AdsML® Framework for E-Commerce Business Standards for Advertising

AdsMLTypeLibrary 2.0.2

Specification & Schema

Document Authors: AdsML Technical Working Group

Document ID: AdsMLTypeLibrary-2.0.2-Specification-AS-1

Document File Name: AdsMLTypeLibrary-2.0-Specification-AS.pdf

Document Status: Approved Specification

Document Date: 15 April 2010

Draft Number: 1

Table of Contents

1 AdsML Standard Documentation

1.1 Document status and copyright

This is the Approved Specification of the AdsML Type Library Specification.

1.2 Non-Exclusive License Agreement for AdsML Consortium Specifications

termination you will cease all use duplication, distribution, and/or exploitation in any manner of the Specification.

9. This User License reflects the entire agreement of the parties regarding the subject matter hereof and supercedes all prior agreements or representations regarding such matters, whether written or oral. To the extent any portion or provision of this User License is found to be illegal or unenforceable, then the remaining provisions of this User License will remain in full force and effect and the illegal or unenforceable provision will be construed to give it such effect as it may properly have that is consistent with the intentions of the parties. This User License may only be modified in writing signed by an authorized representative of the AdsML Consortium. This User License will be governed by the law of Darmstadt (Federal Republic of Germany), as such law is applied to contracts made and fully performed in Darmstadt (Federal Republic of Germany). Any disputes arising from or relating to this User License will be resolved in the courts of Darmstadt (Federal Republic of Germany). You consent to the jurisdiction of such courts over you and covenant not to assert before such courts any objection to proceeding in such forums.

10. Except as expressly provided herein, you may not use the name of the AdsML Consortium, or any of its marks, for any purpose without the prior consent of an authorized representative of the owner of such name or mark.

IF YOU DO NOT AGREE TO THESE TERMS PLEASE CEASE ALL USE OF THIS SPECIFICATION NOW. IF YOU HAVE ANY QUESTIONS ABOUT THESE TERMS, PLEASE CONTACT THE SECRETARIAT OF THE ADSML CONSORTIUM.

AS OF THE DATE OF THIS REVISION OF THE SPECIFICATION YOU MAY CONTACT THE AdsML Consortium at www.adsml.org.

1.3 AdsML Code of Conduct

The AdsML Code of Conduct governs AdsML Consortium activities. A reading or reference to the AdsML Code of Conduct begins every AdsML activity, whether a meeting of the AdsML Consortium, AdsML Working Groups, or AdsML conference calls to resolve a technical issue. The AdsML Code of Conduct says:

Trade associations are perfectly lawful organizations. However, since a trade association is, by definition, an organization of competitors, AdsML Consortium members must take precautions to ensure that we do not engage in activities which can be interpreted as violating anti-trust or other unfair competition laws.

For any activity which is deemed to unreasonably restrain trade, AdsML, its members and individual representatives may be subject to severe legal penalties, regardless of our otherwise beneficial objectives. It is important to realize, therefore, that an action that may seem to make "good business sense" can injure competition and therefore be prohibited under the antitrust or unfair competition laws.

To ensure that we conduct all meetings and gatherings in strict compliance with any such laws and agreements in any part of the world, the AdsML Code of Conduct is to be distributed and/or read aloud at all such gatherings.

- There shall be no discussion of rates, fares, surcharges, conditions, terms or prices of services, allocating or sharing of customers, or refusing to deal with a particular supplier or class of suppliers. Neither serious nor flippant remarks about such subjects will be permitted.

- AdsML shall not issue recommendations about any of the above subjects or distribute to its members any publication concerning such matters. No discussions that directly or indirectly fix purchase or selling prices may take place.
- There shall be no discussions of members' marketing, pricing or service plans.
- All AdsML related meetings shall be conducted in accordance with a previously prepared and distributed agenda.
- If you are uncomfortable about the direction that you believe a discussion is heading, you should say so promptly.

Members may have varying views about issues that AdsML deals with. They are encouraged to express themselves in AdsML activities. However, official AdsML communications to the public are the sole responsibility of the AdsML Consortium. To avoid creating confusion among the public, therefore, the Steering Committee must approve press releases and any other forms of official AdsML communications to the public before they are released.

1.4 Document Number and Location

This document, AdsMLTypeLibrary-2.0.2-Specification-AS-1, is freely available. It is located in the members' area of the AdsML website at http://www.adsml.org/.

1.5 Purpose of this document

This document specifies the definition of the XML structures comprising the type library of components that are used by AdsML standards across the AdsML Framework.

1.6 Audience

The intended audience for this document is primarily user and vendor organizations who seek to implement the AdsML standards in their workflows, advertising systems, or software products. Those assessing the conformance of vendor products to the standard may also use the document.

Comments on this specification should be addressed to the AdsML Consortium and to the Technical Working Group of the AdsML Consortium (technical.wg@adsml.org).

1.7 Accompanying documents

The AdsML Type Library Specification is part of the AdsML Framework, which contains a suite of related documents. Readers of this document are assumed to be familiar with the full range of relevant AdsML documentation. In particular, readers are assumed to have read the *E-Commerce Usage Rules and Guidelines* document.

A description of the entire document set can be found in the *ReadMeFirst* html file associated with this release of the AdsML Framework.

1.8 Definitions & conventions

1.8.1 Definitions of key words used in the specification

The key words "**MUST**", "**MUST NOT**", "**REQUIRED**", "**SHALL**", "**SHALL NOT**", "**SHOULD**", "**SHOULD NOT**", "**RECOMMENDED**", "**MAY**", and "**OPTIONAL**" in this document are used as described in IETF RFC 2119.(S. Bradner. *Key words for use in RFCs to Indicate Requirement Levels*. Internet Engineering Task Force (IETF), Request for Comments: 2119, March 1997, http://www.ietf.org/rfc/rfc2119.txt)

The key word "**DEPRECATED**" is used to indicate that structures are being phased out of the AdsML specifications. Structures marked as **DEPRECATED** will be removed in the next major schema upgrade and should not be used in new implementations.

When any of these words do not appear in upper case as above, then they are being used with their usual English language sense and meaning.

1.8.2 Naming conventions – element, attribute, type, and file names

All element, attribute, and type names follow the 'CamelCase' convention.

Element and type names begin using upper camel case and begin with capitals (UpperCamelCase). For example, 'AdsML', 'MessageRef', and 'AdsMLStatusType'.

Attribute names begin using lower camel case and begin with lower case (*lowerCamelCase*). For example, '*language*' or '*messageId*'.

File names also follow the camel case convention and use upper camel case for each segment of the file name, plus dashes to separate the segments of the file name. Only the first two digits of the version number are included in the file name. The third digit of the version number (if there is one) and the Draft Number are only shown internally within the document. The full naming conventions for AdsML schema and specification file names are described in the document *AdsML Document Names and Identifiers – Guidelines and Examples*, a copy of which is included in this release of the Framework.

Schema for user-defined extensions to AdsML should use AdsML naming conventions as detailed above. For example, 'ExampleInstanceFile.xml', 'ExampleSchemaFile-1.0.xsd', 'ExampleSchemaFile-1.1.xsd'.

1.8.3 Typographical conventions

Element and type names are given in Courier font as, for example, AdOrder.

Attribute names are given in italicized Courier font as, for example, *messageCode*.

When citing examples of values that could be assigned to elements or attributes, the value is given in Courier font, so "…the attribute taking the value of '12'".

1.9 Change History

Version	Date	Changes	Author
2.0.2-1 AS	15 April 2010	Approved version of 2.0.2. Previous change history removed.	UW

Version	Date	Changes	Author
2.0.0-1	2007-10-10	First public release of version 2.0	UW, TS. JC
1.1.1-1	2006-06-01	First version of the complete specification of the Type Library.	UW, JC

1.9.1 Changes in version 2.0.2

1.9.1.1 New Structures

Support for multiple languages

The AdsML Type Library includes internationalization support by providing basic string types with attribute extensions to express language and reading directionality.

Existing elements have been updated to use the new internationalized types in a number of contexts, primarily elements that include human readable text. In order to support multiple language texts in any one context, changes have also been made to cardinalities allowing for instance repeatable `Description` elements with descriptive content in several alternative languages. See for instance the `CodeType` type.

Time durations

The `Duration` element is defined as a `DecimalMeasurementType`. It is used to capture a duration in time.

Support for Proofing

A set of elements usable in a proofing context have been added:

- The `ProofingParty` is a party that will distribute a proof (or has business responsibility for a proofing message as a whole)
- The `ProvenanceParty` is a party that takes responsibility for (parts of) proofing information, e.g. a physical tearsheet or affidavit.
- The `ProofersReference` can be used to express a reference string for a Proofing Party.

Terms and Conditions

The new `TermsAndConditionsDetails` element can be used to reference or include a document with human-readable terms and conditions.

Usage Label Codes

The element `UsageLabel` can be used to capture a code describing usage in any context.

1.9.1.1 Updated Structures

Repeatable Name of parties

The `Name` element in `PartyType`, `RelatedPartyType` and `RelaxedPartype` has been internationalized and made repeatable to support recording a party's name in alternative languages.

Repeatable Party Address in RelaxedPartType

The `PartyAddress` element has been made repeatable in `RelaxedPartyType` and `RelatedPartyType` to support recording addresses in alternative languages and to improve alignment with the `PartyType`.

Exchange rate specifications added to price declarations

The `CurrencyPriceDeclarationType` type has been extended with a new `ExchangeRate` element that expresses information about how currency conversion has been performed between two currencies.

The new element and all of its child elements have been moved without changes to the Type Library from the AdsML Financials schema where it was locally defined.

Price Components

The deprecated *`scheduleEntryReference`* attribute has been removed.

Extensible labeled properties with descriptions

The `LabeledValueType` has been extended with a repeatable `Description` element to capture a human readable explanation of codified values.

Title in ContactType

The `ContactType` type has been extended with a new `Title` element.

Nillable Base Price

The `BasePrice` element, used in the `CalculationSpecification`, is now nillable.

1.9.2 Changes in version 2.0.1

1.9.2.1 New Structures

Additional services

A new structure called `AdditionalService` for specification of generic so-called "additional services" has been defined.

1.9.2.2 Updated Structures

Price Components

The `NamedPriceType`, used by the `PriceComponent`, has two new elements, `ScheduleEntryReference` and `AdditionalServiceReference`, that allow a particular price to be associated with a schedule entry and/or an additional service defined elsewhere in the containing document.

Note that the `ScheduleEntryReference` element replaces the older *`scheduleEntryReference`* attribute that is now **DEPRECATED** and will be removed in the next major upgrade.

1.9.3 Changes in version 2.0

This version includes major new structures resulting from the development of AdsML Financials and AdsML Proof of Publication standards and updates to the AdsML Materials and AdsML Booking standards.

Note that the upgrade to version 2.0 includes a namespace change.

1.9.3.1 New structures

Generic specifications and types

A generic `Specifications` structure has been made available.

A generic type code structure is available as the `Type` element.

Price specifications

The `PriceDeclarationType`, originally developed for AdsMLBookings and extended for AdsMLFinancials has been moved to the Type Library.

Tax information

The ability to provide tax-related information about Parties, which was developed for AdsMLFinancials, has now also been made available in all the standards. As a result, the underlying structures have been moved to the shared public Type Library.

AuxiliaryReferences

A set of named references for major trading partners as well as generic other references are added.

Party elements

A number of parties such as `Advertiser` and `PayerParty`, which plays important roles in advertising workflows and are used in several standards, have been defined in the type library.

Credit Cards

The `CreditCard` structure previously defined in AdsMLBookings has been moved to the library.

Document Currency Code

The `DocumentCurrencyCode` element is defined in the type library and it is used to provide currency information pertinent to a complete e-commerce document.

Absolute positions

Added elements for specifying the precise positioning of an ad as published.

Generic reference identifiers

Added generic role-specific reference identifiers for recording business significant identifiers for transactions. For example, `BuyersReference`, or `SellersReference`.

Payment Terms

Added a structure defining payment terms.

1.9.3.2 Updated structures

Contract with RateReference

The Contract structure now allows for explicit identification of the "rate" or "level" that applies to a given order or invoice, by means of an optional `RateReference` element.

Additional identifiers for Parties

The `PartyType` has been augmented with the new `AuxiliaryReferences` structure, enabling additional explicitly labelled identifiers for a party,

Taxation information for Parties

The `PartyTaxScheme` structure has been added to the party structure enabling more detailed specification of tax information per party.

A Party Address in the Party Type

A `PartyAddress` structure has been added to the Party type, making it possible to provide an address for a Party without needing to specify a Contact.

Properties for Parties and Contacts

User-defined Properties have been added to Parties and Contacts. This allows the transmission of additional machine-processable information about that Party or Contact, for example the breakdown of a contact's name into Forename and Surname. As always, use of the Properties structure requires prior agreement between the trading partners.

Relaxed Party

An optional "related party" structure has been added to all parties that are based on the AdsML `RelaxedPartyType`.

CommunicationChannels

It is now possible when providing an address (or any other type of communications channel information) to transmit a formal reference to a central database (via the `FormalIdentifier`) either in addition to or in lieu of the actual details of that address. This enables trading partners to reference third party address databases in their AdsML messages.

Also, the `Usage` element has been redefined as a `CodeType` from previous `CodeRootType`.

CommunicaitonChannel.Other

Changed the structure of the CommunicationChannel.Other/Type element to CodeType so that users can identify the code list from which their value was derived.

1.10 Acknowledgements

This document is a product of the AdsML Technical Working Group.

Primary authorship and editing was performed by:

- Jay Cousins (RivCom) – jay.cousins@rivcom.com
- Ulf Wingstedt (CNet Svenska AB) – ulf.wingstedt@cnet.se

2 AdsMLTypeLibrary XML Schema – Overview

This section describes the use of XML Schema in the definition of the AdsML Type Library.

2.1 Schema Architecture

The AdsML Framework uses a modular schema architecture consisting of schema for different standards.

The AdsML Type Library schema defines all reusable common components in the AdsML Framework and is imported into every specific AdsML standard such as AdsMLBookings and AdsMLMaterials.

2.1.1 Schema File

The schema file for the AdsML Type Library is named as follows:

```
AdsMLTypeLibrary-2.0-AS.xsd
```

It starts with the name of the schema, "AdsMLTypeLibrary" followed by current version number. The last two characters provide the status of the schema as either PS (Proposed Standard) or AS (Approved Standard) for public releases (internal working document have status code WD for Working Draft).

2.2 AdsMLTypeLibrary Namespace

AdsMLTypeLibrary defines a namespace:

```
'http://www.adsml.org/typelibrary/2.0'
```

This is defined as the default namespace of the AdsML Type Library Schema. The schema specifies this using *targetNamespace* and *xmlns* attributes as illustrated below,

```
<xs:schema targetNamespace="http://www.adsml.org/typelibrary/2.0"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns="http://www.adsml.org/typelibrary/2.0"
elementFormDefault="qualified" attributeFormDefault="qualified"
xml:lang="en-us">
```

3 Usage rules and guidelines for common components

3.1 Parties and contacts

3.1.1 Parties

One of the requirements for successful e-commerce communications is to identify the parties that are associated with the transaction. In AdsML messages, this information is contained in a group of "party" elements near the root of the transaction, typically at the same level as the identifiers for that transaction. (In some messages there are also additional party elements in sub-structures lower down in the message. These are typically scoped to apply just to the part of the message that they are in.)

AdsML party elements are used to identify an organization, division or similar business entity that has a significant relationship to the transaction at hand. In AdsML, the party structure contains a mandatory `Name` for that entity, one or more unique `Identifiers`, and an optional set of `Contacts` who are associated with the party. (Depending on the context in which the AdsML party structure appears, the `Identifiers` may be optional, but the `Name` is always mandatory.)

No address information for the party is provided. AdsML assumes that the trading partners have databases from which they can use the party's name or, preferably, ID to retrieve additional information if necessary.

3.1.2 Use of the OtherParty element

Party elements typically come in a sequence consisting of:

- One or more elements that have context-specific names like `BookingParty`, `SellingParty`, and `AdMaterialsDeliveredBy`
- An optional, repeatable `OtherParty` element.

In all cases, whenever a suitable explicitly-named party element is available, it **MUST** be used in preference to the `OtherParty` element. The `OtherParty` element should only be used to identify parties for whom no suitable explicitly-named element was provided at that level of the message.

3.1.3 Contacts

Whereas the party structure is used to identify a business entity that is usually a party to the transaction at hand, contacts are individuals (or departments) whose contact information is provided as a convenience to the recipient. The contact structure provides a contact name and one or more "communication channels" by means of which that individual or group can be contacted. These channels can convey, for example, a phone number, an email address, or a street address.

No formal ID is provided for an AdsML contact. Unlike the treatment of parties, AdsML assumes that most contact information contained in AdsML messages is ephemeral, is not necessarily on file with the message recipient, and will only be required during the time that the transaction is being processed.

3.2 Taxation Structures

The structures used in AdsML for taxation information are based on the corresponding UBL 2.0 structures, although sometimes simplified with elements removed. As is the case with UBL, AdsML does not provide structures for detailed tax reporting purposes. Instead, it provides structures to identify a tax regime and convey the information on which tax is based. These aim to be generic and are not based on any specific tax regime.[1]

The primary taxation object in AdsML is `TaxScheme`. The `TaxScheme` element identifies and describes a particular tax regime, i.e. a type of tax, as well as the area of jurisdiction in which the tax applies. For example, "Federal" and "State" taxes in the United States would be two different tax schemes, and "Value Added Taxes" (or VAT) in most European countries would be another tax scheme.

Within a tax scheme there may be multiple categories or levels of tax, for example "Non-Taxable" or "Standard-Rate". These are called Tax Categories and described using the `TaxCategory` element.

Information about specific taxes is located in two different contexts:

1) The `PartyType` includes a `PartyTaxScheme` specifying taxation information that is associated with a particular party.

2) The `PriceComponent` element may have an associated `TaxCategory` that plays the same role as `PartyTaxScheme` above by recording tax information associated with a particular price. The `TaxCategory` is used to provide a category of the tax regime that is expressed as a `TaxScheme` child element. It may also include an optional `Percent` element to express a particular tax percentage that is associated with the category of the tax scheme. For instance, a `'VAT'` tax scheme could have `'standard rate'` as tax category.

3.2.1 General usage rules and guidelines

It should be noted that AdsML does not intend to extend or in any other way change the definition of taxation structures compared to the corresponding elements in UBL. But it is important to be aware that detailed guidelines of usage of taxation structures cannot be provided by global standardization bodies such as OASIS or AdsML because tax regulations vary in different countries or other areas of jurisdiction. Users of AdsML standards need to use the taxation structures provided in accordance with local tax regulations and practices as defined by local tax authorities and business organizations.

See the tax oriented elements' reference texts below for further details, in particular `TaxCategory`, `TaxScheme`, `TaxSubTotal` and `TaxTotalType` provide good overviews.

3.2.2 Party Tax Schemes

A `PartyTaxScheme` is used within a party structure to associate the party with a tax scheme, i.e. a particular type of tax and by implication, the rules that apply according to that tax type. The element also includes other properties about the party that can be used in relation to the tax scheme.

[1] To implement specific tax regimes, the OASIS UBL Technical Committee is working with the OASIS TaxXML Technical Committee to provide guidelines for how specific taxation requirements (e.g., Value Added Tax for the European Community) may be implemented using UBL. Please see the OASIS UBL home page for further information.

The example[2] below shows an `InvoicingParty` “Moderna Produkter AB” where data is recorded regarding two different tax schemes and the company in question is tax exempt according to the first tax scheme:

```
<adsml:PartyTaxScheme>
   <adsml:CompanyID>
     <adsml:IDLabel>TaxRegistrationNumber</adsml:IDLabel>
     <adsml:IDValue>5565624223</adsml:IDValue>
   </adsml:CompanyID>
   <adsml:ExemptionReason>
     <adsml:CodeValue>Registered for company tax</adsml:CodeValue>
   </adsml:ExemptionReason>
   <adsml:RegistrationAddress>
     <adsml:CountryCode>SE</adsml:CountryCode>
   </adsml:RegistrationAddress>
   <adsml:TaxScheme>
     <adsml:ID>
       <adsml:CodeList>TaxSchemeCodes</adsml:CodeList>
       <adsml:CodeValue>SWT</adsml:CodeValue>
       <adsml:Description>Special Withholding Tax</adsml:Description >
     </adsml:ID>
   </adsml:TaxScheme>
</adsml:PartyTaxScheme>

<adsml:PartyTaxScheme>
   <adsml:CompanyID>
     <adsml:IDLabel>VATRegistrationNumber</adsml:IDLabel>
     <adsml:IDValue>SE556562422301</adsml:IDValue>
   </adsml:CompanyID>
   <adsml:TaxScheme>
     <adsml:ID>
       <adsml:CodeList>TaxSchemeCodes</adsml:CodeList>
       <adsml:CodeValue>VAT</adsml:CodeValue>
       <adsml:Description>Value Added Tax</adsml:Description >
     </adsml:ID>
   </adsml:TaxScheme>
</adsml:PartyTaxScheme>
```

The first tax scheme is identified as an “SWT”, a Special Withholding Tax. In this case, the company is registered for company tax, as specified in the `ExemptionReason` element, which means that the Payer should not withhold any tax for this payment. The identifier of the company within this tax scheme is provided in the `CompanyID` element.

The second tax scheme shows that the company is registered for VAT sales tax and provides the registration number within that scheme.

3.2.2.1 Usage rules and guidelines

Party tax scheme information must be provided in accordance with local tax regulations.

See also the `PartyTaxScheme` element reference text below for further details.

[2] The example is adapted from the documentation of the Swedish Svefaktura standard, a Swedish invoicing standard based on UBL 1.0.

3.3 Pricing

3.3.1 CPM (Cost per Thousand) pricing

CPM pricing can be described with a textual note in `PriceComponent/DescriptionLine`, and/or by expressing the formula in `PriceComponent/CalculationSpecification`. A CPM calculation specification uses the *`divisor`* attribute of `PricePerUnit` with a value of `1000`, to indicate that the price should be divided by 1000 when applied to an individual unit.

Example:

```
<adsml:PriceComponent adsml:sequenceNo="1">
  <adsml:PriceComponentName>
    <adsml:CodeValue>Metro Area Combined Distribution</adsml:CodeValue>
  </adsml:PriceComponentName>
  <adsml:Amount>7588.00</adsml:Amount>
  <adsml:DescriptionLine>CPM Costing Method ($56 per
thousand)</adsml:DescriptionLine>
  <!-- Calculation identifies the item by name, the quantity that will be
distributed, and price per thousand items. -->
  <adsml:CalculationSpecification>
    <adsml:Unit>
      <adsml:CodeValue xsi:type="adsml-
cv:AdsMLUnitOfMeasureCV">piece</adsml:CodeValue>
    </adsml:Unit>
    <adsml:NumberOfUnits>125000</adsml:NumberOfUnits>
    <adsml:PricePerUnit adsml:divisor="1000">56.00</adsml:PricePerUnit>
   </adsml:CalculationSpecification>
</adsml:PriceComponent>
```

The above example shows pricing of $56 per 1000 pieces. The unit is defined as "`piece`", just as it would be with standard pricing, but the price of $56.00 is adjusted by having a *`divisor`* of 1000. In this example an *`xsi:type`* attribute identifies the controlled vocabulary that was used for the Unit name, but this could equally have been accomplished by conveying the CV name in an `CodeList` element.

3.4 Internationalization support

The AdsML Type Library includes internationalization support by providing basic string types with attribute extensions to express language and reading directionality. The AdsML approach follows the W3C's "Internationalization Tag Set (ITS) Version 1.0" (http://www.w3.org/TR/its/) by adding *`xml:lang`* and *`dir`* attributes. In a multilingual environment content may also be localized, that is, translated and adapted to meet the requirements of the 'locale' where it is used. In light of this, AdsML also provides a specific *`source`* attribute to indicate which language version is the source or 'original' text from which other translations have been derived.

The types with internationalization support are named after their basic predecessors with an 'i18n' extensions suffix, for example, `ShortStringType.i18n`. The acronym 'i18n' is in this context a commonly used abbreviation for the word 'internationalization' based on the number of letters between the 'i' and 'n' in the word 'internationalization'. See section 5 below for a list of the extended simple types.

The internationalized types are used in a number of contexts, primarily for elements that include human readable texts. In order to support multiple language texts in a particular context, elements with i18n capabilities are defined as repeatable. See for instance the `CodeType` where the `Description` element is repeatable to support alternative descriptions in different languages.

i18n attributes may also be available in more complex types with one or more level of child elements. In such cases, the language metadata provided at the parent level is considered to apply to all child elements. When language information is provided at the level of a parent element, additional language information **MUST NOT** be provided in any of its child elements.

4 Type Library Component Reference

This is a reference section describing the attributes, elements, and other components comprising the AdsML Type Library. The components are listed in alphabetical order.

Each component represents a distinct piece of business information and has specific business meaning. The structure and semantics of each component are explained. Where a component is reused in an AdsML standard, then any additional semantics that are given or 'added' to the component because of its use in a specific business context are described in that standard's specification.

Business rules that are not possible to express using XML Schema are expressed in the written description of each component. Note that the XML Schema specification includes additional rules.

4.1 Element: AbsolutePosition

The `AbsolutePosition` element identifies the precise point on the page at which an ad has been published. To specify this it has four optional elements:

- `FromThisPointOnPage` – identifies the point on the page from which the x-y co-ordinates are taken to start.
- `ToThisPointOnAd` – identifies the point on the ad at which the x-y co-ordinates are taken to end.
- `XCoordinate` – identifies the 'x' co-ordinate position of the ad.
- `YCoordinate` – identifies the 'y' co-ordinate position of the ad.

The structure allows positioning to be indicated by either or both of the 'x' and 'y' coordinates in combination with positioning codes recorded in the 'from this point' and 'to this point' elements. If the `FromThisPointOnPage` and `ToThisPointOnAd` elements are not specified then the default position of the x-y coordinates **SHOULD** be assumed to be from the top left of the page to the top left of the ad, with the 'x' coordinate expanding to the right of the page and the 'y' coordinate expanding down the page.

Attributes

No attributes.

4.2 Element: Amount

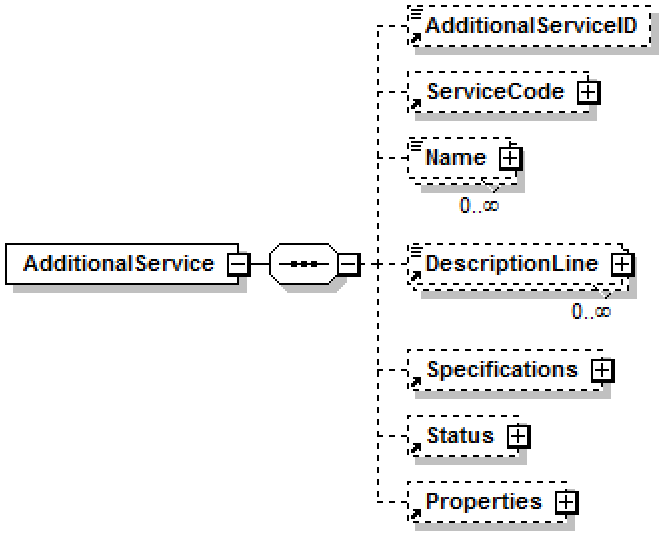
The `Amount` element records a financial amount with an accuracy of two decimal points. The element is declared as `AmountType`.

Attributes

No attributes.

4.3 Element: AdditionalService

The `AdditionalService` element is a generic structure that should be used to specify any kind of service that would be considered to be "additional" in its usage context. For instance, in AdsMLBookings, the `AdditionalService` element is used to specify services that go beyond the actual publication of an ad during a specific time. It could be repro or other materials/artwork services that are performed by the publisher.



The AdditionalService element includes a number of child elements where all are optional in order to support many different usage scenarios.

The AdditionalServiceID can be used to record a unique identifier for the service instance; this id can later be used to reference the service from, for instance, a price structure.

The ServiceCode element should be used to record a code that represents the service. It is defined as a CodeType.

The Name element should be used to record a human friendly name for the service. The DescriptionLine can be used to hold a short description of the service. Both elements may be repeated to record the name and description of the service using alternative languages.

The generic Specifications element allows the user to capture other codes and/or instructions on how the service should be performed.

The Status element should be used to record current status, a feature that is expected to be used mainly in responses and status messages.

If required, it is also possible to include application-specific data using the general Properties element.

Attributes

No attributes.

4.4 Element: AdditionalServiceID

The AdditionalServiceID element records a unique identifier for an AdditionalService. The element is declared as QIDType.

See AdditionalService for further information.

Attributes

No attributes.

4.5 Element: AdministrativeResponse

The AdministrativeResponse element enables message responses on a technical level, either acknowledging the receipt of a specific AdsML XML message or reporting technical errors with the message and its transmission.

The child element `TransmissionDescription` must be used to identify the message the administrative response is about.

The *`messageCode`* attribute must always be set to the same message code as in the message the response is about.

As a receipt of a successfully arrived message, the *`messageClass`* attribute **MUST** be set to '`MessageReceivedAcknowledgment`'.

In case of an error, the *`messageClass`* **MUST** be set to '*TechnicalError*' and the error **SHOULD**, if possible, be specified using the `Error` child element, recording the error using the `CodeType` content model.

If required, it is also possible to include application-specific data using the general `Properties` element.

Attributes

messageCode (required)

Records the AdsML Framework message type code for the message that the response is about.

messageClass (required)

The message class defines if the response is an acknowledgement or an error.

4.6 Type: AdsMLItemType

The `AdsMLItemType` type is the abstract base type for definition of AdsML messages. The `AdsMLItemType` type content model is a required `Header` and optional and repeatable `Properties` elements.

The `Header` element contains a sequence of required `TransmissionFrom`, `TransmissionTo`, and optional `DigitalSignatures` elements.

The `TransmissionFrom` element identifies the sender of the AdsML message; the `TransmissionFrom` element is declared as `PartyType`.

The `TransmissionTo` element identifies the intended recipient of the AdsML message; the `TransmissionTo` element is declared as `PartyType`.

The `DigitalSignatures` element records any digital signatures that have been applied to the AdsML message.

The `Properties` element can be used to define application-specific extensions.

Attributes

transmissionID (required)

A globally unique identifier for the whole XML message. Every AdsML message **MUST** have a unique identifier. The *`transmissionID`* attribute is declared as `QIDType`.

transmissionStatus (optional)

The status of the message. Can be used for specifying that the message and/or its included transactions is a test. The *`transmissionStatus`* attribute is declared as `TransmissionStatusCV`. If the attribute is omitted, the message status **MUST** be interpreted as "`Production`".

firstTransmissionDateTime (required)

The time the message was first transmitted. Records the local transmission time according to the sending system. This time stamp should not be updated in case of a re-transmission. The *firstTransmissionDateTime* attribute is declared as DateTimeType.

transmissionDateTime (required)

The time the message was transmitted. Records the local transmission time according to the sending system. In case of a message re-transmission, this time stamp must show the actual transmission time. The *transmissionDateTime* attribute is declared as DateTimeType.

systemsID (required)

Expresses an ID for the system that generates the message. The *systemsID* attribute is declared as ShortStringType.

transmissionSequence (required)

An incremental value assigned by the sender of the message that will allow the receiver to sort incoming AdsML messages in the right order, or notify when messages are received in an improper sequence. It is supposed to be used as alternate sorting mechanism to the sending time and should thus not be based on the sending system's local time.

The *transmissionSequence* attribute is declared as LongTokenType.

administrativeResponseRequired (optional)

Allows the sender to specify that a particular message requires an administrative response from the receiver. The *administrativeResponseRequired* attribute is declared as BooleanType. If the attribute is not specified in a message, the message **MUST** be interpreted as if the attribute had been given with a value of "false".

sendCount (optional)

In case of re-transmission of messages, the send count should be incremented for each re-transmission. The *sendCount* attribute is declared as PositiveIntegerType. If the *sendCount* attribute is not specified in a message, the message **MUST** be interpreted as having a send count of "1".

schemaVersion (required)

Records the version of the schema that the AdsML message conforms to. The *schemaVersion* is recorded as a SchemaVersionType.

schemaProfile (optional)

Records the name of the profile schema that an AdsML message conforms to. A profile is a definition of a subset of an AdsML standard, including usage rules, that trading partners may agree to use. The default value for a profile identifier in any AdsML message is blank (omitted), which indicates that the message conforms to the AdsML specification as a whole and no formal profile has been applied. The *schemaProfile* is defined as a VersionedQIDType.

4.7 Element: AdType

The `AdType` element should be used to record the type of an advertisement. It is defined as a `CodeType` and can be validated against a user defined controlled vocabulary. Typical values are insert, display, classified display or classified liner ad.

Attributes

No attributes.

4.8 Element: Advertiser

The `Advertiser` element identifies a party taking the role as advertiser in a transaction. It is defined as a `RelaxedPartyType`.

Attributes

No attributes.

4.9 Element: AdvertisersReference

The `AdvertisersReference` element is used by a party acting as the advertiser to record their own reference identifier. The value is recorded as a `LongNormalizedStringType`.

See also `AuxiliaryReferences`.

Attributes

No attributes.

4.10 Type: AnyMixedContentType

The `AnyMixedContentType` allows mixed content of any type as long as the content does not invalidate the well-formedness of the surrounding XML document instance.

In the event that an XML document is contained as content then the `AnyMixedContentType` element allows elements from any namespace to appear without validation.

Attributes

No attributes.

4.11 Element: AuthorizationCode

The `AuthorizationCode` element specifies the code that is returned by the credit card processing system indicating that the credit card payment can be collected. Note that this is not the same as the actual collection of the payment itself.

See `CreditCard` for more information.

Attributes

No attributes.

4.12 Element: AuthorizationExpires

The `AuthorizationExpires` element contains the date and time on which the Authorization Code specified in the `AuthorizationCode` element will expire. This will mean that a payment request using the Authorization Code will be rejected by the payment processor. This element is specified as `DateTimeType`, which means it should include a time component as well as a date.

See `CreditCard` for more information.

Attributes

No attributes.

4.13 Element: AuthorizedTime

The `AuthorizedTime` element indicates the date/time when the credit card authorization was made. In other words, this is the date and time of the `AuthorizationCode` element.

See `CreditCard` for more information.

Attributes

No attributes.

4.14 Element: AuxiliaryReferences

The `AuxiliaryReferences` element may be used in many contexts to supply additional reference identifiers for the context object. It includes a set of explicitly named reference elements, whose names indicate the origin of each reference. For instance, if used inside an advertiser party, the `SellersReference` would be the selling party's reference to that advertiser.

The following optional explicit reference elements are provided: `BuyersReference`, `SellersReference`, `InvoicersReference`, `PayersReference`, `AdvertisersReference` and `DeliverersReference`.

Finally, an optional and repeatable `OtherReference` element allows for further references. It is defined as a `ReferenceValueType`.

In any given usage context, one or more of the references contained within `AuxiliaryReferences` may be irrelevant. In such cases only the references that make sense in the current context should be used, and the rest should be ignored.

Attributes

No attributes.

4.15 Element: BasePrice

See `PriceComponent`.

Attributes

No attributes.

4.16 Element: BookingParty

Defined as a `PartyType`, the `BookingParty` is a party taking the role of a booker in a transaction.

Attributes

No attributes.

4.17 Element: BusinessMessageDate

The `BusinessMessageDate` element is intended to be used to record a business significant date for a transaction that might be different from other recorded dates such as date of transmission or the date the message was assembled.

It is defined as a `DateTimeDateType`.

Attributes

No attributes.

4.18 Element: BuyersReference

The `BuyersReference` element is used by a party acting as the buyer of advertising space to record their own reference identifier value for a transaction or other business object. The value is recorded as a `LongNormalizedStringType`.

See also `AuxiliaryReferences`.

Attributes

No attributes.

4.19 Element: CalculationRate

The `CalculationRate` element records the factor used for the conversion of an amount from the source currency to the target currency. The `CalculationRate` element is used within the `ExchangeRate` structure.

See `ExchangeRate` for more information.

Attributes

None.

4.20 Element: CalculationSpecification

See `PriceComponent` for more information.

Attributes

No attributes.

4.21 Element: Campaign

The `Campaign` element records a reference code and name of a campaign (also called “Estimate” in some regions) that a booking, invoice or other AdsML object may be related to. It is defined as a `CodeType`.

Attributes

No attributes.

4.22 Element: CardExpires

The `CardExpires` element is used to carry the expiry date of the credit card (normally in a 'mm/yy' format). When carrying the details for a future credit card payment, this field should be considered to be mandatory, as it is required for payment processing. However, the `CreditCard` element does not require this to be mandatory due to the capability of recording credit card payments already made.

See `CreditCard` for more information.

Attributes

No attributes.

4.23 Element: CardHoldersAddress

The `CardHoldersAddress` element contains the Cardholder's address with full structure, based on `PhysicalAddress`. This is required when a credit card transaction is processed and the customer is not present, and is designed to reduce fraud.

See `CreditCard` for more information.

Attributes

No attributes.

4.24 Element: CardNumber

The `CardNumber` element is the number that is embossed on the credit card (i.e. the long number in the middle on the front of the card). The length of this number varies from card type to card type. This number **SHOULD** not include white space.

See `CreditCard` for further details.

Attributes

No attributes.

4.25 Element: CardStartDate

Some types of credit card or payment processors require that a Start Date be specified as part of the payment criteria. The Start Date is sometimes embossed on the front of the card adjacent to the 'End Date' or 'Valid Until'. This element (where used) should be in the format 'mm/yy'.

See `CreditCard` for further information.

Attributes

No attributes.

4.26 Element: CardTransactionReference

The `CardTransactionReference` is a unique reference number, which is generated by the card processor to identify the individual transaction. This can be used at a later date in the credit card reconciliation process.

See `CreditCard` for further information.

Attributes

No attributes.

4.27 Element: CardType

See `CreditCard` for further information.

Attributes

No attributes.

4.28 Element: CardVerificationValue

The `CardVerificationValue` is a 3 or 4 digit number that is printed on the card either on the front or, more often, on the signature strip on the reverse of the card. This number is used as an additional security measure, as it is not included in the information held on the magnetic strip on the card, and it is used to ensure that the person executing the transaction actually has the physical card in their possession.

See `CreditCard` for more information.

Attributes

No attributes.

4.29 Element: ChangeSpecification

The `ChangeSpecification` element captures a description of a requested change, recording this using a content model based on `CodeType`. The change can be described using a mandatory machine readable code, together with an optional text description. Both values can use a controlled vocabulary for validation.

A pointer reference to a location within the message can be provided using the optional `ChangeLocationReference` element. When used, its value **MUST** correspond to a unique `QIDType` value of an attribute or element in the message. The `ChangeLocationReference` element is also itself declared as `QIDType`.

Attributes

importance (optional)

A value expressing the importance of the change in a scale of 1-5. If the attribute is not specified in a message, the message **MUST** be interpreted as if the attribute had been specified with a value of "`3`".

4.30 Type: CodeList

See `CodeType` for more information.

Attributes

No attributes.

4.31 Type: CodeType

The `CodeType` is a general structure for recording codified values.

The `CodeType` content model is an optional `CodeList` and mandatory `CodeValue` elements followed by an optional `Description` element.

The `CodeValue` element records the actual code value as a machine-readable code. It is defined as a `LongCodeRootType` taking values of up to 255 characters.

The `CodeList` element identifies the code list or controlled vocabulary from which the code value is taken. Usage of this element allows the receiving application to both know which code list a code value is taken from, and by custom application logic ensure that the code value is coming from that list.

In cases where no explicitly named code list (controlled vocabulary) is given, the code value is to be resolved according to definitions in the Trading Partner Agreement.

Both `CodeList` and `CodeValue` are defined as `CodeRootType`, allowing schema defined controlled vocabularies to be used and validated through the *`xsi:type`* attribute. This approach has the advantage of that validation of code values can be performed by generic XML Schema validators reducing the need for custom application logic. Controlled vocabularies can come from either the set of recommended controlled vocabularies in the *AdsML Controlled Vocabularies Schema*, or be defined in user extension schemas if agreed by trading partners.

For instance, the following sample shows how the value of the `CodeValue` element is defined as being a member of the `adsml-cv:AdsMLStatusCodeCV`:

```
<CodeValue xsi:type=”adsml-cv:AdsMLStatusCodeCV”>
Completed
</CodeValue>
```

An alternative approach using the `CodeList` element instead of the *`xsi:type`* attribute would look like:

```
<CodeList>AdsMLStatusCodeCV</CodeList>
<CodeValue>Completed</CodeValue>
```

The `Description` element provides a human-readable descriptive text of the codified value. It may be repeated to capture descriptions in alternative languages.

Attributes

No attributes.

4.32 Type: CodeValue

See `CodeType` for more information.

Attributes

No attributes.

4.33 Attribute group: commonMessageAttributes

The *commonMessageAttributes* group specifies a set of attributes that appear in all AdsML business messages.

Attributes

messageID (required)

A globally unique message identifier. Each message **MUST** have a unique ID. The value is recorded as `QIDType`.

messageHeaderLine (optional)

Text header providing a short headline for the message. It can be used as a human readable reference text. The value is recorded as `LongStringType`.

messageClass (fixed: 'BusinessTransaction')

The message class to which the message belongs.

messageAssembledTime (optional)

A time stamp specifying the time when the message was assembled by the sending system. The main situation where this attribute is expected to be used is when there is an expected latency between the assembly of the message and its transmission. If this time stamp is not provided, the assemble time can be assumed by the receiver to be the same as the transmission time. The value is recorded as `DateTimeType`.

presentationTransformation (optional)

A text string reference to a stylesheet that can be used to transform the message into a human readable presentation. This value is expected to usually contain a URI pointing to an XSLT or CSS stylesheet, but could alternatively be a name or any other convention the trading partners agree to use.

4.34 Type: CommunicationChannel.BaseType

The `CommunicationChannel.BaseType` provides an abstract base type intended for extension in order to derive specific types of communication channel. The communication channel base type consists of optional and repeatable `Usage` elements.

The optional `FormalIdentifier` element can be used to provide a formal reference to an address (or any other type of communications channel information) in a central database, either in addition to or in lieu of the actual details of that address. This enables trading partners to reference third party address databases in their AdsML messages

The `Usage` element specifies the intended use of the communication channel. For example, `Usage` can be used to indicate if a telephone number is a business or private line.

The `Usage` element is declared as `CodeType`.

Attributes

priority (optional)

Assigns a priority rating used to identify the preferential sequence in which communication channels should be used in the event that more than one communication channel element is present.

4.35 Element: CommunicationChannel.EMail

The `CommunicationChannel.EMail` element records an electronic mail address as a string. The `CommunicationChannel.EMail` element is declared as `EMailAddressType`.

Attributes

No attributes.

4.36 Element: CommunicationChannel.Other

The `CommunicationChannel.Other` element extends the `CommunicationChannel.Base` type to define a generic address structure. The `CommunicationChannel.Other` content model is a sequence of required `Type`, and optional and repeatable `Specification` elements.

The `Type` element classifies the type of address being recorded. The `Type` element is declared as `CodeType`.

The `Specification` element records the address details in a generic form as labelled values. The `Specification` element is declared as `LabeledValueType`.

Attributes

No attributes.

4.37 Element: CommunicationChannel.Phone

The `CommunicationChannel.Phone` element records a phone number as a string. The `CommunicationChannel.Phone` element is declared as `PhoneAddressType`.

Attributes

No attributes.

4.38 Element: CommunicationChannel.PhysicalAddress

The `CommunicationChannel.PhysicalAddress` element records a physical address as a string. The `CommunicationChannel.PhysicalAddress` element is declared as `PhysicalAddressType`.

Attributes

No attributes.

4.39 Element: CommunicationChannel.WWW

The `CommunicationChannel.WWW` element records an Internet address as a URI. The `CommunicationChannel.WWW` element is declared as `URIAddressType`.

Attributes

No attributes.

4.40 Element: CompanyID

The `CompanyID` element identifies a company as registered with the relevant authority for company regulation. Note that this element is used within the `PartyTaxScheme` and may be different from the identifier of the party.

Attributes

No attributes.

4.41 Element: Contact

The `Contact` element is defined as a `ContactType`, please see its description.

Attributes

See ContactType.

4.42 Type: ContactType

The `ContactType` type provides a content model for specifying contact information using optional `Role`, `Name`, `Title`, and optional and repeatable choice between different `CommunicationChannel` elements given by the `HumanCommunicationChannelsGroup`.

The `Role` element specifies the role played by the contact. Values can be validated by controlled vocabularies, if required. In addition, using the `Role` element it is possible to specify the role played by this contact in the parent context.

The `Name` element specifies the name of the contact.

The `HumanCommunicationChannelsGroup` elements specify where telephone, physical address, Email, and other methods for communicating with the contact can be recorded.

If required, it is also possible to include application-specific data using the general `Properties` element.

Attributes

priority (optional)

Assigns a priority rating to the `Contact` element. The priority rating is used to identify the sequence in which contacts should be contacted in the event that more than one `Contact` element is present.

i18nAttributes (optional)

The *i18nAttributes* group supports internationalization by providing attributes to record language, directionality and source.

4.43 Element: ContentData

The ContentData element is a container in which inline content is held. Note that any inline content **MUST** conform to the constraints applicable to character data in XML documents as defined by the W3C's XML 1.0 specification.

The ContentData element is declared as AnyMixedContentType, which allows mixed content of any type.

Attributes

No attributes.

4.44 Element: ContentDataEncoding

The ContentDataEncoding element records the encoding applied to data described by this element. The ContentDataEncoding element records encoding using the EncodingRootType. As with all contexts where a root type is used, the EncodingRootType can be restricted to a list of values defined by a controlled vocabulary.

Attributes

No attributes.

4.45 Element: ContentDataRef

The ContentDataRef element provides a reference to externally located content in the form of a URI. The ContentDataRef element is declared as URIType.

Attributes

No attributes.

4.46 Element: ContentProperties

The ContentProperties element is used to describe a content file using a sequence of optional MIMEType, Format, FormatProfile, ContentDataEncoding, EncryptionMethod, ContentSizeInBytes, and FileName elements. See these element definitions for more information.

Note that the content properties describe the characteristics of the content file before any encoding or encryption has been applied to it.

Attributes

No attributes.

4.47 Element: ContentSizeInBytes

The ContentSizeInBytes records the size in bytes of a content file as a positive integer.

Attributes

No attributes.

4.48 Element: Contract

The `Contract` element records a set of metadata about a contract. See the *type* `ContractType` type for more information.

Attributes

No attributes.

4.49 Element: ContractReference

The `ContractReference element` records a reference to a contract as a `LongStringType`.

Attributes

No attributes.

4.50 Element: ContractType

The `ContractType` element is defined as `CodeType` and used within the `ContractType` type to record the type of a contract.

See the *type* `ContractType` for more information.

Attributes

No attributes.

4.51 Type: ContractType

The ContractType type is used to record a set of properties of a contract document. It includes the following optional elements:

- `ContractReference` – A string reference, e.g. a contract identifier or a reference to a rate card.
- `IssueDate` – The date and time of issue for the contract.
- `ContractType` – The type of contract recorded as a `CodeType`.
- `ValidityPeriod` – Defined as a `PeriodType`, it records the validity period of the contract as a date range or duration.
- `SectionReference` – A reference string to a subsection of the contract.
- `RateReference` – A reference string to a rate (or a level) in the contract.

If required, it is also possible to include application-specific data using the general `Properties` element.

Attributes

No attributes.

4.52 Element: Country

The `Country element` records a geopolitical country by means of a country code and name. See `CountryType` for more information.

Attributes

No attributes.

4.53 Type: CountryType

The `CountryType` type records a geopolitical country by means of a country code and name expressed using `CountryCode` and `CountryName` elements. It is **RECOMMENDED** to use the ISO country codes; a controlled vocabulary containing the ISO codes is defined in the AdsML Controlled Vocabularies.

Attributes

No attributes.

4.54 Element: CountryCode

The `CountryCode element` records a country code. It is **RECOMMENDED** to use the ISO country codes as defined in the AdsML Controlled Vocabularies.

Attributes

No attributes.

4.55 Element: CountryName

The `CountryName element` records a country name as a `ShortStringType`.

Attributes

No attributes.

4.56 Element: CreditCard

The `CreditCard` element captures data needed to either process a credit card payment or to convey the details of a payment that has already been processed or authorized elsewhere.

See `CreditCardType` for full details.

Attributes

No attributes.

4.57 Type: CreditCardType

The `CreditCardType` type captures data needed to either process a credit card payment or to convey the details of a payment that has already been processed or authorized elsewhere. The `Status` element is used to record the current status of the credit card transaction. Typical values could be "authorized", "collected" or "referred" ("referred" means the recipient of the message needs to make a call to the bank to get the transaction approved).

For cases when the seller should process the complete credit card transaction, the following elements should be used:

- The `CardType` element holds the type of card, e.g. `'AMEX'` or `'VISA'`. It is defined as a `CodeRootType` and can thus use a controlled vocabulary for validation.
- The `CardNumber` element holds the card number as a straight sequence of digits without white space.

- The IssueNumber is an optional element used where the card type has an Issue Number printed on the front. This is used to indicate the number of cards the customer has had for the account.
- The CardStartDate element holds the optional 'Start Date' or 'Valid From' date indicated on the card (usually in the format mm/yy).
- The CardExpires element holds the card's expiration day and month.
- The NameOnCard holds the name as printed on the card.
- The CardholdersAddress holds the address of the owner of the card. It can be used to verify that the owner of the card is the same as the payer.
- The CardVerificationValue holds the Card Verification Value (CVV), sometimes also referred to as CVC (Card Verification Code), CVV-2 or CVC-2. The CVV code is an additional 3 or 4-digit security number that typically is printed (not embossed) on the signature strip on the back of a bank-issued credit card, though in some cases it can be found on the front of the card. It can be used to verify that the payer is in possession of the card.
- The AuthorizedPayment element is a mandatory Boolean that indicates if the payment has been authorized. This will be set to 'false' in the case of a payment that is due to be taken.
- The generic Properties element for additional data.

NOTE: The CardStartDate and CardExpires elements require the creating system to generate a full date using either the first or last day of the month in question, even when only a partial date is involved (e.g. month/year),

For cases when the payment has been reserved by the payer in advance, a smaller set of data is needed in order to collect the credit card payment:

- CardType
- CardNumber
- CardExpires
- The AuthorizationCode element holds a code that can be used to collect a reserved payment. It is valid until the expiration time expressed in its AuthorizationExpires sibling.
- The AuthorizationExpires element holds the expiration time for the reserved payment.
- The AuthorizedPayment element should be set to 'false' in the case of a payment that is due to be taken.
- The AuthorizedTime element indicates the date/time when the credit card authorization was made. In other words, this is the date and time of the AuthorizationCode element.

For cases where the credit card payment has already been taken, the following elements should be used:

- The MerchantCode element contains the code that identifies the company that has processed the credit card payment
- The CardTransactionReference contains the unique transaction reference generated by the card processor.
- The DataSource element is used to indicate the origin of the credit card data.

- `CardType`
- `CardNumber` (although part of this may have been replaced with an additional substitution such as *)
- `CardExpires`
- `NameOnCard`
- The `AuthorizedPayment` element is used to indicate that the Payment has been processed and should be set to `'true'`.

Note that credit card refunds are not supported in this release.

Attributes

No attributes.

4.58 Element: CurrencyCode

The `CurrencyCode` element specifies currency codes and is used in connection with prices where it qualifies a financial amount (recorded by the `Amount` element) by the currency in which the amount is specified. The `CurrencyCode` element is declared as `CurrencyCodeRootType`. The code values can be validated against a controlled vocabulary.

Attributes

No attributes.

4.59 Type: CurrencyPriceDeclarationType

The Currency`PriceDeclarationType` type is an extension of the `PriceDeclarationType` with optional currency information in the `CurrencyCode` element.

All price components given **MUST** be in the same currency code as stated in the `CurrencyCode` element.

The optional `ExchangeRate` element can be used to provide information about conversion to or from the currency identified by the `CurrencyCode` element.

See also `PriceDeclarationType` for further information.

Attributes

No attributes.

4.60 Element: DataSource

The `DataSource` element is used to specify a source from which information such as credit card data was obtained. Example values in this case could be "phone", "mail", "in person", etc. As this element is based on a `CodeType`, it can use a controlled vocabulary for validation.

See also `CreditCard` for more information.

Attributes

No attributes.

4.61 Element: Date

The `Date element` records a date as a `DateType`.

Attributes

No attributes.

4.62 Type: DecimalMeasurementType

The `DecimalMeasurementType` records a measurement value qualified by a unit of measure.

The `DecimalMeasurementType` content model is required `UnitOfMeasure` and `Value` elements.

The `UnitOfMeasure` element identifies the measurement unit in which the value is specified and which gives the measurement value semantic meaning. The `UnitOfMeasure` element is declared as `CodeRootType`; as with all contexts where a root type is used, the `CodeRootType` can be restricted to a list of values defined by a controlled vocabulary.

The `Value` element records the measurement value as a decimal. The `Value` element is declared as `DecimalType`.

For example, if a measurement of 1.25 cm were being recorded then the value '`1.25`' would be recorded as the `Value` and the `UnitOfMeasure` would record '`cm`' as the measurement qualification that gave the value its semantic significance.

Attributes

No attributes.

4.63 Element: DeliverersReference

The `DeliverersReference` element is used by a party performing a materials delivery to record their own reference identifier value for a transaction or other business object. The value is recorded as a `LongNormalizedStringType`.

See also `AuxiliaryReferences`.

Attributes

No attributes.

4.64 Element: DeliveryOrderingParty

The `DeliveryOrderingParty` element identifies the party responsible for ordering a materials delivery. The `DeliveryOrderingParty` is declared as a `PartyType`.

The `DeliveryOrderingParty` always engages the services of a third party to make the delivery on their behalf (i.e. a `DeliveringParty`).

Note that the `DeliveryOrderingParty` has similar semantics to the `adsml:DeliveringParty` element, with the exception that the `adsml:DeliveringParty` party may or may not use a third party service provider to make the delivery on their behalf.

Attributes

No attributes.

4.65 Element: DeliveringParty

The `DeliveringParty` element identifies the party making a materials delivery. The `DeliveringParty` is declared as a `PartyType`.

Note that,

- When used alone without an accompanying `DeliveryOrderingParty`, for example in `AdMaterial` delivery messages, the `DeliveringParty` will always have the business responsibility for making the delivery.
- When used in `AdMaterialDeliveryOrder` messages in conjunction with the `DeliveryOrderingParty`, the `DeliveringParty` may or may not have this business responsibility, depending on their arrangements with the `DeliveryOrderingParty` and with the party to which they will deliver the materials.

Attributes

No attributes.

4.66 Element: Description

The `Description` element records a general descriptive text with no restrictions on the length of the description.

The `Description` element is declared as `StringType.i18n` and thus supports language metadata according to the *`i18nAttributes`* group.

Attributes

i18nAttributes (optional)

The *`i18nAttributes`* group supports internationalization by providing attributes to record language, directionality and source.

4.67 Element: DescriptionLine

The `DescriptionLine` element records a general short text that is used to record descriptive text. The `DescriptionLine` element is declared as `LongStringType.i18n` and thus supports language metadata according to the *`i18nAttributes`* group..

Attributes

i18nAttributes (optional)

The *`i18nAttributes`* group supports internationalization by providing attributes to record language, directionality and source.

4.68 Group: DigitalDeliveryCommunicationChannelsGroup

The `DigitalDeliveryCommunicationChannelsGroup` element group provides a choice between communication channel elements used to specify phone, Email, Internet, and other communication channels by which a digital delivery of content can be made.

Four elements are contained in the group:

The `CommunicationChannel.EMail` element is used to record an EMail address with which a digital delivery of content can be made.

The `CommunicationChannel.Phone` element is used to record a phone number with which a digital delivery can be made (i.e. by ISDN).

The `CommunicationChannel.WWW` element is used to record an Internet Web address location at which a digital delivery can be made.

The `CommunicationChannel.Other` element is used to record other forms of communication channel with which a digital delivery can be made, recording the communication channel address in a generic way.

4.69 Element: DigitalSignatures

The `DigitalSignatures` element records any digital signature(s) applied to an AdsML message. A digital signature **MUST** be recorded as a W3C XML Signature, the signature elements contained as element children inside `DigitalSignatures`, and the signature produced as specified in the W3C XML Signature specifications.

Attributes

No attributes.

4.70 Element: DisclaimerText

The `DisclaimerText` element is used to record a legal disclaimer. It is used in contexts such as a booking where it is required to provide a legal disclaimer for business and legal reasons. The element is not strictly limited to disclaimers, but can be used for any legalistic language that the sender of a message wishes to convey.

The `DisclaimerText` element is declared as `StringType.i18n` and thus supports language metadata according to the *`i18nAttributes`* group.

Attributes

i18nAttributes (optional)

The *`i18nAttributes`* group supports internationalization by providing attributes to record language, directionality and source.

4.71 Element: DocumentCurrencyCode

The `DocumentCurrencyCode` element is used to identify a currency pertinent to a complete e-commerce document.

Attributes

No attributes.

4.72 Element: DocumentRendering

The `DocumentRendering` element allows the sender of a business message to convey a human-readable digital rendering of the document.

See `DocumentRenderingType` for further information.

Attributes

i18nAttributes (optional)

The *i18nAttributes* group supports internationalization by providing attributes to record language, directionality and source.

4.73 Type: DocumentRenderingType

The DocumentRenderingType type allows the sender of a business document to convey a digital rendering of a document either by containership (e.g. a PDF is embedded in the message) or reference (a URL or equivalent is provided so that the recipient can automatically retrieve the rendering).

The element describes a file and its location in the same way as other binary attachments in AdsML, by a sequence of optional ContentProperties, ContentData and DigitalDeliveryCommunicationChannelsGroup elements.

DocumentRendering supports the *i18nAttributes* group for language metadata. Data in these attributes refer to the language in the rendered document that is either contained or referenced.

Note that the DocumentRendering structure does not cover delivery of an actual paper document.

Attributes

i18nAttributes (optional)

The *i18nAttributes* group supports internationalization by providing attributes to record language, directionality and source.

4.74 Element: Duration

The Duration element is defined as a DecimalMeasurementType and is used to capture an amount of time. For example, thirty minutes.

Attributes

No attributes.

4.75 Element: DurationMeasure

See PeriodType.

Attributes

No attributes.

4.76 Type: EMailAddressType

The EMailAddressType extends the CommunicationChannel.Base type to define an electronic mail address.

The EMailAddressType content model is a required EMailAddress element.

The EMailAddress element records an electronic mail address as a string.

Attributes

No attributes.

4.77 Element: EncryptionMethod

The `EncryptionMethod` element records the encryption method used to encrypt the data contained inside the `ContentData` element. The `EncryptionMethod` element records encryption method using `EncryptionMethodRootType`, which is a `ShortTokenType` data type. As with all contexts where a root type is used, the `EncryptionMethodRootType` can be substituted with a controlled vocabulary of specific values if required.

Attributes

No attributes.

4.78 Element: EndDateTime

The `EndDateTime` element records a date or a date time. See also `StartDateTime` and `PeriodType`.

Attributes

No attributes.

4.79 Element: Error

The `Error` element records a description of an error as a code with code list and description. It is defined as a `CodeType`.

Attributes

No attributes.

4.80 Element: ExemptionReason

The `ExemptionReason` element records text that explains the reason for a party's exemption from a tax.

Attributes

No attributes.

4.81 Element: ExchangeMarketID

The `ExchangeMarketID` is used to identify a currency exchange market in an `ExchangeRate` structure. It is defined as an `LabeledIDType`.

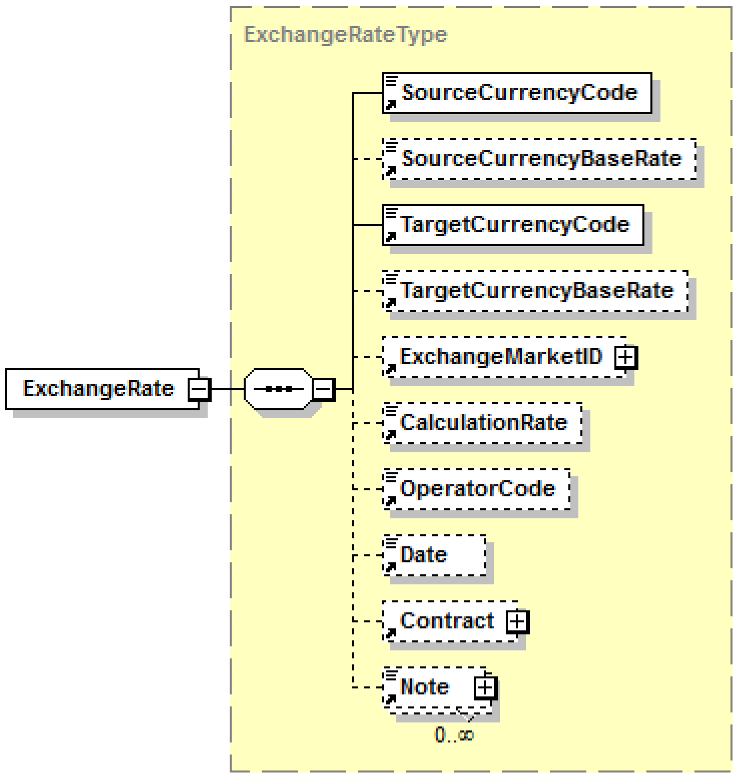
`See ExchangeRate for more information.`

Attributes

None.

4.82 Element: ExchangeRate

The `ExchangeRate` element expresses information about how currency conversion has been performed between two currencies.



The mandatory `SourceCurrencyCode` and `TargetCurrencyCode` express the source and target currencies involved in the exchange.

Optional `SourceCurrencyBaseRate` and `TargetCurrencyBaseRate` specify the unit base of the source and target currencies for currencies with small denominations. For example, in a case such as "**1 Turkish Lira = 0.000000716 US Dollars**", it is common to express conversions using a different base rate. A base rate of 1000000 for the lira would then give a calculation rate of 0.716.

The optional `ExchangeMarketID` identifies the currency exchange market from which the exchange rate is taken. The value is defined as a `LabeledIDType`.

The `CalculationRate` element records the factor used for conversion of an amount from the source currency to the target currency.

The `OperatorCode` identifies the operator that should be applied to obtain the target currency from the source currency. It must take one of two values: "`Multiply`" or "`Divide`", where "`Multiply`" is the default if/when `OperatorCode` is not present.

The date on which the exchange rate was in effect may be specified using a `Date` element.

A foreign exchange contract in which a rate of exchange has been agreed may be identified and described by a `Contract` element.

A repeatable `Note` element may be used to include any free form text pertinent to the exchange rate information. This element may contain notes or any other information that is intended for a human reader and is not contained explicitly in another structure. Notes may be repeated for information in alternative languages, but **MUST NOT** be repeated for any other reason.

Attributes

No attributes.

4.83 Element: ExpirationTime

Specifies a time at which something is deemed to have expired. For example, the expiration time for an ad order reservation would be recorded using the `ExpirationTime` element. The element is declared as `DateTimeDateType` and can take a date and time, or only a date as values.

Attributes

No attributes.

4.84 Element: FileName

The `FileName` element records a name for a file, recording the value as a `ShortStringType`. A use case for this element would be where a file is renamed on receipt with a local file name. In such a use case, the value of the `FileName` is a suggested name to use for the file.

Attributes

No attributes.

4.85 Attribute: firstTransmissionDateTime

The *`firstTransmissionDateTime`* attribute records a time stamp for the first transmission of an AdsML message. It is defined as a `DateTimeType`.

4.86 Element: FormalIdentifier

The `FormalIdentifier` element records a formal identifier. See for instance `CommunicationChannel.BaseType`.

Attributes

No attributes.

4.87 Element: Format

The `Format` element records the format used to represent data. The *`version`* attribute of `Format` can optionally be used to record the version number of that format. The `Format` element records format using the `FormatRootType` of data type `ShortTokenType`. As with all contexts where a root type is used, the `FormatRootType` can be restricted to a list of values defined by a controlled vocabulary.

For example, Format=”PDF/X1a”, version=”2003”.

Attributes

version (optional)

Records the version of the format as a `ShortStringType` data type.

4.88 Element: FormatProfile

The `FormatProfile` element is used to identify a specific profile or subset of a format, recording the value as a `ShortStringType` data type. For example, ‘MyAdAggregatorCo’ is using a subset of XHTML to represent ads on its website. This ‘MyAdAggregatorCo’ would be identified in the `FormatProfile` element.

Attributes

No attributes.

4.89 Element: FromThisPointOnPage

The `FromThisPointOnPage` element identifies the point on the page from which the x-y co-ordinates specifying the location of an ad on the page are taken to start.

The allowed values of the element are restricted to:

- '`TopLeft`' – the x-y co-ordinates identify the top left corner of the ad.
- '`TopRight`' – the x-y co-ordinates identify the top right corner of the ad.
- '`Center`' – the x-y co-ordinates identify the center of the ad.
- '`BottomLeft`' – the x-y co-ordinates identify the bottom left corner of the ad.
- '`BottomRight`' – the x-y co-ordinates identify the bottom right corner of the ad.

If `FromThisPointOnPage` is not specified then its value **SHOULD** be assumed to be `TopLeft`.

Attributes

No attributes.

4.90 Group: HumanCommunicationChannelsGroup

The `HumanCommunicationChannelsGroup` element group provides a choice between communication channel elements used to specify phone, physical address, Email, and other methods for communicating with human beings. Four elements are contained in the group:

The `CommunicationChannel.Phone` element is used to record a phone number with which a human being can be contacted.

The `CommunicationChannel.PhysicalAddress` element is used to record a physical address for a location at which a human being can be contacted.

The `CommunicationChannel.EMail` element is used to record an EMail address with which a human being can be contacted.

The `CommunicationChannel.Other` element is used to record other forms of communication channel with which a human being can be contacted, recording the address in a generic way.

4.91 Attribute group: i18nAttributes

The *`i18nAttributes`* group specifies a set of attributes that provides internationalization support by providing attributes for language and directionality. The AdsML approach follows the W3C's "Internationalization Tag Set (ITS) Version 1.0" (http://www.w3.org/TR/its/) by adding *`xml:lang`* and *`dir`* attributes. An AdsML specific *`source`* attribute is also provided to indicate the language version that is considered as the source or 'original text' in case of translations.

Attributes

xml:lang (optional)

Designed to identify the human language used in the scope of the element to which it's attached. This attribute must be set to a language identifier, as defined by IETF RFC 4646 (http://www.ietf.org/rfc/rfc4646.txt) or its successor.

dir (optional)

Designed to identify the direction of the language in the *xml:lang* attribute. It takes values from the TextDirectionsCV.

source (optional)

Designed to identify if the language specified in *xml:lang* is the source from which other available language versions have been translated.

4.92 Element: ID

The ID element specifies a general identifier of a context object. It has different content model depending on context.

Attributes

No attributes.

4.93 Element: Identifier

The Identifier element specifies a general identifier in for instance the PartyType type. It is declared as LabeledIDType.

Attributes

No attributes.

4.94 Attribute: inResponseToMessageID

The *inResponseToMessageID* attribute records the unique message identifier for the message to which a message is a response. The value of that message's *messageID* attribute is recorded in the *inResponseToMessageID* attribute.

The *inResponseToMessageID* value is recorded as a QIDType.

4.95 Attribute: inResponseToMessageCode

The *inResponseToMessageCode* attribute records the code value referencing the message code of the message a response is about. The value of that message's *messageCode* attribute is recorded in the *inResponseToMessageCode* attribute.

For instance, a request for a new order would have the message code 'AD-O' (AdOrder). A response to this request must record the same message code in the *inResponseToMessageCode* attribute.

4.96 Element: Instructions

The Instructions element records a set of instructions describing how the entity they qualify should be handled. The Instructions element is declared as RequirementSpecType.

Attributes

No attributes.

4.97 Element: InvoicingParty

The `InvoicingParty` element identifies the party taking the role as invoicer in a transaction.

Attributes

No attributes.

4.98 Element: IssueDate

The `IssueDate` element is a generic element to record the date and time on which a referenced document or similar business object was made available. It is defined as a `DateTimeDateType`.

Attributes

No attributes.

4.99 Element: IssueNumber

The `IssueNumber` is a number that is given on some types of credit card. In this case, each replacement card for a given card number has an "issue number", as in "2nd card issued to this person with this number.

Note that this issue number is only related to credit cards and not to an issue of a publication such as a magazine.

See `CreditCard` for more information about `IssueNumber`.

Attributes

No attributes.

4.100 Element: InvoicersReference

The `InvoicersReference` element takes any string value assigned by an invoicer as a reference identifier for a transaction or other business object.

See also `AuxiliaryReferences`.

Attributes

No attributes.

4.101 Element: JurisdictionRegionAddress

The `JurisdictionRegionAddress` element is used within `TaxScheme` to associate the tax scheme with particular information that identify and locate the geographic area in which a tax scheme applies.

See `TaxScheme` for further information.

Attributes

No attributes.

4.102 Type: LabeledIDType

The `LabeledIDType` type is used for specification of an identifier together with a code label that provides the origin or type of the identifier value.

The `IDLabel` element captures the label, such as DUNS (for Dun and Bradstreet organizational number) and the `IDValue` captures the actual identifier value string. The `IDValue` element is declared as `LongStringType`.

The `IDLabel` element is declared as `IDLabelRootType`; as with all contexts where a root type is used, the `IDLabelRootType` can be restricted to a list of values defined by a controlled vocabulary.

For example, if a DUNS number were being recorded then the identifier string would be recorded as the `Value` and the `Label` would identify the value as being '`DUNS`'.

Attributes

No attributes.

4.103 Element: LabeledProperty

The `LabeledProperty` element is defined as a `LabeledUnlimitedValueType` and used within `Properties`. See *E-Commerce Usage Rules and Guidelines* for further information.

Attributes

No attributes.

4.104 Type: LabeledUnlimitedValueType

The `LabeledUnlimitedValueType` type is used for specification of a value together with a code label that provides the origin or type of the value. It is identical to the `LabeledValueType` with the exception that the `Value` child element is defined as an unlimited string.

Attributes

No attributes.

4.105 Type: LabeledValueType

The `LabeledValueType` type is used for specification of a value together with a code label that provides the origin or type of the value.

The `Value` element records a value. The `Value` element is declared as `LongStringType`.

The `Label` element records a label that describes the origin of the value. The `Label` element is declared as `CodeRootType`; as with all contexts where a root type is used, the `CodeRootType` can be restricted to a list of values defined by a controlled vocabulary.

The `Description` element provides a human readable descriptive text of a codified value. It may be repeated to capture descriptions in alternative languages.

Attributes

No attributes.

4.106 Attribute: lastReceivedMessageID

The `lastReceivedMessageID` attribute records the unique message identifier for the last message relevant to the flow of a message exchange and so identifies the message that has most recently been received by the sender of the message in that message exchange. The value of the last received message's `messageID` attribute is recorded in the `lastReceivedMessageID` attribute.

The `lastReceivedMessageID` is recorded as a `QIDType`.

4.107 Element: MaterialsPreparerParty

The `MaterialsPreparerParty` element identifies the party who has created or prepared a set of ad materials for publication. For example a Repro House or Full Service Agency.

The `MaterialsPreparerParty` element is declared as a `PartyType`.

Attributes

No attributes.

4.108 Element: MaterialsRecipientParty

The `MaterialsRecipientParty` element identifies the party that is the intended recipient of the materials in a materials delivery.

The `MaterialsRecipientParty` element is declared as a `PartyType`.

Attributes

No attributes.

4.109 Element: MediaType

The `MediaType` element should be used to record the type of media. It is defined as a `CodeType` and can be validated against a user defined controlled vocabulary. Typical values are online, newspaper, outdoor or broadcast.

Attributes

No attributes.

4.110 Element: MerchantCode

The `MerchantCode` indicates a code for a company that has processed a credit card payment. This element will not be used where the credit card details are being passed for later processing.

See `CreditCard` for further details.

Attributes

No attributes.

4.111 Element: MIMEType

Records the `MIMEType`, of a content file as a string of data type `LongStringType`.

Attributes

No attributes.

4.112 Element: Name

The `Name` element records a value representing a name by which something is commonly known and so the name can be used for identification purposes. Records the name as a string of data type `LongStringType` or `LongStringType.i18n` depending on context. The *`i18nAttributes`* group is only available in the latter case.

Attributes

i18nAttributes (optional)

The *`i18nAttributes`* group supports internationalization by providing attributes to record language, directionality and source.

4.113 Type: NamedPriceType

See `PriceComponent` for information.

Attributes

No attributes.

4.114 Element: NameOnCard

The full name of a credit card holder, as printed on the card.

See `CreditCard` for more information.

Attributes

No attributes.

4.115 Element: NatureOfResponse

The `NatureOfResponse` element should be used in responses to indicate how the response should be interpreted, in relation to the prior request. A typical use of this element would be when a publisher cannot fully accept an order as requested, but still does not see a need to completely deny the request. The `NatureOfResponse` element can then specify if the request was accepted "as-is" or with modifications.

The `NatureOfResponse` element is defined as a `CodeType` and may have controlled vocabularies for both `CodeList` and `Code` elements as well as an optional descriptive text.

See also `AdMessageResponseModule` for more information.

Attributes

No attributes.

4.116 Type: NegatableCodeType

The `NegatableCodeType` extends the `CodeType` to create a negatable version of the `CodeType`. The `CodeType` is extended to add a *`negated`* attribute.

The *`negated`* attribute allows code values to be negated. For instance, consider a case when a code '`P3`' means '`positioning on page 3`'. If the negated attribute is '`false`' (default) the required position is '`page 3`'. If the negated attribute is set to '`true`', the requirement is '`not page 3`'.

Attributes

negated (optional)

When set to '`true`', the semantics of the code is negated, i.e. '`not(ABC)`'. If the attribute is not specified in a message, the message **MUST** be interpreted as if the attribute had been given with a value of '`false`'.

4.117 Type: NegatableRequirementSpecType

The `NegatableRequirementSpecType` is a general structure used in several contexts. Typically, it is used to capture a set of requirements provided using an agreed machine-readable code value using the `Code` element and/or descriptive free text requiring human interpretation using the `Text` element.

Both `Code` and `Text` elements are repeatable and it **MUST** be considered to be a logical AND operator between the requirements.

The `Code` element is declared as `NegatableCodeType`. Code values can be negated using the *`negated`* attribute on the `Code` element. For instance, consider a case when a code '`P3`' means '`positioning on page 3`'. If the negated attribute is '`false`' (default) the required position is '`page 3`'. If the negated attribute is set to '`true`', the requirement is '`not page 3`'.

The `Text` element records a requirement as a free text string. The `Text` element is declared as `LongStringType.i18n`.

In the case that multilingual text is present – i.e. `Text` elements whose *`xml:lang`* attributes indicate that more than one human language is used – then the `Text` elements should first be filtered into specific language groups, all but one of which may be ignored. (Unless the Trading Partners have agreed otherwise, it can be assumed that the set of `Texts` in each language repeat the same information.) The logical AND operator is then applied to the `Text` instructions in the one selected language.

It is **RECOMMENDED** to use XML schema defined controlled vocabularies for the `Code` element. Even though the actual values are simple strings, the name of the type provides a label describing the value as well as acts as a signal that the value is used in accordance with the agreement between trading partners.

Attributes

No attributes.

4.118 Attribute: negated

The *`negated`* attribute is used to negate the semantics of the element content qualified by the attribute. The default value of the attribute is '`false`'; when set to '`true`' the semantics of the code are negated, i.e. '`not(ABC)`'. For example, in the case when a code '`P3`' means '`positioning on page 3`'. If the negated attribute is `'false'` (default) the required position is `'page 3'`. If the negated attribute is set to `'true'`, the requirement is `'not page 3'`.

The *`negated`* value is recorded as a `BooleanType`.

4.119 Element: Note

The `Note` element records general and unspecified text descriptions. The `Note` element is declared as `StringType.i18n` and thus supports language metadata according to the *`i18nAttributes`* group.

Attributes

i18nAttributes (optional)

The *`i18nAttributes`* group supports internationalization by providing attributes to record language, directionality and source.

4.120 Element: NoteLine

See the `NotesType` type.

Attributes

timeStamp (required)

The *`timeStamp`* attribute records the date and time at which the note was created, and is declared as `DateTimeType`.

author (required)

The *`author`* attribute records the name of the note's author, and is declared as `ShortStringType`*.*

i18nAttributes (optional)

The *`i18nAttributes`* group supports internationalization by providing attributes to record language, directionality and source.

4.121 Element: Notes

The `Notes` element records a set of notes intended for human interpretation that support decision making on how the entity they qualify should be handled. The `Notes` element is declared as `NotesType`.

Attributes

No attributes.

4.122 Type: NotesType

The `NotesType` type records information as human-readable text intended to be read and interpreted by a human.

The `NotesType` content model is a required and repeatable `NoteLine` element. A `NoteLine` element records the note information as text of `StringType.i18n`. In addition to the *`i18nAttributes`* it is qualified by two required attributes, *`timeStamp`* and *`author`*.

Attributes

No attributes.

4.123 Element: NumberOfUnits

The `NumberOfUnits` captures a number of units as a decimal value.

Attributes

No attributes.

4.124 Element: OperatorCode

The `OperatorCode` identifies the operator that should be applied to obtain the target currency from the source currency in an exchange rate structure. It must take one of two values: `“Multiply”` or `“Divide”`.

See `ExchangeRate` for more information.

Attributes

No attributes.

4.125 Element: OrderersReference

The `OrderersReference` element is used by a party ordering a transaction to record their own reference identifier value for the transaction or other business object. The value is recorded as a `LongNormalizedStringType`.

Note: the ‘orderer’ is not to be confused with the party placing a booking for an advertisement, who is its ‘buyer’.

Attributes

No attributes.

4.126 Type: OtherLabeledIDType

The `OtherLabeledIDType` type extends the `LabeledIDType` to create a version where a role can be associated with the labelled ID value. The `LabeledIDType` is extended to add a `Role` element. The role describes the part or function played by the ID value in the process or operational workflow where the ID is significant.

The `Role` element is declared as `RoleRootType`; as with all contexts where a root type is used, the `RoleRootType` can be restricted to a list of values defined by a controlled vocabulary.

Attributes

No attributes.

4.127 Element: OtherParty

The `OtherParty` is an extension of the `RelaxedPartyType` type to add a locally declared `Role` element. An `OtherParty` is intended to be used to record an organization that takes part in the advertisement process without being one of the primary parties.

See the `OtherPartyType` for more information.

Attributes

No attributes.

4.128 Type: OtherPartyType

The `OtherPartyType` type is an extension of the `RelaxedPartyType` type to add a locally declared `Role` element. It is used for associating a party with a role where the party's role is not provided by the context.

The `Role` element identifies the part or function performed by the other party in the process or operational workflow. The `Role` element is declared as `PartyRoleRootType`; as with all contexts where a root type is used, the `PartyRoleRootType` can be restricted to a list of values defined by a controlled vocabulary.

Attributes

No attributes.

4.129 Element: OtherReference

The `OtherReference` element is defined as a `ReferenceValueType`. Please see its description for more information.

Attributes

No attributes.

4.130 Element: PartyAddress

The `PartyAddress` is used within the PartyType structure and provides a content model for specifying contact information related to the party itself, for instance a street address, phone number or web address for a company.

The `Role` element specifies the role taken by the address. Values can be validated by controlled vocabularies, if required.

The `AllCommunicationChannelsGroup` elements specify where telephone, physical address, Email, and other methods for communicating with the party can be recorded.

If required, it is also possible to include application-specific data using the general `Properties` element.

Attributes

priority (optional)

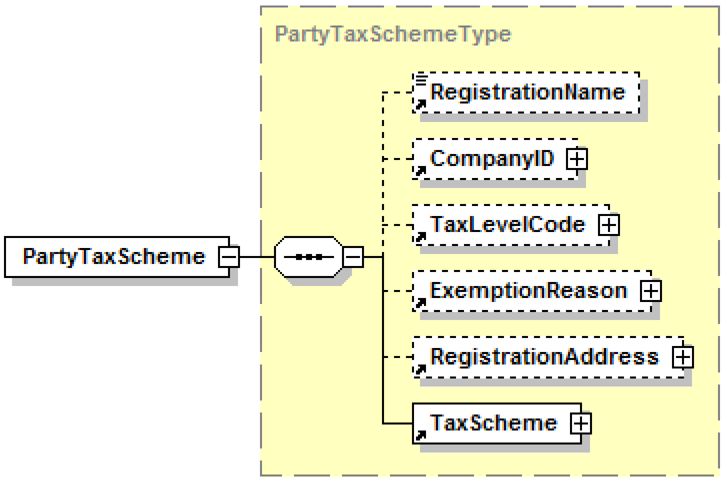
Assigns a priority rating to the `Contact` element. The priority rating is used to identify the sequence in which contacts should be contacted in the event that more than one `Contact` element is present.

i18nAttributes (optional)

The *`i18nAttributes`* group supports internationalization by providing attributes to record language, directionality and source.

4.131 Element: PartyTaxScheme

A PartyTaxScheme is used within a party structure to associate the party with a tax scheme, i.e. a particular type of tax and, by implication, the rules that apply according to that tax type. The element also includes other properties about the party that can be used in relation to the tax scheme.



The RegistrationName element records the name of the party as registered with the relevant tax authority. The party's identification as registered with and assigned by the authority can be recorded in the CompanyID element. Note that this identifier may be different from the identifiers of the party as expressed in the Identifier or AuxiliaryReferences elements. For instance, when the tax scheme is VAT (sales tax), the CompanyID should record a VAT registration number, whereas Identifier might contain a DUNS number or any other appropriate identifier.

The TaxLevelCode element allows a section or role within the tax scheme that applies to the party to be specified.

In cases where a party may be tax exempt, the ExemptionReason element records a code that explains the reason for exemption.

The RegistrationAddress element associates the party tax scheme with the registered address of the context party.

The TaxScheme element provides further details such as identification of the tax scheme that the party is associated with. See TaxScheme for more information.

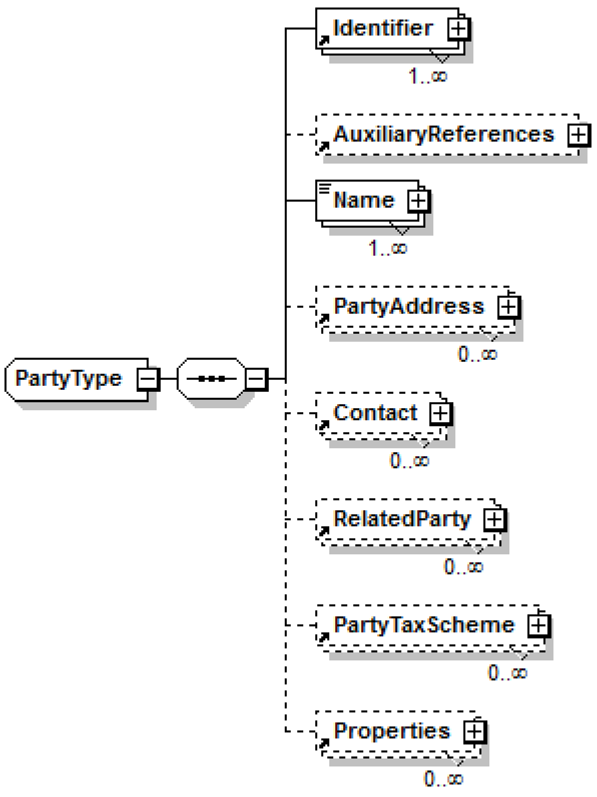
A general *Overview of Taxation Structures* is provided above in this document.

Attributes

None.

4.132 Type: PartyType

The PartyType is a generic component for specification of various parties, i.e. organizations and persons, which appear in a message.



A party has a mandatory and repeatable identification structure expressed using `Identifier` elements where labeled ID values are captured. The `AuxiliaryReferences` structure adds capability to also specify alternative references assigned by various trading parties playing different roles.

The `PartyType` has a mandatory `Name` element, which may be repeated to record the name using several alternative languages. It includes optional `PartyAddress` and `Contact` elements for name and various types of addressing information relatind to the party and/or the party's contact persons.

The `RelatedParty` element can be used to express relationship to other parties. Typically, this can be a parent company or a sub division. For each related party, the relationship to the main party should be specified using the `Role` element (child to `RelatedParty`).

The intention is that related parties express a relationship which is independent of the particular business object context (i.e. it is not directly related to "this order" or "this invoice"), while `OtherParty/Role` expresses a relationship between that other party and this business object. We do not expect related parties to be used very often, while `OtherParty` is quite common.

An optional `PartyTaxScheme` element is used to associate the party with a tax scheme, i.e. a particular type of tax and, by implication, the rules that apply according to that tax type. The element also includes other properties about the party that can be used in relation to the tax scheme.

If required, it is also possible to include application-specific data using the general `Properties` element.

Attributes

No attributes.

4.133 Element: PayeeParty

The `PayeeParty` element identifies a party taking the role as payee in a transaction. It is defined as a `PartyType`.

Attributes

No attributes.

4.134 Element: PayerParty

The `PayerParty` element identifies a party taking the role as payer in a transaction.

The `Name` element should be the name of the legal entity that is responsible for making the payment.

Attributes

No attributes.

4.135 Element: PayersReference

The `PayersReference` element is used by a party acting as the payer to record its own reference identifier value for a transaction. The value is recorded as a `LongNormalizedStringType`.

See also `AuxiliaryReferences`.

Attributes

No attributes.

4.136 Element: PaymentDueDate

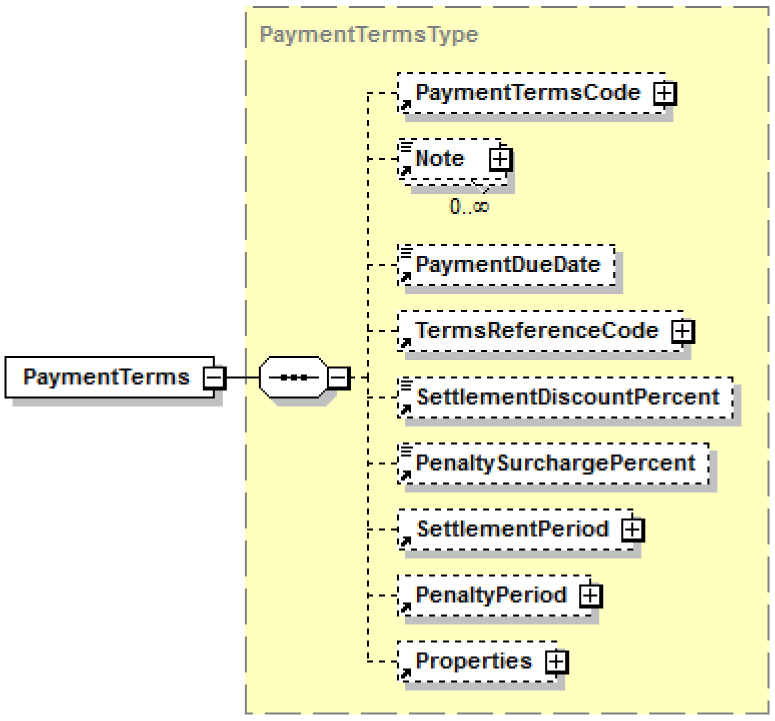
The `PaymentDueDate` element can be used to record a point in time at which a payment is to be made.

Attributes

No attributes.

4.137 Element: PaymentTerms

The `PaymentTerms` element typically associates an invoice or order with the payment terms and conditions applicable/offered.



A `PaymentTermsCode` element can be used to identify the set of conditions attached to an agreement or contract relating to a payment.

A repeatable `Note` element may be used to include any free form text pertinent to the payment terms information. This element may contain notes or any other similar information intended for a human reader and that is not contained explicitly in another structure. It may be repeated to include the text in alternative languages, but **MUST NOT** be repeated for any other reason.

The point in time at which the payment is to be made can be recorded in the `PaymentDueDate`

The `TermsReferenceCode` element records the event from which terms are offered for a length of time, identified by a standard code, e.g. "`InvoiceTransmissionDate`" or "`RunDate`" from the AdsML Payment Terms Reference Event CV.

Settlement discount rate (percentage) offered for payment within the settlement period can be provided in the `SettlementDiscountPercent` element. The settlement period is defined in the `SettlementPeriod` element.

The `PenaltyPeriod` element associates the payment terms with the period after which a penalty is charged. The penalty rate (percentage) charged for late payment may be held in the `PenaltySurchargePercent` element.

The optional `adsml:Properties` element can be used to define application-specific extensions.

Attributes

No attributes.

4.138 Element: PaymentTermsCode

See `PaymentTerms` for information.

Attributes

No attributes.

4.139 Element: PenaltyPeriod

The `PenaltyPeriod` element can be used to describe a period after which a penalty is charged. It is defined as a `PeriodType` including start and end date-times and/or a duration of time.

See also `PaymentTerms`.

Attributes

No attributes.

4.140 Element: PenaltySurchargePercent

The `PenaltySurchargePercent` element can be used to specify a penalty rate (percentage) charged for late payment.

See also `PaymentTerms`.

Attributes

No attributes.

4.141 Type: PeriodType

The `PeriodType` type is used for recording time periods as either a span between two given date times, or as duration measure such as "1 month."

Date times can be specified using the `StartDateTime` and `EndDateTime` elements. A duration may be expressed using a `DurationMeasure` element, defined as a `DecimalMeasurementType` accepting a value qualified by a unit of measure.

An additional description of the period may be provided using a `Description` element.

Attributes

No attributes.

4.142 Type: PhoneAddressType

The `PhoneAddressType` extends the `CommunicationChannel.BaseType` type to define a telephone address.

The inherited `Usage` element identifies the usage of the phone address – e.g. ISDN, voice. For example, for artwork delivery the `Type` could be "`data`" and `Usage` "`ISDN`"; for example for mobile phone `Type` could be "`voice`" and `Usage` "`personal`"

The `PhoneAddressType` content model is a sequence of required `Type`, `PhoneNumber`, and `optional CountryCode`, `AreaCode`, `SubscriberNo`, and `Extension` elements. The required elements enable a telephone number to be identified and classified; the optional elements allow a telephone number to be broken down in to its constituent components.

The `Type` element classifies the type of phone line being used. For example, identifying a phone address as being used for verbal, fax, or data communications. The `Type` element is declared as `PhoneTypeCV`.

The `PhoneNumber` element records the phone number as a single string. The `PhoneNumber` element is declared as `ShortStringType`.

The `CountryCode` element records that part of a phone number that identifies the country component of a phone number. The `CountryCode` element is declared as `ShortStringType`.

The `AreaCode` element records that part of a phone number that identifies the local area code component of a phone number. The `AreaCode` element is declared as `ShortStringType`.

The `SubscriberNo` element records that part of a phone number that identifies the local number component of a phone number. The `SubscriberNo` element is declared as `ShortStringType`.

The `Extension` element records that part of a phone number that identifies an extension number component of a phone number. The `Extension` element is declared as `ShortStringType`.

Attributes

No attributes.

4.143 Type: PhysicalAddressType

The `PhysicalAddressType` extends the `CommunicationChannel.Base` type to define an address describing the location of a physical structure in a geographical location.

The `PhysicalAddressType` content model is a sequence of optional `Department`, optional to four occurrences of `Street`, optional `POBox`, `ZipPostalCode`, `City`, `StateProvince`, `CountryName`, and `CountryCode` elements.

The `Department` element records a departmental name component of an address. The element is used to identify a specific department inside an organisation when the address is describing the location of an organisation with a departmental structure. The `Department` element is declared as `ShortStringType`.

The `Street` element records a street line component of a physical address as a single string and is limited to a maximum of four (4) occurrences. The `Street` element is declared as `LongStringType`.

The `POBox` element records a postal box address component of a physical address. The `POBox` element is declared as `ShortStringType`.

The `ZipPostalCode` element records a ‘zip’ or ‘post code’ component of a physical address. The `ZipPostalCode` element is declared as `ShortStringType`.

The `City` element records the name of the town or city that is the urban location component of a physical address. The `City` element is declared as `ShortStringType`.

The `CountryName` element records the name of the country that is the international geographical location component of a physical address. The `CountryName` element is declared as `ShortStringType`.

The `CountryCode` element records a code identifying the country that is the international geographical location component of a physical address. The `CountryCode` element is declared as `CodeRootType`; as with all contexts where a root type is used, the `CodeRootType` can be restricted to a list of values defined by a controlled vocabulary.

Attributes

No attributes.

4.144 Element: Percent

The `Percent` is a general element for capturing percentage values that can be used in several contexts. See for instance `CalculationSpecification`. The value is a decimal value that **MUST** be interpreted as a percentage value. For example, a value of '`0.75`' **MUST** be interpreted as 0.75%.

Attributes

No attributes.

4.145 Element: PositionOnPage

The `PositionOnPage` element contains a set of optional elements that record the actual positioning of an ad on a newspaper or magazine page. The positioning can be recorded in codified or text form, using optional `Code` or `Text` elements (See `RequirementSpecType` for a definition of these elements).

If required, the position on the page in exact x-y co-ordinates can be specified using the `AbsolutePosition` element.

Attributes

No attributes.

4.146 Type: PositionOnPageType

See description for the `PositionOnPage` element.

Attributes

No attributes.

4.147 Element: PriceComponent

`PriceComponent` is used to specify a single component in a price structure. By including a "stack" of 1 or more price components, the sender can identify all of the elements (including chargeable units of items, discounts, surcharges, etc.) that directly contribute to the total price.

The mandatory `PriceComponentName` element should be used to capture a shorter name or code for the price component. The code can be validated against a controlled vocabulary.

The mandatory `Amount` element specifies the final resulting amount for the price component.

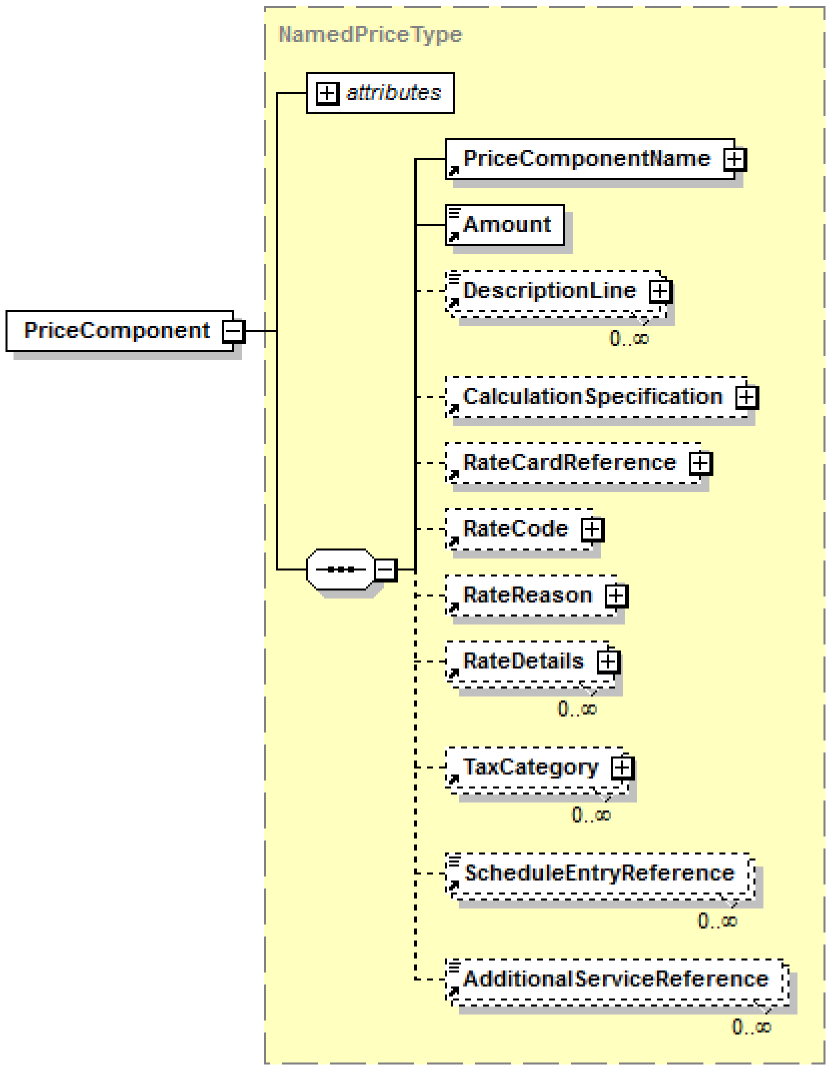
An optional short description of the price amount can be given using the `DescriptionLine` element. It is repeatable to allow description lines in alternative languages.

If required, the optional `CalculationSpecification` can be used to declare the base components and facts that were used to achieve the value in the `Amount` element.

The `CalculationSpecification` can either specify a price per unit structure, or a percentage and base price structure. The price per unit includes `Unit`, NumberOfUnits and PricePerUnit elements that can handle situations like "5 columns @ 100" where "5" is the number of units, "columns" the unit and "100" the price per unit.

For situations where the price is expressed in terms of a multiple number of units, e.g. "CPM" or cost-per-thousand pricing, it is possible to specify the amount by which the specified price should be divided using the *divisor* attribute of the PricePerUnit element. CPM pricing, for example, is expressed by placing the value 1000 in the *divisor* attribute.

For situations where the amount is calculated as a percentage of another amount, it is possible to specify the percentage and base price of that calculation using the Percentage and BasePrice elements.



A means to convey the source of the rate that was used in the calculation leading to the concluding sum as expressed in the Amount, can be provided in a set of elements referring to a rate card and rates:

- RateCardReference – A reference to a rate card
- RateCode – A rate code
- RateReason – A reason for why a rate was applied

- `RateDetails` – A repeatable description of any other rate details

The `TaxCategory` element can be used to associate information about how taxes apply to the price component.

In order to indicate that the `PriceComponent` applies to particular services and/or schedule entries (e.g. insertion dates), two optional and repeatable references are available, the `AdditionalServiceReference` and `ScheduleEntryReference` respectively.

Attributes

sequenceNo (required)

Used to indicate the intended sequence of price component siblings. Note that when used together with `SubTotal` element siblings, the sequence number **MUST** be unique for the union set of `PriceComponent` and `SubTotal` elements (i.e. numbered in a single sequence).

4.148 Element: PriceComponentName

A name of a price component. See `PriceComponent` for more information.

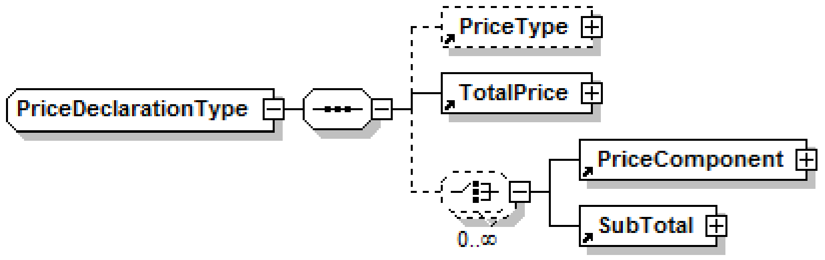
Attributes

No attributes.

4.149 Type: PriceDeclarationType

The `PriceDeclarationType` type is a complete pricing structure with total price, and an optional breakdown of details using price components and subtotals.

The `PriceDeclarationType` type can be used in several different contexts. In order to be able to handle different usage scenarios, `PriceDeclarationType` includes an optional `PriceType` element that, based on `CodeType`, allows specification of a code and optional description that declares the type of price, e.g. "Confirmed", "NotToExceed", "Estimated".



One element is required: `TotalPrice` defines the total amount of the price.

It is also possible to provide more detailed price information where the total price is broken down into components: a set of ordered line items with sub totals using the `PriceComponent` and `SubTotal` elements.

For cases when it is required to also define the currency used for the price, see the `CurrencyPriceDeclarationType`, an extension of the `PriceDeclarationType` with currency information.

Attributes

No attributes.

4.150 Element: PricePerUnit

See `PriceComponent`.

Attributes

divisor (optional)

The divisor attribute can be used to indicate that the price should be divided by a divisor when applied to an individual unit. For instance, a value '1000' can be used to indicate Cost Per Thousands (CPM).

4.151 Element: PriceType

The `PriceType` element can be used to record the type of a price. It is defined as a `CodeType`.

See for instance `PriceDeclarationType`.

Attributes

No attributes.

4.152 Attribute: priority

The *`priority`* attribute records a priority rating for the element that it qualifies. The priority attribute is used to describe a rating of the priority or sequence with which the qualified element should be handled in relation to other priority qualified elements of the same element type. For example, in the `PartyType` context where more than one `Contact` element can be specified, the priority rating is used to identify the sequence in which contacts should be contacted in the event that more than one `Contact` element is present.

The priority rating is recorded as an integer value in the range of 1-9 inclusive, using the `PriorityType` controlled vocabulary defined in the AdsML Type Library.

4.153 Element: Priority

The `Priority` element records a priority rating that specifies the priority with which the message or message component with which the priority is associated is to be handled. The priority rating is recorded as an integer value in the range of 1-9 inclusive, using the `PriorityType` controlled vocabulary.

Attributes

No attributes.

4.154 Element: ProofingParty

Defined as a `PartyType`, the `ProofingParty` is a party that is distributing proof information and has overall business responsibility for the contents of a proofing message as a whole.

Attributes

No attributes.

4.155 Element: ProofersReference

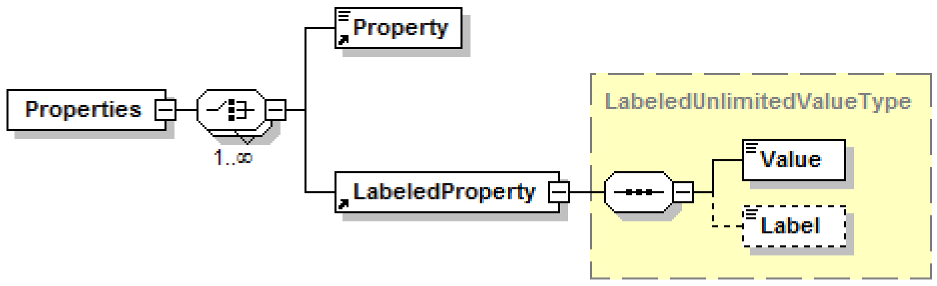
The `ProofersReference` can be used to record a reference string for a Proofing Party, defined as a `LongNormalizedStringType`.

Attributes

No attributes.

4.156 Element: Properties

The `Properties` element allows user-specific properties to be recorded as a simple name/value pair using a sequence of repeatable `Property` or `LabeledProperty` children elements. Each property is recorded by a separate `Property` or `LabeledProperty` element using a controlled vocabulary defined by the user.



When using the `Property` element, specific types of property **MUST** be derived from the `Property` element's root type in an extension XML Schema and substituted for the `PropertyRootType` in instance documents by use of the *`xsi:type`* type cast mechanism. This enforcement is imposed to restrict property use to those properties agreed and formally defined in XML Schema by trading partners.

The name of the property is provided by the name of the derived property type that has been specified as the property's data type in the *`xsi:type`* attribute of the `Property` element.

The `LabeledProperty` element provides a less strict approach for user defined properties, not requiring any XML Schema defined types nor use of the *`xsi:type`* attribute. Instead, an agreed and descriptive name of the property may be provided using a `Label` child element. It is **RECOMMENDED** to label properties with a unique name to avoid name clashes. See *E-Commerce Usage Rules and Guidelines* for further information.

Attributes

No attributes.

4.157 Element: Property

Records a user-defined property as a name:value pair. The `Property` element is declared as `PropertyRootType`. When using the `Property` element, specific types of property **MUST** be derived from the `Property` element's root type and substituted for the `PropertyRootType` in instance documents. This is required because of the way the property name:value pair is constructed. The name of the property is provided by the name of the derived property type that has been specified as the property's data type in the *`xsi:type`* attribute of the `Property` element. The value of the property is recorded as element content of

ShortTokenType data type. See PropertyRootType for the root type definition. See *E-Commerce Usage Rules and Guidelines* for further information.

Attributes

No attributes defined by Property *element. If used, then the* *xsi:type* *attribute will be present.*

4.158 Element: ProvenanceParty

Defined as a RelaxedPartyType, the ProvenanceParty is a party that takes responsibility for (parts of) the proofing information in a transaction, e.g. a physical tearsheet or affidavit.

Attributes

No attributes.

4.159 Element: PublisherParty

The PublisherParty element identifies the party with the business responsibility for publishing an advertisement. The PublisherParty element is defined as a PartyType.

Attributes

No attributes.

4.160 Element: PublishersReference

The PublishersReference element is used by a party publishing an advertisement to record their own reference identifier value. The value is recorded as a LongNormalizedStringType.

Attributes

No attributes.

4.161 Element: PurchaseOrderReference

The PurchaseOrderReference element may be used to record references to purchase orders in a variety of contexts.

Attributes

No attributes.

4.162 Element: RateCardReference

The RateCardReference element can be used to record a rate card reference code. It is defined as a CodeType.

Attributes

No attributes.

4.163 Element: RateCode

The RateCode element can be used to record a rate code. It is defined as a CodeType.

`RateCode` is intended to be used in contexts where machine-processable information is required, a corresponding element `RateReference` is available for contexts where simple string is acceptable.

Attributes

No attributes.

4.164 Element: RateDetails

The `RateDetails` element is a generic structure that can be used to record details about a rate used in a price calculation. It is defined as a CodeType.

Attributes

No attributes.

4.165 Element: RateReason

The `RateReason` element is a generic structure that can be used to record a reason for why a rate was applied in a price calculation. It is defined as a CodeType.

Attributes

No attributes.

4.166 Element: RateReference

The `RateReference` element records a reference string to a rate (or a level).

Attributes

No attributes.

4.167 Element: ReasonForCancellation

The `ReasonForCancellation` element is used to describe why a previous request or agreement should be cancelled. The reason can be described using a mandatory machine readable code, together with an optional text description. Both values can use a controlled vocabulary for validation.

The `ReasonForCancellation` element is declared as `CodeType`.

Attributes

No attributes.

4.168 Element: ReasonForDenial

The `ReasonforDenial` element is used to describe a reason for a denied action. It is based on the `CodeType` using a mandatory machine readable code, together with an optional text description. Both values can use a controlled vocabulary for validation.

Attributes

No attributes.

4.169 Element: ReceiversReference

The `ReceiversReference` element is used by a party receiving a materials delivery to record their own reference identifier value for a materials delivery transaction. The value is recorded as a `LongNormalizedStringType`.

Attributes

No attributes.

4.170 Type: ReferenceValueType

The `ReferenceValueType` extends the `LabeledValueType` to create a reference identifier that enables the value to be associated with the party that created the reference identifier value and the party to whom that value is likely to be of interest to.

The `ReferenceValueType` content model is an optional `CreatedBy` element and an optional `OfInterestTo` element.

The `CreatedBy` element identifies the party that 'created' or 'assigned' the reference identifier value; it is declared as `RelaxedPartyType`.

The `OfInterestTo` element identifies the party to whom the reference identifier value is of interest to, i.e. to whom the value is meaningful and carries business significance in their workflow; it is declared as `RelaxedPartyType`. The `CreatedBy` and `OfInterestTo` elements implicitly represent a workflow between two parties by asserting a relationship between them.

For example, in a workflow where a Buyer of Advertising commissions a Producer of Ad Materials to produce ad content, the Producer of Ad Materials may assign its own reference identifier to the ad content, this identifier being used for reconciliation purposes during the approval cycle between buyer and producer. In such a scenario, the `CreatedBy` element would identify the Producer and the `OfInterestTo` element would identify the Buyer.

Attributes

No attributes.

4.171 Element: RegistrationAddress

The `RegistrationAddress` element is used to record an officially registered address of, for instance, a company. It is defined as a `PhysicalAddressType`.

See also `PartyTaxSceheme` for more information.

Attributes

No attributes.

4.172 Element: RegistrationName

The `RegistrationName` element can be used to record an officially registered name of, for instance, a company such as a name registered with the tax authority.

See also `PartyTaxScheme` for more information.

Attributes

No attributes.

4.173 Element: RelatedParty

The `RelatedParty` element is defined used in party types to express a relationship between a party and another party. The relationship is expressed using a `Role` element in the `RelatedParty` structure.

The intention is that related parties express a relationship which is independent of the particular business object context (i.e. it is not directly related to "this order" or "this invoice"), while `OtherParty/Role` expresses a relationship between that other party and this business object. We do not expect related parties to be used very often, while `OtherParty` is quite common.

The `RelatedParty` content model is similar to the `PartyType` content model, with the difference that it has a mandatory Role, but is lacking the ability to relate yet another party, i.e it does not include a `RelatedParty` structure.

Depending on context, `RelatedParty` exists in both a relaxed and a non-relaxed version, i.e. with and without mandatory `Identifier`.

Attributes

No attributes.

4.174 Element: RelationshipName

The `RelationshipName` element can be used to record a code defining a relationship between objects.

Attributes

No attributes.

4.175 Type: RelaxedPartyType

The `RelaxedPartyType` is a generic component for specification of various parties, i.e. organizations and persons, which appear in a message.

The `RelaxedPartyType` content model is similar to the `PartyType` content model, with the difference that the `Identifier` is optional. Thus, the `RelaxedPartyType` is used for elements where an `Identifier` of the party is not mandatory.

Attributes

No attributes.

4.176 Element: RequestDenied

The `RequestDenied` element is part of business level responses and it should include an explanation of the reason(s) why a request was denied using the `ReasonForDenial` element. The reasons for denial can be described using a mandatory machine readable code, together with an optional text description. Both values can use a controlled vocabulary for validation.

Attributes

No attributes.

4.177 Type: RequirementSpecType

The `RequirementSpecType` is a general structure used in several contexts. Typically, it used to capture a set of requirements provided using an agreed machine-readable code value using the `Code` element and/or in descriptive free text requiring human intervention using the `Text` element. Both `Code` and `Text` elements are repeatable and it **MUST** be considered to be a logical AND operator between the requirements.

The `Code` element is declared as `CodeType`.

The `Text` element records a requirement as a free text string. The `Text` element is declared as `LongStringType.i18n`.

In the case that multilingual text is present – i.e. `Text` elements whose *`xml:lang`* attributes indicate that more than one human language is used – then the `Text` elements should first be filtered into specific language groups, all but one of which may be ignored. (Unless the Trading Partners have agreed otherwise, it can be assumed that the set of `Texts` in each language repeat the same information.) The logical AND operator is then applied to the `Text` instructions in the one selected language.

It is **RECOMMENDED** to use XML schema defined controlled vocabularies for the `Code` element. Even though the actual values are simple strings, the name of the type provides a label describing the value as well as acts as a signal that the value is used in accordance with the agreement between trading partners.

Attributes

No attributes.

4.178 Element: RevisionIdentifier

The `RevisionIdentifier` element may be used in various contexts to hold a revision number or other identifier of a revision. It is defined as a `ShortStringType`.

It is **RECOMMENDED** that the initial instance of a business object (e.g. a new Reservation or Order) **SHOULD** have a revision number of "`0`", and that each subsequent revision to the object **SHOULD** increment its revision number by 1. But note that in the AdsML Framework the `RevisionIdentifier` element is not included in the initial request message, so the first value that is actually transmitted will be "`1`".

Attributes

No attributes.

4.179 Element: Role

The `Role` element describes a part or function performed by the entity associated with the role.

The `Role` element is declared as `RoleRootType`; as with all contexts where a root type is used, the `RoleRootType` can be restricted to a list of values defined by a controlled vocabulary.

Attributes

No attributes.

4.180 Element: RoundingAmount

The `RoundingAmount` element is defined as an `AmountType` and can be used to express rounding amounts in various contexts.

Attributes

No attributes.

4.181 Attribute: schemaVersion

The *`schemaVersion`* attribute records the version of the schema to which an instance document conforms. For example, if an instance conforms to version 1.0.2 of a schema, then the *`schemaVersion`* attribute would take that value. The *`schemaVersion`* attribute is intended to support the major and minor versioning policy of AdsML Schema and to enable an application processing an instance document to interrogate the value of the *`schemaVersion`* attribute in order to identify the exact schema with which to validate the instance.

The *`schemaVersion`* is recorded as a `SchemaVersionType`.

4.182 Attribute: schemaProfile

The *`schemaProfile`* attribute records a unique name identifier of a usage profile of an AdsML standard to which an instance document conforms.

The *`schemaProfile`* is recorded as a `VersionedQIDType`.

4.183 Element: SectionReference

The `SectionReference` element can be used to record a reference string to a section of a document such as a contract or other complex object. It is defined as a `ShortStringType`.

Attributes

No attributes.

4.184 Element: SellingParty

Defined as a `PartyType`, the `SellingParty` is a party taking the role of a seller in a transaction.

Attributes

No attributes.

4.185 Attribute: sendCount

The *`sendCount`* attribute records a sequence number as a positive integer for a sequence of possible re-transmission of an AdsML message.

4.186 Element: SellersReference

The `SellersReference` element is used by a party acting as the seller (i.e. the 'publisher') of advertising space to record their own reference identifier value for an ad order transaction. The value is recorded as a `LongNormalizedStringType`.

See also `AuxiliaryReferences`.

Attributes

No attributes.

4.187 Attribute: sequenceNo

The *`sequenceNo`* attribute records a positive integer value used to identify the sequential positioning of an element in a set of elements in the same context that are also qualified by *`sequenceNo`* attributes.

For example, in the event that a booking contains multiple placements, the sequence number can be used to identify the sequential ordering that applies to that set of placements.

The *`sequenceNo`* is recorded as a `PositiveIntegerType`.

4.188 Element: ServiceCode

The `ServiceCode` element records a code representing an `AdditionalService`. The element is declared as `CodeType`.

See `AdditionalService` for further information.

Attributes

No attributes.

4.189 Element: SettlementDiscountPercent

The `SettlementDiscountPercent` element can be used to specify a settlement discount rate (percentage) offered for payment within the settlement period (`SettlementPeriod` element).

See also `PaymentTerms` for more information.

Attributes

No attributes.

4.190 Element: SettlementPeriod

The `SettlementPeriod` element is used to specify a period of time. It is used together with the `SettlementDiscountPercent` element within the `PaymentTerms` structure to specify a time period for which a discount is offered.

See also `PaymentTerms` for more information.

Attributes

No attributes.

4.191 Type: SinglePriceType

The `SinglePriceType` type holds a single `Amount` and an optional and repeatable `DescriptionLine` element and is typically used to express a value and a descriptive text. The `DescriptionLine` is repeatable to allow texts in alternative languages, but **MUST NOT** be repeated for any other reason.

See for instance `PriceDeclarationType`.

Attributes

No attributes.

4.192 Element: SourceCurrencyBaseRate

The `SourceCurrencyBaseRate` element is used within the `ExchangeRate` structure to specify the unit base of the source currency for currencies with small denominations.

See `ExchangeRate` for more information.

Attributes

No attributes.

4.193 Element: SourceCurrencyCode

The `SourceCurrencyCode` is used within the `ExchangeRate` structure to record a currency code. It is defined as a `CurrencyCodeRootType`.

See `ExchangeRate` for more information.

Attributes

No attributes.

4.194 Element: SpecialRequirements

The `SpecialRequirements` is used in several contexts to capture additional requirements. It is declared as `NegatableRequirementSpecType`.

Attributes

No attributes.

4.195 Element: Specifications

The `Specifications` is used in several contexts to capture additional requirements. It is based on the `NegatableRequirementSpecType` type.

Attributes

No attributes.

4.196 Element: StartDateTime

The `StartDateTime` element records a date or a date time. See also `EndDateTime` and `PeriodType`.

Attributes

No attributes.

4.197 Element: Status

The `Status` element records the current status of the event or set of information that it qualifies. For example, a future content delivery can be given a status of '`pending`' and an acknowledgement for a successfully retrieved content delivery can be given a status of '`retrieved`'. The status is recorded in code form using the `CodeType`.

The optional and repeatable `StatusQualifier` child element may be used to further qualify the status value. For instance, for the future content delivery case above with a status value of 'pending', two possible status qualifiers might have the values `'waiting for booking'` and `'waiting for material's due date'` providing further details behind the main status code. The `StatusQualifier` is defined as a `CodeType`.

Attributes

No attributes.

4.198 Element: StatusDate

The `StatusDate` element is intended to be used to record a business significant datetime for a status message to be used in addition to other datetime values of more technical importance such as *`transmissionDateTime`* or *`messageAssembledTime`*.

Attributes

No attributes.

4.199 Element: StatusQualifier

See the `Status` element for more information.

Attributes

No attributes.

4.200 Element: SubTotal

The `SubTotal` is used to specify a subtotal of a set of price components in a `PriceDeclarationType` type.

The optional `SubTotalName` element provides the possibility to capture a shorter name or code for the sub total. The code can be validated against a controlled vocabulary.

An optional short description of the price amount can be given using the `DescriptionLine` element. The element can be repeated to provide the description in alternative languages.

Attributes

sequenceNo (required)

Used to indicate the intended sequence of sub total and price component siblings. Note that when used together with `PriceComponent` element siblings, the sequence number **MUST** be unique for the union set of `PriceComponent` and `SubTotal` elements (i.e. numbered in a single sequence).

4.201 Element: SubTotalName

The `SubTotalName` records a name of a sub total component line in a price declaration. See `SubTotal` for more information.

Attributes

No attributes.

4.202 Element: TargetCurrencyBaseRate

The `TargetCurrencyBaseRate` element specifies the unit base of the target currency for currencies with small denominations. It is used within the `ExchangeRate` structure.

See `ExchangeRate` for more information.

Attributes

No attributesattributes.

4.203 Element: TargetCurrencyCode

The `TargetCurrencyCode` records a currency code; it is defined as a `CurrencyCodeRootType`. It is used within the `ExchangeRate` structure.

See `ExchangeRate` for more information.

Attributes

No attributes.

4.204 Element: TaxAmount

The `TaxAmount` records a total amount of tax.

See `TaxTotalType` for further information.

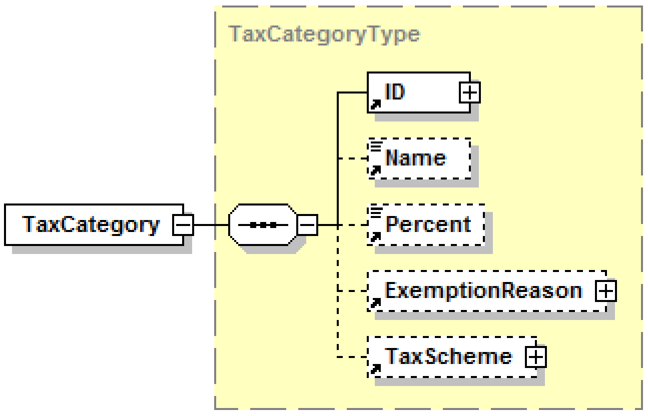
Attributes

No attributes.

4.205 Element: TaxCategory

The `TaxCategory` element is used in a variety of contexts (including price components, order specifications and financial document subtotals) to associate the parent structure with information about how taxes apply to it.

The `ID` element identifies the tax category by a code and, by implication, often identifies the tax rate that applies. For example: "`NotTaxable`" or "`StandardRate`".



The `Percent` element defines the tax rate as a percentage.

If the context object is tax exempt, the optional `ExemptionReason` element holds text that explains the reason.

The `TaxScheme` element provides further details such as identification of the tax scheme with which the tax category is associated. See `TaxScheme` for more information.

A general *Overview of Taxation Structures* is provided in a separate section above in this document.

Attributes

No attributes.

4.206 Element: TaxLevelCode

The `TaxLevelCode` element is used to define a section or a role within a tax scheme that applies to a particular party. See also PartyTaxScheme.

Attributes

No attributes.

4.207 Element: TaxPointDate

The `TaxPointDate` element provides an explicit date for tax purposes in accordance with applicable tax regulations.

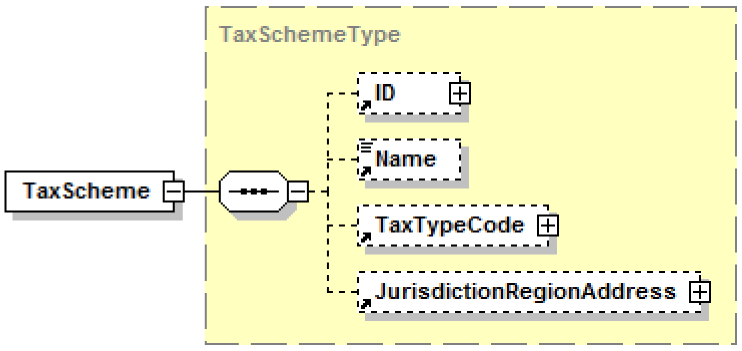
Attributes

No attributes.

4.208 Element: TaxScheme

The `TaxScheme` element identifies and describes a particular tax scheme, i.e. a type of tax as well as the area of jurisdiction in which the tax applies.

The `ID` element holds an identifier of the tax scheme. It is defined as a `CodeType`. For a tax scheme for sales tax, the `ID` code could for instance be "`UKVAT`" `(VAT in the United Kingdom)`. Other tax scheme identifiers could be "`GST`" (Australia) or "`CaliforniaStateTax`".



The `TaxTypeCode` can be used to identify the type of tax. As the "`UKVAT`" in the example above is the Value Added sales Tax, the `TaxTypeCode` in that case could be "`SalesTax`".

The `JurisdictionRegionAddress` associates the tax scheme with information that makes it possible to identify and locate the geographic area in which the tax scheme applies.

A general *Overview of Taxation Structures* is provided in a separate section above in this document.

Attributes

No attributes.

4.209 Element: TaxSubTotal

The `TaxSubTotalType` records amounts and information relating to the tax sub-total for one tax scheme (i.e. a type of tax such as VAT (Value Added Tax)) and one tax category within that tax scheme.

The optional `TaxableAmount` element records the amount to which the tax rate is applied in order to calculate the tax amount due. The tax rate is expressed in the `TaxCategory/Percent` element.

The mandatory `TaxAmount` element holds the amount of tax due, calculated from the taxable amount and the tax rate. It is explicitly stated and not derived, and will therefore convey the results of any rounding that may have occurred.

The mandatory `TaxCategory` element associates the `TaxSubTotal` with a tax category within the applicable tax scheme.

A general *Overview of Taxation Structures* is provided in a separate section above in this document.

Attributes

No attributes.

4.210 Element: TaxTotal

The `TaxTotal` element can be used to describe a summary of tax information..

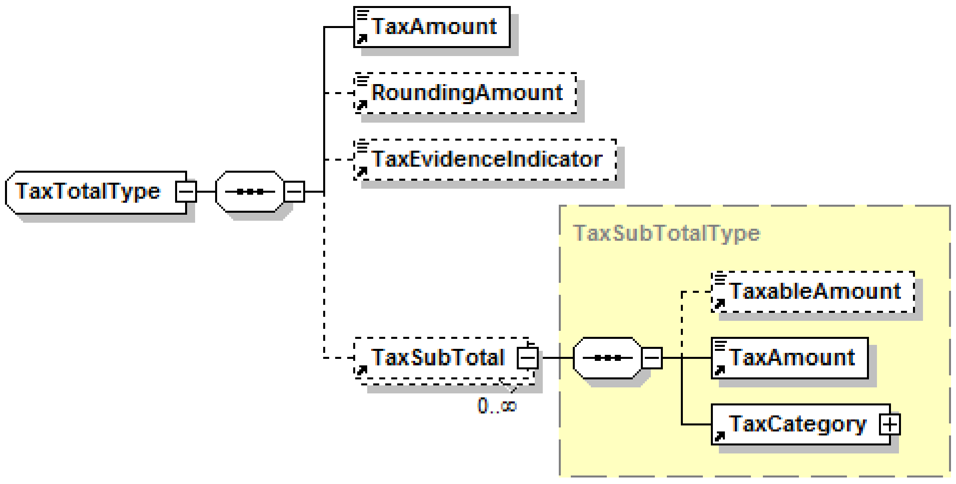
See `TaxTotalType` for further details.

Attributes

No attributes.

4.211 Type: TaxTotalType

The `TaxTotalType` type records information relating to the total tax for one particular type of tax, expressed as a `TaxScheme` (e.g. VAT (Value Added Tax)), and for all categories of that tax type (e.g. Standard Rate, Special Assessment, etc.), expressed as one or more `TaxCategory` elements.



Each `TaxTotal` **MUST NOT** include more than one type of `TaxScheme` type.

Note that elements based on `TaxTotalType` are always repeatable in context, in order to convey tax totals that relate to more than one `TaxScheme`.

The `TaxAmount` records the total amount of tax due for the tax scheme, calculated from the sum of each of the tax sub totals (where each subtotal is for a separate tax category within that Tax Scheme). If specification of a rounding amount is required, the optional `RoundingAmount` element should be used.

The repeatable `TaxSubTotal` element records information relating to the tax subtotal for one `TaxCategory` per `TaxSubTotal` instance. Note that all instances of `TaxSubTotal` elements that appear within a `TaxTotalType` type instance **MUST** be associated with the same `TaxScheme` so that there **MUST NOT** be more than one `TaxSubTotal` for a particular `TaxCategory` within a `TaxScheme`.

A general *Overview of Taxation Structures* is provided in a separate section above in this document.

Attributes

No attributes.

4.212 Element: TaxTypeCode

The `TaxTypeCode` is used to identify a type of tax in a tax scheme. It is defined as a `CodeType`. See `TaxScheme` for further information.

Attributes

No attributes.

4.213 Element: TermsAndConditionsDetails

The `TermsAndConditionsDetails` element allows the sender of a business document to convey a digital rendering of a document covering human-readable terms and conditions.

See `DocumentRenderingType` for further information.

Attributes

No attributes.

4.214 Element: TermsReferenceCode

The `TermsReferenceCode` can be used to record an event from which terms are offered for a length of time, identified by a standard code.

See `PaymentTerms` for further information.

Attributes

No attributes.

4.215 Element: Title

The `Title` can be used to record title of e.g. a contact. It is defined as a `LongStringType`.

See `ContactType` for further information.

Attributes

No attributes.

4.216 Element: TotalPrice

The `TotalPrice` element defines the total amount of a price.

See `PriceDeclarationType` for further information.

Attributes

No attributes.

4.217 Element: ToThisPointOnAd

The `ToThisPointOnAd` element identifies the point on the advertisement at which the x-y co-ordinates specifying the location of an ad on the page are taken to end.

The `ToThisPointOnAd` element takes the same set of allowed values as the `FromThisPointOnPage` element. See that element definition for the allowed values.

If `ToThisPointOnAd` is not specified then its value **SHOULD** be assumed to be `TopLeft`.

Attributes

No attributes.

4.218 Attribute: transmissionDateTime

The *`transmissionDateTime`* attribute records a time stamp for a specific transmission of an AdsML message. It is defined as a `DateTimeType`.

4.219 Element: TransmissionDescription

The `TransmissionDescription` is used in administrative responses to describe a transmission that the administrative response is responding to. The description contains the id of the original transmission and the time point it was received.

Attributes

transmissionIDRef (required)

A reference to a previous transmission's `transmissionID`.

transmissionReceivedDateTime (optional)

The time point when the referenced transmission was received.

4.220 Element: Type

The `Type` element is used in several contexts. It is defined as a `CodeType` and can thus use a controlled vocabulary for validation.

See its parent element for descriptions of its role in a particular context.

Attributes

No attributes.

4.221 Type: URIAddressType

The `URIAddressType` extends the `CommunicationChannel.Base` type to define a Uniform ResourceIdentifier (URI) address.

The `URIAddressType` content model is a required `URI` element followed by an optional `Label` element.

The `URI` element records the URI address in a form conformant to RFC 3986. The `URI` element is declared as `URIType`.

The `Label` element records a label recording descriptive text about the URI address as a string. The `Label` element is declared as `ShortStringType`.

Attributes

No attributes.

4.222 Element: Unit

The `Unit` element is defined as `CodeType` and should be used to records units. See for instance the `PriceComponent` element.

Attributes

No attributes.

4.223 Element: UsageLabel

The element `UsageLabel` can be used to capture a code describing usage in any context. It is defined as a `CodeType`.

Attributes

No attributes.

4.224 Element: Usage

The `Usage` element is defined as a CodeType.

Attributes

No attributes.

4.225 Element: ValidityPeriod

The `ValidityPeriod` element specifies the period of time for which the referenced business object is considered to be valid and so has contractual or legal significance. For example, the period of time for which a contract is considered valid and so is legally binding. The element is defined as a `PeriodType`. See `PeriodType` for more information.

Attributes

No attributes.

4.226 Attribute: version

The *`version`* attribute records a version identifier rating for the element that it qualifies. For example, where it is required to identify the specific version of a content file format or a software application, then the version attribute would specify the requisite version information. For instance, '`1.2`'.

The *`version`* is recorded as a `ShortStringType`.

4.227 Element: XCoordinate

The `XCoordinate` element identifies the 'x' co-ordinate position of the ad on the page, recording it as a Unit of Measure and a value.

Attributes

No attributes.

4.228 Element: YCoordinate

The `YCoordinate` element identifies the 'y' co-ordinate position of the ad on the page, recording it as a Unit of Measure and a value.

Attributes

No attributes.

5 Data types

Common data type definitions are data types derived by AdsML for common use in the AdsML standards.

Data types can be simple, a data value only, or complex, where the data value is qualified by attributes recording ancillary metadata essential to the meaning of the data value.

5.1 Simple data types

Name	Base type	Description
`AmountType`	`xs:decimal`	Restricted to be a decimal number with a maximum of 2 fraction digits.
`BooleanType`	`xs:boolean`	No restrictions imposed.
`DateType`	`xs:date`	No restrictions imposed.
`DateTimeType`	`xs:dateTime`	No restrictions imposed.
`DateTimeDateType`	`Union of xs:dateTime and xs:date`	No restrictions imposed.
`DecimalType`	`xs:decimal`	No restrictions imposed.
`DoubleType`	`xs:double`	No restrictions imposed.
`IDType`	`xs:id`	No restrictions imposed.
`ImportanceType`	`xs:positiveInteger`	Restricted to minimum inclusive value of 1 and maximum inclusive value of 5. Records a scale from 1 (low) to 5 (high).
`IntegerType`	`xs:int`	No restrictions imposed.
`LanguageType`	`xs:language`	No restrictions imposed.
`LimitedDecimalType`	`xs:decimal`	Restricted to maximum 10 fraction digits.
`LongNormalizedStringType`	`NormalizedStringType`	Restricted to a maximum length of 255 characters.
`LongTokenType`	`xs:token`	Restricted to a maximum length of 255 characters.
`LongStringType`	`xs:string`	Restricted to a maximum length of 255 characters.
`NormalizedStringType`	`xs:normalizedstring`	No restrictions imposed.
`PositiveIntegerType`	`xs:positiveInteger`	No restrictions imposed.
`PriorityType`	`xs:integer`	Restricted to minimum inclusive value of 1 and maximum inclusive value of 9. Records a scale from 1-9 where '`1`' signifies the highest rating, '`8`' signifies the lowest rating and '`9`' signifies 'user defined'

Name	Base type	Description
`QIDType`	`LongTokenType`	A data type used for recording identifiers following the AdsML approach to create globally unique identifiers from local values. The `QIDType` is derived from `LongTokenType` and so is restricted to a maximum length of 255 characters. The structure of a `QIDType` value **MUST** be according to the following: `[domainname][/subdomain]:[date]:[local id]`. The Backus Naur Form (BNF) expression for this is: `<GUID> ::= <domainname> {"/"<subdomain>} ":" <date> ":" <local_id>` `<domainname>` (required) is the internet domain name owned by the authority issuing the identifier. `<subdomain>` (optional, repeatable) is an internet sub domain within the domain name. `<date>` (required) is an ISO 8601 date with XML Schema restrictions. The date **MUST** record a date when the domain name used was in possession of the issuing authority. `<local_id>` (required) is a local identifier within the domain of the issuing authority. The local identifier **MUST** be unique within the domain and date provided.
`SchemaVersionType`	`xs:string`	Restricted to a pattern of [1-9][0-9]?\.[0-9]+\.[0-9]+.
`ShareType`	`xs:decimal`	No restrictions imposed.
`ShortTokenType`	`xs:token`	Restricted to a maximum length of 50 characters.
`ShortStringType`	`xs:string`	Restricted to a maximum length of 50 characters.
`StringType`	`xs:string`	No restrictions imposed.
`URIType`	`xs:anyURI`	No restrictions imposed.

Name	Base type	Description
`VersionedQIDType`	`LongTokenType`	A datatype defined as a `QIDType` with an optional version extension. The structure of a `VersionedQIDType` **MUST** conform to the rules of the `QIDType` and thus be valid according to its definition of the three first sections. The optional fourth section is a version identifier for the `QIDType` value. It may be in any format. The colon (":") character is reserved for use as a separator between the four sections of the VersionedQIDType. The colon character **MUST NOT** appear in any of the data sections themselves.

5.2 Simple types with internationalization extensions

The following complex types are extensions of corresponding simple types with internationalization attributes from the *`i18nAttributes`* attribute group including *`xml:lang`* , *`dir`* and *`source`* attributes. The types that have been extended are those often used for recording human-readable string values.

The name of the type with the extension attributes have been derived from the base simple type with an added '`.i18n`' suffix:

- `LongStringType.i18n`
- `ShortStringType.i18n`
- `StringType.i18n`.

5.3 Simple root data types

In some element contexts AdsML provides an extension facility that allows specific values to be used if desired. This is achieved by specifying a default type known as a 'root type' for the element context. Using XML Schema type derivation, the root type can be derived from and the derived type substituted in an instance document, thereby allowing controlled vocabularies to be created and used in AdsML. Using this mechanism it is possible to create controlled vocabularies by deriving from a 'root type'.

See the '*E-Commerce Usage Rules & Guidelines*' document for an explanation of the rules for implementing and using controlled vocabularies in AdsML messages.

Root types are defined in the AdsML Type Library schema where they are intended for public reuse across all AdsML specifications and schema. Root types particular to a specific specification and schema are defined by that specification and schema.

Simple root types are defined for use in the following contexts:

Root type	**Usage context**
`BusinessMessageRootType`	*`messageCode`* attribute context of an AdsML message; `ItemType` element context. Records a code identifying the type of the message or Item as a `ShortTokenType`.
`CodeRootType`	`RequirementSpecType` type context and multiple other contexts where a value must be recorded in codified form. Records the code as a `LongCodeRootType` restricted to 50 characters.
`ContactRoleRootType`	`Role` element context. `ContactRoleRootType` records the role of the subject identified as a contact as a `LongStringType`.
`CurrencyCodeRootType`	`CurrencyCode` element context. Records a currency code as a `ShortTokenType`.
`EncodingRootType`	`Encoding` element context. Records the type of encoding used as a `ShortTokenType`.
`EncryptionMethodRootType`	`EncryptionMethod` element context. Records the type of encryption method used as a `ShortTokenType`.
`IDLabelRootType`	`IDLabel` element context. Records the type of an identifier as a `ShortStringType`.
`LongCodeRootType`	Used in `CodeValue`. Defined as a `LongTokenType`.
`PartyRoleRootType`	`Role` element context. Records the role of a party as a `ShortStringType`.
`PreflightStatusRootType`	`PreflightStatus` element context. Records the status of a preflight check as a `ShortTokenType`.
`PropertyRootType`	`Property` element context. Records a property as a `ShortTokenType`.
`RoleRootType`	`Role` element context. Records a role as a `ShortStringType`.
`StringRootType`	`Description` element context. Records a description as an unrestricted string of type xs:string.

5.4 Enumerated simple data types - Normative Controlled Vocabularies

Normative controlled vocabularies are used to allow the specification and control of the values that are used in particular element or attribute contexts. AdsML defines and creates normative controlled vocabularies using type derivation. In some element or attribute contexts only an AdsML controlled vocabulary is allowed and in such places an AdsML controlled vocabulary will be directly specified as the type of the element or attribute in question.

5.4.1 AdminMessageClassCV

Defines a list of administrative message class types as a restriction on the MessageClassCV.

AdminMessageClassCV	
Code	**Definition**
MessageReceived Acknowledgment	An administrative message acknowledging successful receipt of a business transaction.
TechnicalError	An administrative message reporting a technical error with a received business transaction message.

5.4.2 AdsMLBusinessMessageCV

Defines a list of the types of business messages that are possible to use in execution of AdsML business processes. For each code identifying a message and message group given in the '*Advertising Component Interactions Analysis*' document an enumeration is defined.

`AdsMLBusinessMessageType` is derived from `BusinessMessageRootType`.

AdsMLBusinessMessageCV	
Code	**Definition**
	Please see the XML Schema for the list of values and definitions.

5.4.3 MessageClassCV

Defines a list of AdsML message classifications for an AdsML message. `MessageClassCV` defines three message classifications that can be used to distinguish a message as a business message or an administrative message, either an acknowledgement or an error message.

MessageClassCV	
Code	**Definition**
BusinessTransacti on	A standard business transaction. The `Item` contains an AdsML business message of the type identified by the `MessageClass` element's sibling `ItemType` element.
MessageReceived Acknowledgement	An administrative message acknowledgeing successful receipt of a business transaction. The type of the business transaction message is identified by the `MessageClass` element's sibling `ItemType` element.
TechnicalError	An administrative message reporting a technical error with a received business transaction message. The type of the business transaction message is identified by the `MessageClass` element's sibling `ItemType` element.

5.4.1 OperatorCodeCV

Defines a set of math operator types to be used in various contexts.

OperatorCodeCV	
Code	**Definition**
Divide	The divide operator
Multiply	The multiply operator

5.4.2 PhoneTypeCV

Defines a list of phone types intended for use in the `CommunicationChannel.Phone` element context. `PhoneTypeCV` is derived from `ShortTokenType` and is used by the `MessageClass` element child of the `ItemHeader` element context.

PhoneTypeCV	
Code	**Definition**
Voice	The phone is used for verbal communications.
Fax	The phone is used for fax communications.
Data	The phone is used for data communications.

5.4.3 PointOfOriginTypeCV

Defines a list of point of origin types intended for use in the `AbsolutePosition` element context.

PointOfOriginTypeCV	
Code	**Definition**
TopLeft	Top left corner
TopRight	Top right corner
Center	Center position
BottomLeft	Bottom left corner
BottomRight	Bottom right corner

5.4.1 ResponseConditionsCV

The ResponseConditionsCV list a set of values that is intended to be used to specify special conditions in a message response.

ResponseConditionsCV	
Code	**Definition**
AcceptedAsRequestedByBuyer	The content of the response is an acceptance of the requested options.
AcceptedWithChangesBySeller	The content of the response is an acceptance of the requested options, but with changes or amendments by the

ResponseConditionsCV	
	seller.

5.4.1 TextDirectionsCV

Defines a list of text reading direction codes taken from the W3C's "Internationalization Tag Set (ITS) Version 1.0" (http://www.w3.org/TR/its/)

TextDirectionsCV	
Code	**Definition**
ltr	left-to-right text
rtl	right-to-left text
lro	left-to-right override
rlo	right-to-left override

5.4.2 TransmissionStatusCV

Defines a list of transmission status types used to distinguish production from test messages.

TransmissionStatusCV	
Code	**Definition**
Production	Identifies the transmission as a production transmission.
TransmissionTest	Identifies the transmission as a test transmission.
BusinessMessageTest	Identifies the transmission as a test business message transmission.

5.5 Complex data types

Name	**Base type**	**Description**
`VersionedStringType`	`ShortStringType`	Records text as a `ShortStringType`. Extends `ShortStringType` to add a required *`version`* attribute. The *`version`* attribute is used to record version information as a string of `ShortStringType`.

5.6 Complex root data types

Root type	**Usage context**

Root type	Usage context
`FormatRootType`	`Format` element context. Records the type of format used as a code, extending the `CodeRootType` to add an optional *`version`* attribute. The *`version`* attribute is used to record the version of the format as a `ShortStringType`.