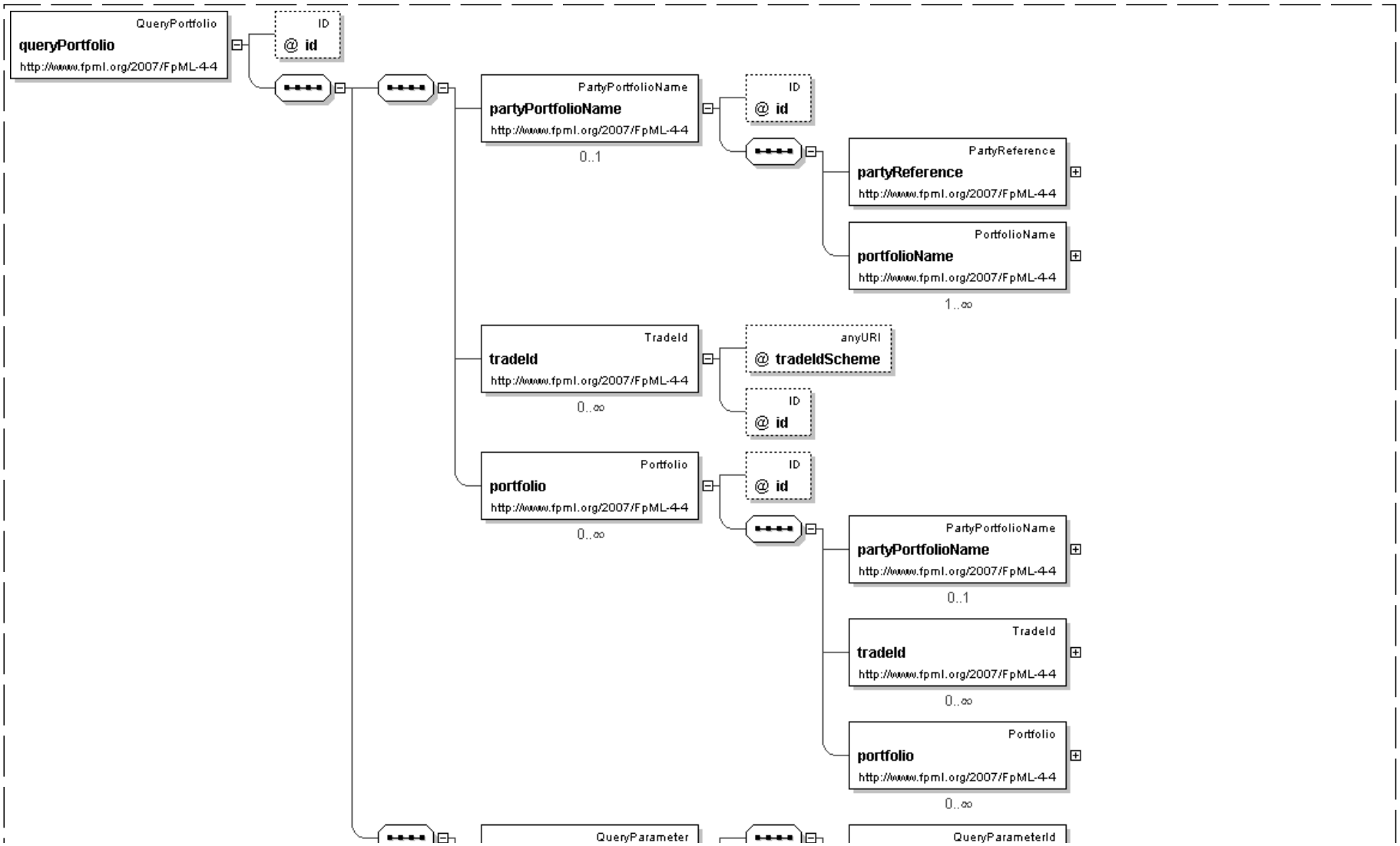
top

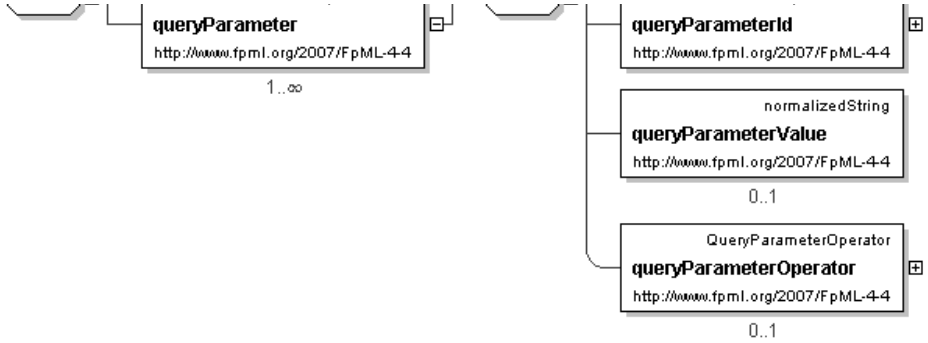
Element: queryPortfolio

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

Name	queryPortfolio
Type	QueryPortfolio
Nillable	no
Abstract	no
Documentation	Global element used to substitute for "portfolio".

Logical Diagram

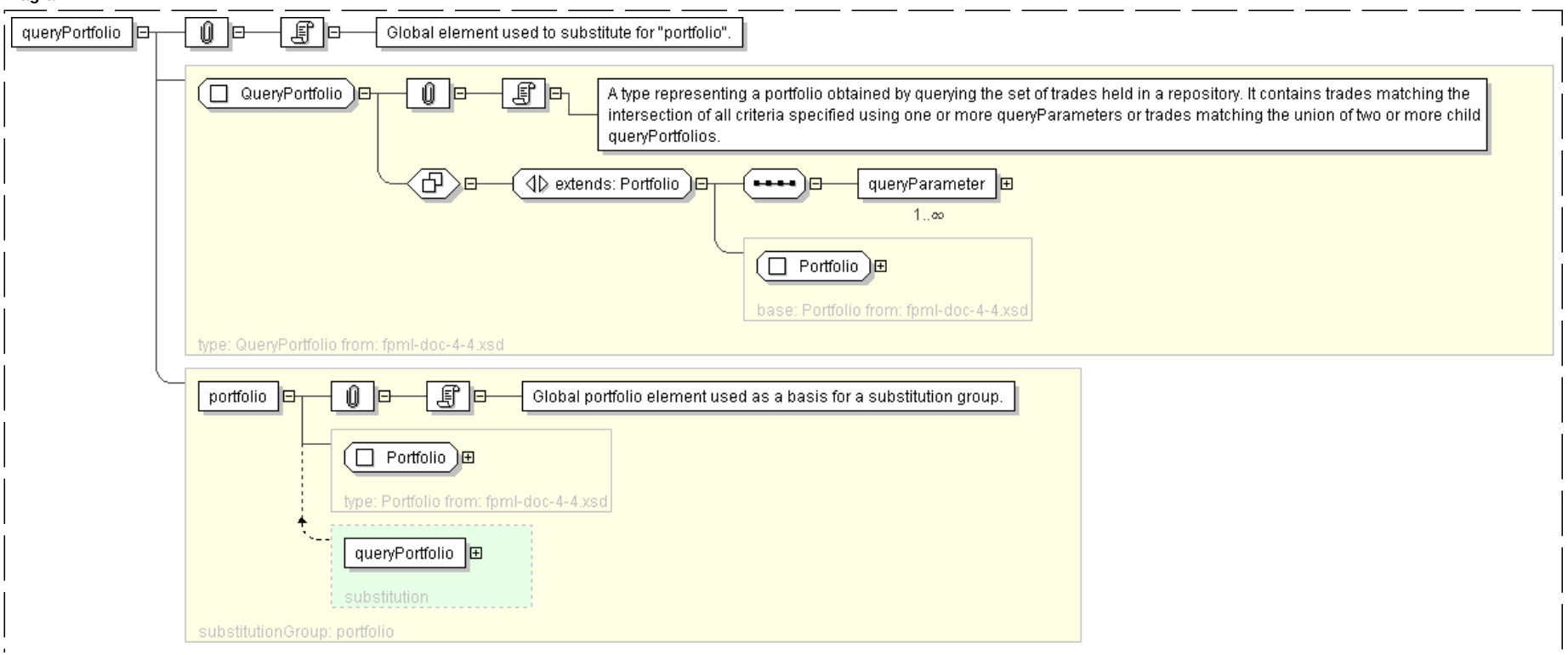




XML Instance Representation

```
<queryPortfolio
id=" xsd:ID [0..1]">
   <partyPortfolioName> PartyPortfolioName </partyPortfolioName> [0..1]
   'The name of the portfolio together with the party that gave the name.'

   <tradeId> TradeId </tradeId> [0..*]
   <portfolio> Portfolio </portfolio> [0..*]
   'An arbitary grouping of trade references (and possibly other portfolios).'

   <queryParameter> QueryParameter </queryParameter> [1..*]
</queryPortfolio>
```

Diagram



Schema Component Representation

```
<xsd:element name="queryPortfolio" type=" QueryPortfolio " substitutionGroup="portfolio"/>
```

top

Global Definitions

Complex Type: PortfolioValuationItem

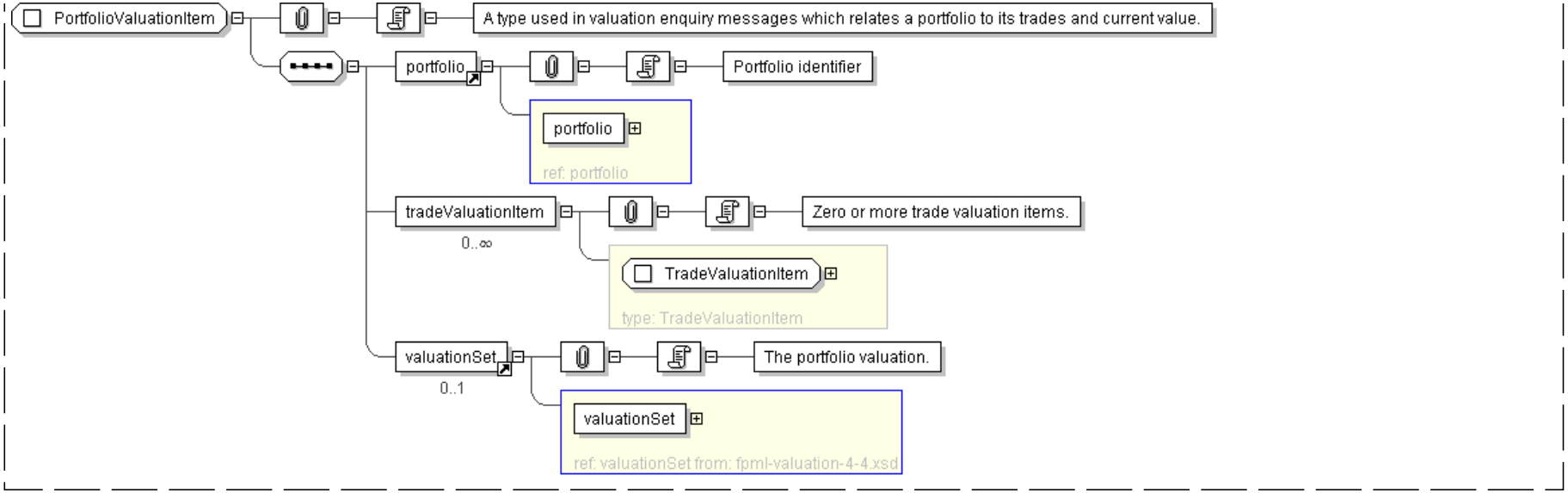
Super-types:	None
Sub-types:	None

Name	PortfolioValuationItem
Used by (from the same schema document)	Complex Type **RequestValuationReport** , Complex Type **ValuationReport**
Abstract	no
Documentation	A type used in valuation enquiry messages which relates a portfolio to its trades and current value.

XML Instance Representation

```
<...>
   <portfolio> ... </portfolio> [1]
   'Portfolio identifier'

   <tradeValuationItem> TradeValuationItem </tradeValuationItem> [0..*]
   'Zero or more trade valuation items.'

   <valuationSet> ... </valuationSet> [0..1]
   'The portfolio valuation.'

</...>
```

Diagram



Schema Component Representation

```
<xsd:complexType name="PortfolioValuationItem">
   <xsd:sequence>
      <xsd:element ref=" portfolio "/>
      <xsd:element name="tradeValuationItem" type=" TradeValuationItem "
      minOccurs="0" maxOccurs="unbounded"/>
      <xsd:element ref=" valuationSet " minOccurs="0"/>
   </xsd:sequence>
</xsd:complexType>
```

top

Complex Type: PositionReport

Super-types:	NotificationMessage < **PositionReport** (by extension)
Sub-types:	None

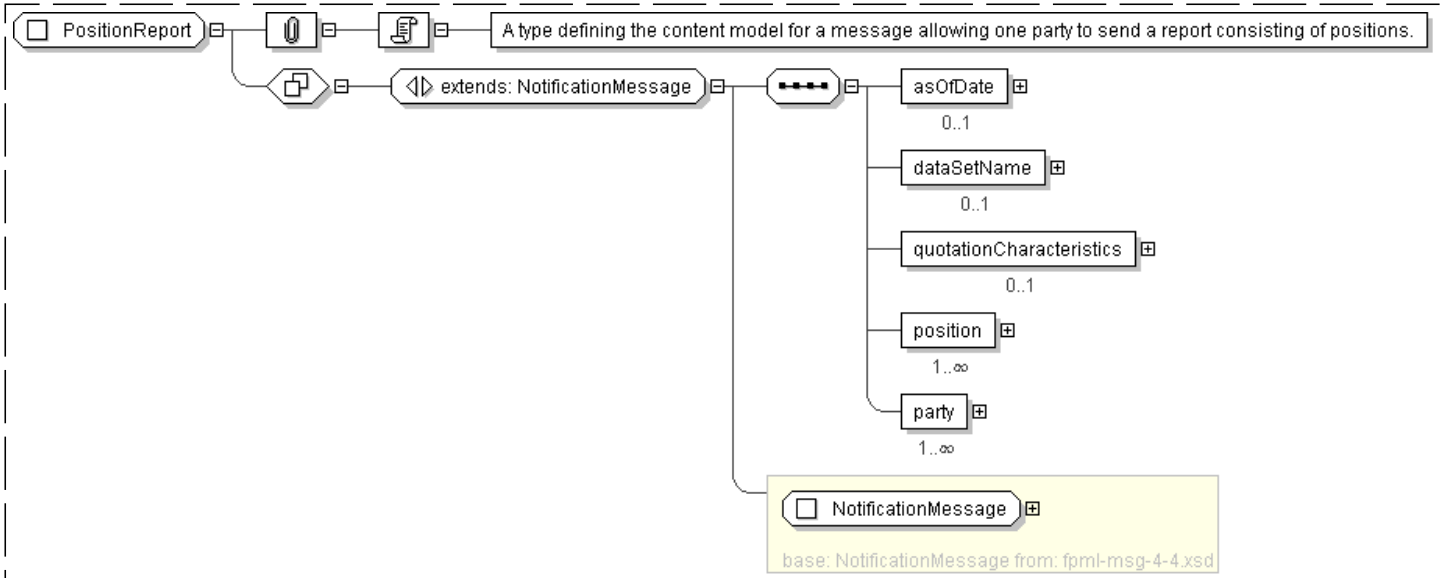
Name	PositionReport
Abstract	no
Documentation	A type defining the content model for a message allowing one party to send a report consisting of positions.

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> NotificationMessageHeader </header> [1]
   <validation> Validation </validation> [0..*]
   <asOfDate> IdentifiedDate </asOfDate> [0..1]
   'The date for which this document reports positions and valuations.'

   <dataSetName> xsd:string </dataSetName> [0..1]
   'The name of the data set (portfolio, product type, etc.) that this report corresponds to.
   Used to help document the contents of the report.'

   <quotationCharacteristics> QuotationCharacteristics </quotationCharacteristics> [0..1]
   'The default quotation characteristics for this document (e.g. currency, location).
   Currency must be specified; other fields may be specified.'

   <position> Position </position> [1..*]
   'The positions included in the position report.'

   <party> Party </party> [1..*]
   'The parties whose trades are included included in this position report.'

</...>
```

Diagram



Schema Component Representation

```
<xsd:complexType name="PositionReport">
```

```
   <xsd:complexContent>
      <xsd:extension base=" NotificationMessage ">
         <xsd:sequence>
            <xsd:element name="asOfDate" type=" IdentifiedDate " minOccurs="0"/>
            <xsd:element name="dataSetName" type=" xsd:string " minOccurs="0"/>
            <xsd:element name="quotationCharacteristics" type=" QuotationCharacteristics " minOccurs="0"/>
            <xsd:element name="position" type=" Position " maxOccurs="unbounded"/>
            <xsd:element name="party" type=" Party " maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
```

top

Complex Type: RequestPositionReport

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

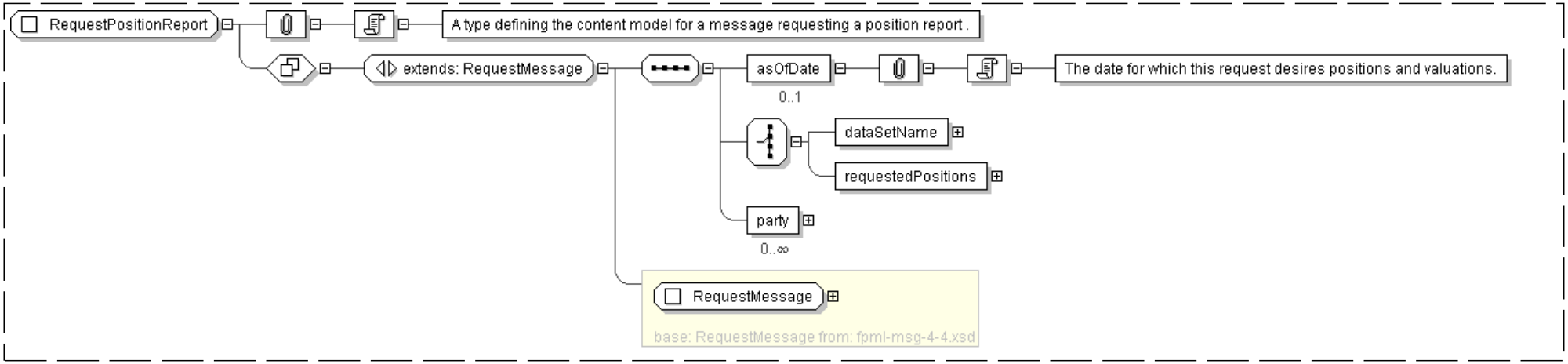
Name	RequestPositionReport
Abstract	no
Documentation	A type defining the content model for a message requesting a position report .

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..*]
   <asOfDate> ... </asOfDate> [0..1]
   'The date for which this request desires positions and valuations.'

Start Choice [1]
      <dataSetName> xsd:normalizedString </dataSetName> [1]
      'The name of the data set (portfolio, product type, etc.) that this request corresponds
      to. Describes the desired report.'

      <requestedPositions> RequestedPositions </requestedPositions> [1]
```

```
        'The name of the data set (portfolio, product type, etc.) that this request corresponds
        to. Describes the desired report.'

End Choice
   <party> Party </party> [0..*]
</...>
```

Diagram



Schema Component Representation

```xsd:complexType name="RequestPositionReport">
   <xsd:complexContent>
      <xsd:extension base=" RequestMessage ">
         <xsd:sequence>
            <xsd:element name="asOfDate" minOccurs="0"/>
            <xsd:choice>
               <xsd:element name="dataSetName" type=" xsd:normalizedString "/>
               <xsd:element name="requestedPositions" type=" RequestedPositions "/>
            </xsd:choice>
            <xsd:element name="party" type=" Party " minOccurs="0" maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
```

top

Complex Type: RequestValuationReport

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

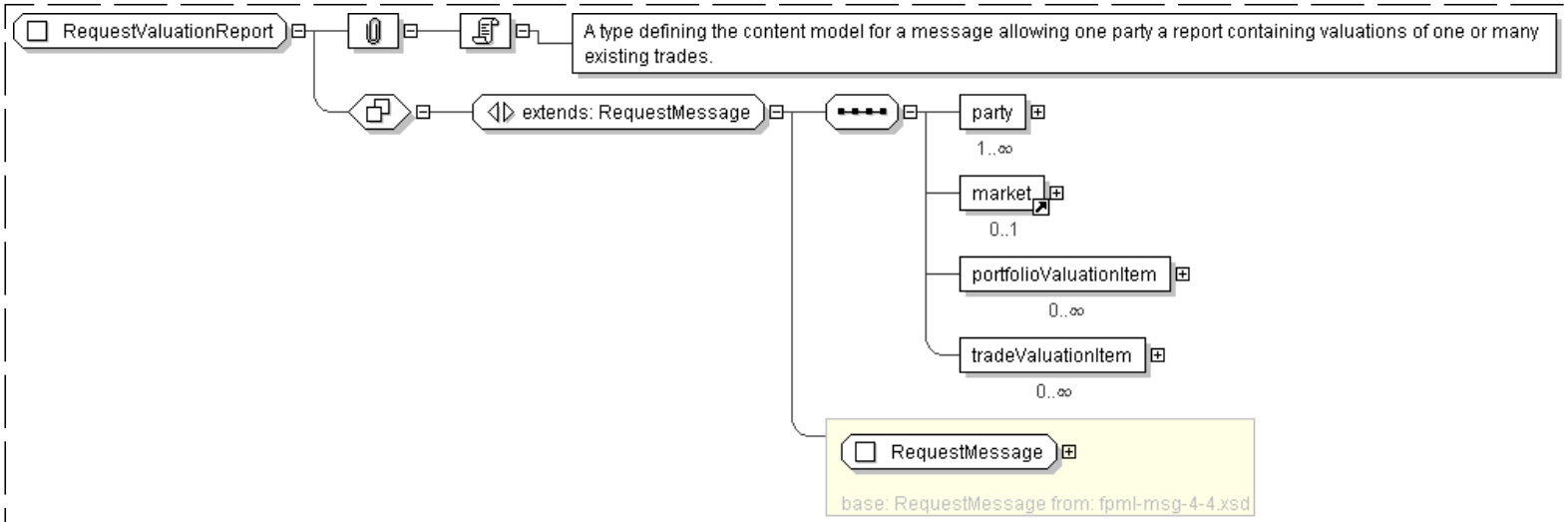
Name	RequestValuationReport
Abstract	no
Documentation	A type defining the content model for a message allowing one party a report containing valuations of one or many existing trades.

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..*]
   <party> Party </party> [1..*]
   <market> ... </market> [0..1]
   <portfolioValuationItem> PortfolioValuationItem </portfolioValuationItem> [0..*]
   'An instance of a unique portfolio valuation.'

   <tradeValuationItem> TradeValuationItem </tradeValuationItem> [0..*]
   'An instance of a unique trade valuation.'

</...>
```

Diagram



Schema Component Representation

```
<xsd:complexType name="RequestValuationReport">
   <xsd:complexContent>
      <xsd:extension base=" RequestMessage ">
```

```
        <xsd:sequence>
           <xsd:element name="party" type=" Party " maxOccurs="unbounded"/>
           <xsd:element ref=" market " minOccurs="0"/>
           <xsd:element name="portfolioValuationItem" type=" PortfolioValuationItem "
           minOccurs="0" maxOccurs="unbounded"/>
           <xsd:element name="tradeValuationItem" type=" TradeValuationItem "
           minOccurs="0" maxOccurs="unbounded"/>
        </xsd:sequence>
     </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
```

top

Complex Type: RequestedPositions

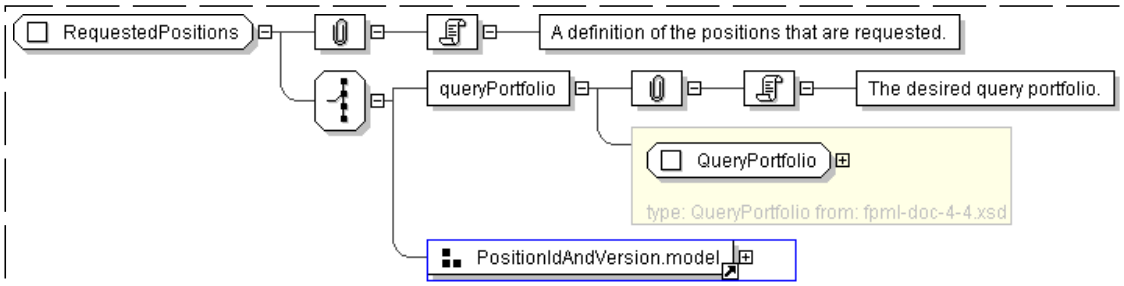
Super-types:	None
Sub-types:	None

Name	RequestedPositions
Used by (from the same schema document)	Complex Type **RequestPositionReport**
Abstract	no
Documentation	A definition of the positions that are requested.

XML Instance Representation

```
<...>
Start Choice [1]
      <queryPortfolio> QueryPortfolio </queryPortfolio> [1]
      'The desired query portfolio.'

      <positionId> PositionId </positionId> [1]
      'A version-independent identifier for the position, possibly based on trade identifier.'

      <version> xsd:positiveInteger </version> [0..1]
      'A version identifier. Version identifiers must be ascending, i.e. higher numbers imply
      newer versions. There is no requirement that version identifiers for a position be
      sequential or small, so for example timestamp-based version identifiers could be used.'

End Choice
</...>
```

Diagram



Schema Component Representation

```
<xsd:complexType name="RequestedPositions">
   <xsd:choice>
      <xsd:element name="queryPortfolio" type=" QueryPortfolio "/>
      <xsd:group ref=" PositionIdAndVersion.model "/>
   </xsd:choice>
</xsd:complexType>
```

top

Complex Type: TradeValuationItem

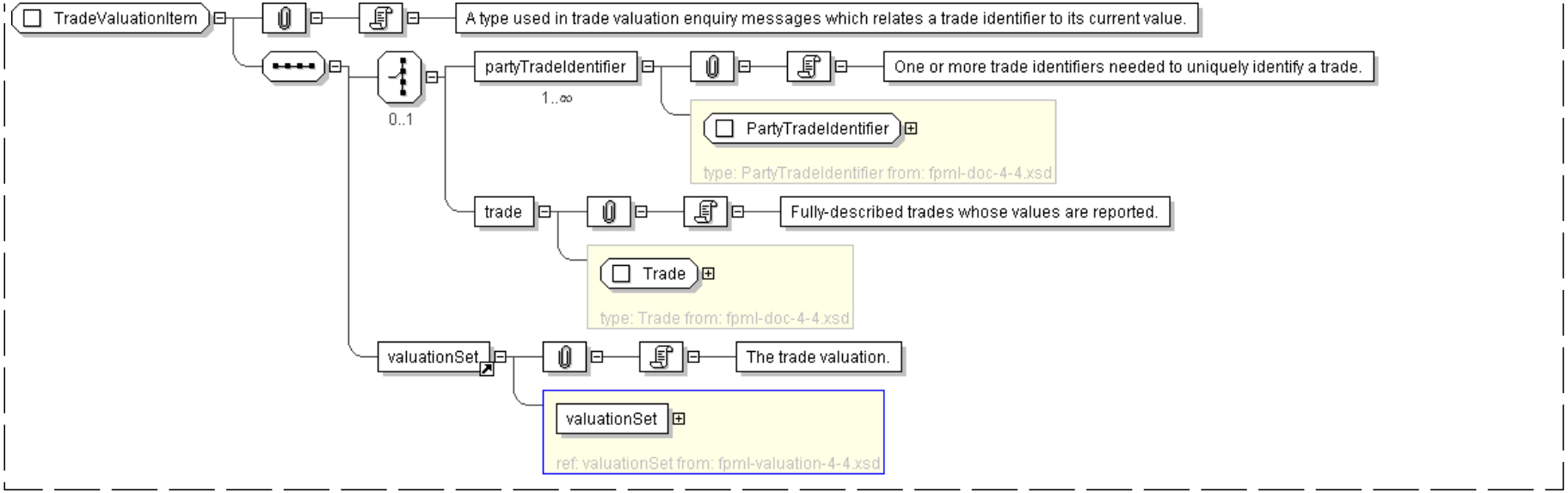
Super-types:	None
Sub-types:	None

Name	TradeValuationItem
Used by (from the same schema document)	Complex Type **PortfolioValuationItem** , Complex Type **RequestValuationReport** , Complex Type **ValuationReport**
Abstract	no
Documentation	A type used in trade valuation enquiry messages which relates a trade identifier to its current value.

XML Instance Representation

```
<...>
Start Choice [0..1]
      <partyTradeIdentifier> PartyTradeIdentifier </partyTradeIdentifier> [1..*]
      'One or more trade identifiers needed to uniquely identify a trade.'

      <trade> Trade </trade> [1]
      'Fully-described trades whose values are reported.'

End Choice
   <valuationSet> ... </valuationSet> [1]
   'The trade valuation.'

</...>
```

Diagram



Schema Component Representation

```
<xsd:complexType name="TradeValuationItem">
   <xsd:sequence>
      <xsd:choice minOccurs="0">
         <xsd:element name="partyTradeIdentifier" type=" PartyTradeIdentifier " maxOccurs="unbounded"/>
         <xsd:element name="trade" type=" Trade "/>
      </xsd:choice>
      <xsd:element ref=" valuationSet "/>
   </xsd:sequence>
</xsd:complexType>
```

top

Complex Type: ValuationReport

Super-types:	NotificationMessage < **ValuationReport** (by extension)
Sub-types:	None

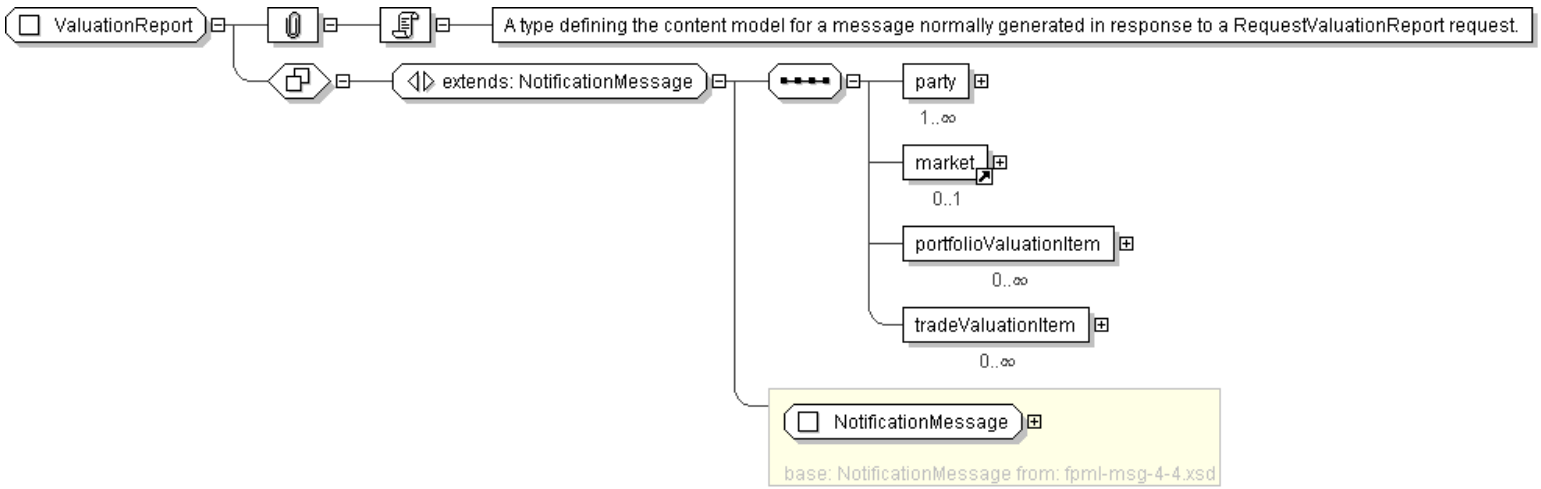
Name	ValuationReport
Abstract	no
Documentation	A type defining the content model for a message normally generated in response to a RequestValuationReport request.

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> NotificationMessageHeader </header> [1]
   <validation> Validation </validation> [0..*]
   <party> Party </party> [1..*]
   <market> ... </market> [0..1]
   <portfolioValuationItem> PortfolioValuationItem </portfolioValuationItem> [0..*]
   'An instance of a unique portfolio valuation.'

   <tradeValuationItem> TradeValuationItem </tradeValuationItem> [0..*]
   'A collection of data values describing the state of the given trade.'

</...>
```

Diagram



Schema Component Representation

```
<xsd:complexType name="ValuationReport">
   <xsd:complexContent>
      <xsd:extension base=" NotificationMessage ">
         <xsd:sequence>
            <xsd:element name="party" type=" Party " maxOccurs="unbounded"/>
            <xsd:element ref=" market " minOccurs="0"/>
            <xsd:element name="portfolioValuationItem" type=" PortfolioValuationItem "
            minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element name="tradeValuationItem" type=" TradeValuationItem "
            minOccurs="0" maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
```

top

Legend

Complex Type: **AusAddress**

Schema Component Type — 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.