Aviation Fuel Data Standards Group

XML Standard for Electronic Fuel Invoices

Version 3.2.1

17 December 2018

Copyrights

Table of Contents

DOCUMENT REVISION HISTORY 4
OVERVIEW 6
PROPOSED FUEL INVOICE STANDARD 6
REVIEW OF SAMPLE FUEL INVOICES 7
BASIC MODELLING CONCEPTS 8
TABLE 1 - FUEL INVOICE STANDARD: CHANGES COMPARED TO VERSION 3.0.1 ... 13
CURRENCY CONVERSION: EXAMPLES 14

DOCUMENT REVISION HISTORY

Revision Record

Version	Date	Modified Sections / Description
1.0 Draft	4/27/04	RamiroOrtiz,Vaishnavi Madineni (BE)/Initial Draft
1.1 Draft	7/8/04	Vaishnavi Madineni (BE)/Added additional fields in the Invoice Header section-CustomerIEntityId, InvoiceTaxes, SubTotals. Deleted the field InvoiceDeliveryLocation from Invoice Header
1.2 Draft	9/12/04	Laurie Kern (BE)/ Added additional fields in the Invoice Header, Removed Attachements and Subtotals, Addded ItemDescripton and ItemInvoiceAmount, Clarified Invoice Summary Fields, and Updated Schema discrepancies.
1.3 Draft	9/22/2004	Laurie Kern (BE)/ Modified and standardized number formats for rate/percent/amount/quantity. Moved ItemTax to SubItem (tax). Added SubTax to new SubItemTax. Clarify and add additional codes.
1.3.1 Draft	10/14/2004	Laurie Kern (BE) added element SubItemQuantity
1.3.2 Draft	10/21/2004	Laurie Kern (BE) changed: InvoiceDeliveryLocation to Required with a repeat of 1 to many, InvoiceReferenceValue to Optional and the attribute Required, ItemDeliveryReferenceValue changed repeat from unbounded to 2000 and added Previous Invoice Number to the enumerated list of attributes
1.3.3 Draft	11/11/2004	Laurie Kern (BE) Add Prepaid Invoice Correction to InvoiceTranscationType
1.3.4 Draft	1 Dec 2004	Laurie Kern (BE) Final changes for release canidate. Add ExchangeRate information, reordering of elements, additional codes.
1.0.0 Release Candidate	2 Dec 2004	Laurie Kern (BE)
1.0.1 RC	28 Feb 2005	Add SubInvoice above InvoiceLine to allow for different/multiple affiliates. Remove Affiliate from InvoiceIDDetails attribute InvoiceIDType. Add Version element to InvoiceTransmissionHeader.
1.0.2 RC	08 Mar 2005	Corrected InvoicePaymentTermsDateBasis and InvoicePaymentTermsType. Changed Length of 20 to MaxLen of 20.
1.1	28 Ap 2005	Included subinvoice header .
1.2	5 Jul 2005	Bryan Terry (BE) Minor changes following Group meeting in Singapore
1.3.0	16 Dec 2006	Mark Johnson (CGFL) amendments relating to the use of the code directory by this schema
2.0.0	16 Dec 2006	Mark Johnson (CGFL) amendment s relating to the structure of the schema following Miami discussions. Also now references an external file (namespace) for its enumeration content.
2.0.1	15 Jan 2006	Version numbering system amanded to increase flexibility and remove from filename
2.0.2	24 Feb 2006	SubInvoiceIDState field updated in schema to match documentation
3.0.0	26 Jun 2007	Numerous changes made by the Fuel Data standard group, mostly with regard to local currency and exchange rates, and with regard to summarizing/recapturing taxes at different levels.
3.0.1	09 May 2008	Extraction from Table 2 into an independent PDF and Excel file. Minor correction of the documentation about taxes being included in amount fields. Correction of the documentation of the format for the elements InvoiceCreationDate, InvoiceIssueDate, SubInvoiceIssueDate to align the documentation with the schema.
3.1.0	09 May 2008	Optional elements UOMFactor added in the invoice SubItem, SubItemTax and SubTaxInvoice (SubItemInvoiceUOMFactor, SubItemTaxInvoiceUOMFactor, SubTaxInvoiceUOMFactor).
3.1.0	13 Jun 2008	Optional elements added in the transmission (TransmissionID, IssuingOrganizationID, ReceivingOrganizationID, TestFlag, TransmissionData @Name), in the invoice header (InvoiceDeliveryCountryCode). Version name changed to IATA:FuelInvoiceV3.1.0
3.1.0	18 Aug 2008	Correction of an error (unnecessary field TransmissionID)

Version	Date	Modified Sections / Description
3.1.0	24 Sep 2008	Correction of an error (unnecessary field TransmissionData)
3.1.0	01 Oct 2008	Additional documentation of the use of UOM factors
3.2.1	17 December 2018	Addition of DensityInformation (3.2.0) and import names adjusted to conform to IATA standard file naming. Version attribute also added to header to conform with IATA standards.

Overview

PURPOSE:

The IATA Aviation Fuel Data Standards Group (the "Group") is proposing a standardized electronic invoice for fuel purchases that can be used by both fuel suppliers and airlines in the airline industry. The purpose of the proposed standard is to facilitate a cost effective exchange of data between parties to consummate a buy/sell transaction.

Background:

The Group has created a proposal for a standardized format for electronic fuel invoices. An initial draft of the proposed standardized file layout was reviewed and modified based on comments from the Group. Additional comments received after the conclusion of meetings have been incorporated into this document. Other comments and suggestions may be incorporated with future revisions to the standard. The basis for the proposed standardized fuel invoices file layout are presented below:

1) The use of two EDI standards are used by the industry, AVNET UN/EDIFACT – INVOIC and ASC X12 – 810, to transmit invoices from fuel suppliers to the airlines.

2) The structure and elements were identified as they related to either the AVNET UN/EDIFACT – INVOIC and ASC X12 – 810 standards. There were some element references that were omitted and in other cases elements were added that were not referenced in the applicable AVNET EDI standard.

3) The EDI standards usage is not consistent between parties.

4) Fuel suppliers modify EDI transactions for specific customers within the context of the existing standards.

5) Fuel suppliers and airlines only use a small number of the EDI data elements, which was originally designed as generic invoice format for all industries and uses.

6) There is a desire in the airline industry to streamline the data exchange and process to facilitate the financial settlement of the fuel invoice.

Proposed Fuel Invoice Standard

The proposed fuel invoice standard is based on the existing EDI AVNET INVOIC and 810 standards. The AVNET requirements were reviewed along with anecdotal usage of the EDI standards and comments from the Group to develop the proposed XML standard. The proposal contains the invoice data elements that are necessary to provide information to the customer for review, attestation and approval of a fuel purchase.

The proposed fuel invoice standard is presented in Table 1 and the XML schema is available in Appendix 1. As a pictorial representation of the XML schema as well as the source code. The XML schema may vary in structure as represented in Table 1 due to the use of XML standards in constructing the schema. Table 1 is constructed as follows:

1. Structure – the major sections of a fuel invoice. Indentations represent subsets of the structure. Depending on the nature of the subset, the subset may be repeated to provide additional information. In addition, a subset may be options. If a subset is optional, any required element is only required when the subset is used.

2. Element – the element represents the invoice data element containing a value or information required by the supplier or customer to process and settle a fuel invoice. Some elements may be required while others are optional. The values of an element may follow the existing industry standards or specific requirements by the parties involved.
3. Description – the description provides additional information or clarification of the Element.
4. Repeat – the number of times this element is to appear in the structure.
5. Element Attributes – describes the attributes of each element: R=required, O=optional, Type: N=numeric, AN= alpha-numeric, DT=date time, Length=length of element
6. Value Source – identifies the source of the value or brief description of the value.
7. Notes – additional information on the use of the invoice element.

Review of Sample Fuel Invoices

Several fuel suppliers provided samples of their invoice formats representing different types of invoices. The samples were compared to the proposed invoice layout to identify if any structural issues were evident with the proposed standard. In general, the sample invoices could be mapped to the standard layout. There are a few items noted that will require further discussion and direction from the Fuel Data Standards Group. The noted items are presented below.

1. Invoice Header
   a. Customer Account Number – Several invoices contained an account number that identified the supplier's customer. The proposed layout does contain an explicit field for an account number.
   b. Reference Numbers – Several types of reference numbers appeared in the sample invoices header section. These include references such as Original Document Number, Contract Number, Order Number, Flight Number and Registration Number. The use of the Invoice Header custom fields can accommodate these references.
   c. Additional Date References – Additional dates for Contract, Delivery, Trade and Order were identified. The Invoice Header proposed format does not accommodate these additional date references.
   d. Freight Terms – One-invoice type included freight terms in the invoice header. The Invoice Header proposed format does not accommodate these terms.
   e. Quantities – One invoice type contained total delivered quantities at the invoice header. The amounts available for presentation were restricted to monetary amounts in the proposed layout.
2. Line Detail
   a. Invoice Number – Invoice line detail is broken into three sections
      i. Sub-total – a logical grouping of lines (i.e. by ticket or by destination)
      ii. Line Number – standard invoice line
      iii. Sub-item – components of a line
   b. Delivery Month – In addition to delivery date, one invoice type used Delivery Month at the line level.

Basic Modelling Concepts

When two parties (such as airports, fuel companies or similar) communicate in a business sense there is a *transmission* happening between them.

The result of each transmission is an exchange of a business document. The document is created in XML format and then populated with specific business data. In other words, one party will generate and then send a business document to another party.

An instance of a document in XML format populated with business data is called business document. If we talk about fuel invoice function – then there is an fuel invoice document.

As a very first line in each XML document should be the following line - <?xml version="1.0" encoding="UTF-8" ?>.

Transmission

The following diagram represents the high-level view of a transmission, where transmission represents an instance of a business document for a specific business function.

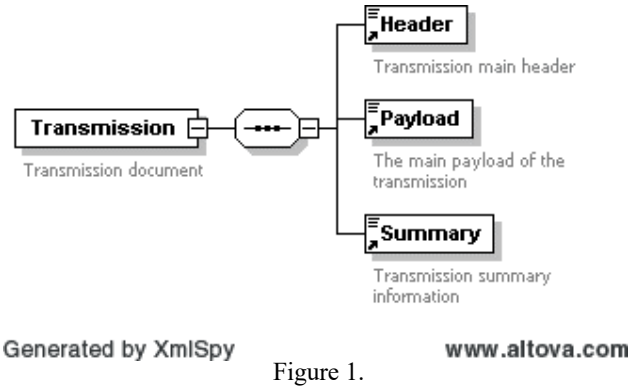

Figure 1.

The header element will contain main high level data related to the particular transmission.

The payload element is the one to contain business related data. It will contain business data sets that are the main purpose of the transmission. The data can be grouped and classified so several groups of data can be stored and send in one business document.

The payload can contain one or more business components – for example one or more airport invoices within one transmission. All components will be placed in a sequential order and each component will contain only data relating to the specific business item - such as invoice for example.

The following figure presents XML model with specific cardinality included.

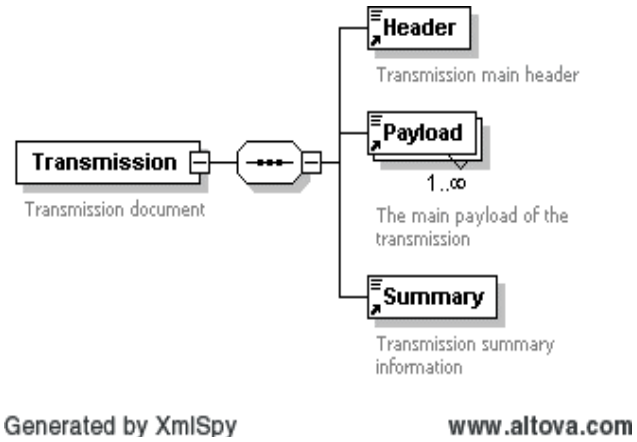

Figure 2.

The summary element will contain summary data relating to the entire transmission. It will enable summary checks to be performed in order to check validity of the overall transmission document.

Payload

The following figure presents the anticipated view of the Payload element.

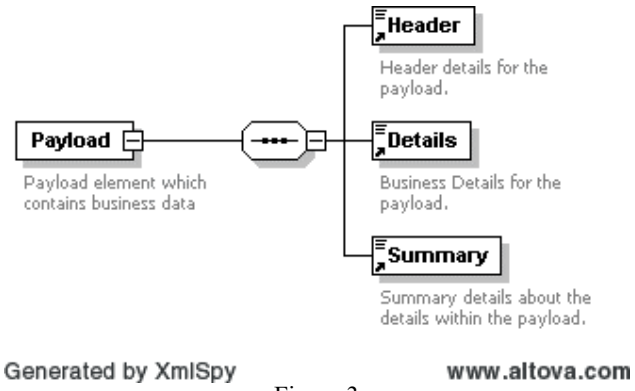

Figure 3.

Each Payload element consists of three main elements:

- Header – this element is used to store all header level details that related to the business content. For example all header information within an invoice (such as address details, seller and buyer organization, etc.) would be contained here.
- Details - this element is used to store all business specific data such as line items information within an invoice.
- Summary – this element is used to summarize all info sent within the payload. That information could be used for validation checks.

Fuel Invoice Basic XML Structure

Fuel Invoice Model - High-Level

The high level view of the Fuel Invoice document is presented on the figure 4. The payload element represented at figure 2 is presented here as Invoice element and relates to the business purpose which is fuel invoicing.

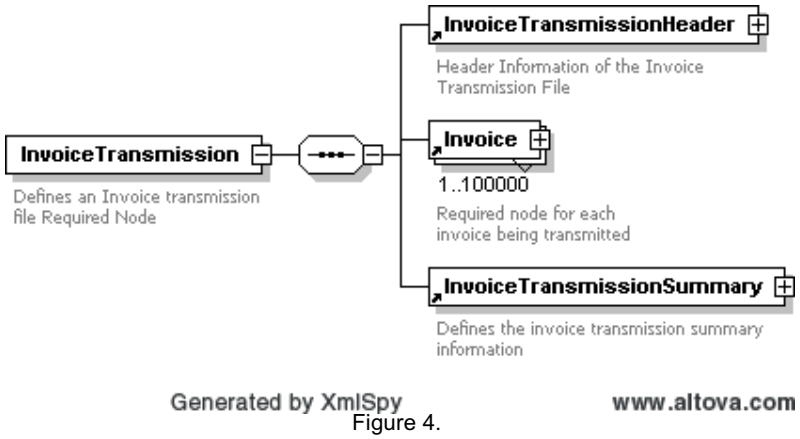

Figure 4.

The Fuel Invoice document is an instance of the above XML model called InvoiceTransmission. From the high-level each airport invoice contains three main elements as follows:

- InvoiceTransmissionHeader
- Invoice
- InvoiceTransmissionSummary

The InvoiceTransmissionHeader element contains header level data that will describe the content of the transmitted document. There is only one instance of this element for each Fuel Invoice document. It will indicate what will be transmitted within the next element – Invoice. The high-level view is presented ion the following figure.

Figure 5.

The Invoice element is the main payload component (in message sense) for each Fuel Invoice document. From the above figure we can notice that each InvoiceTransmission can contain one or more invoices (Invoice elements) within. Invoice element contains invoice related data only – each invoice element will contain data related to one hard copy invoice only. The following figure presents the high level view of the Invoice element.

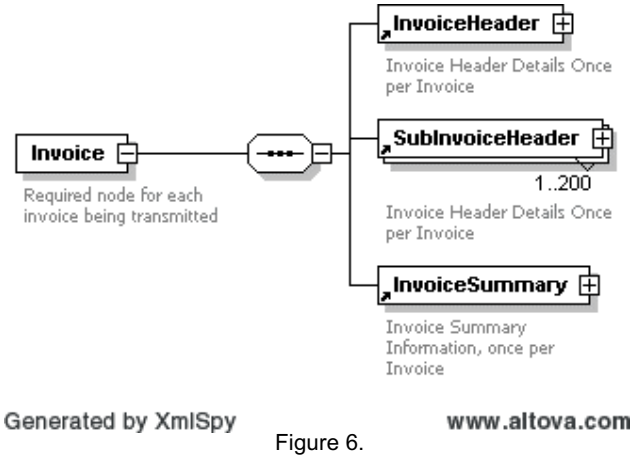

Figure 6.

The InvoiceTransmissionSummary element contains summary level data for all invoices included within one Fuel Invoice document. This element would be used to validate the entire business document – which in turn could contain one or more invoices. The high-level view is presented on thefigure bellow.

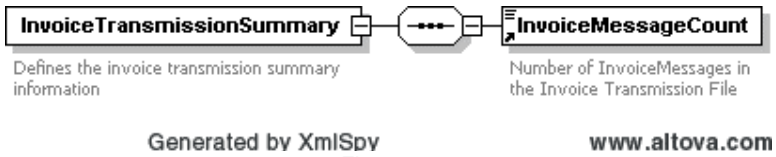
InvoiceTransmissionSummary
InvoiceMessageCount
Defines the invoice transmission summary information
Number of InvoiceMessages in the Invoice Transmission File
Generated by XmlSpy
www.altova.com

Figure 7.

Table 1 - Fuel Invoice Standard: 3.1.0 changes compared to version 3.0.1

ROW	ELEMENT NAME	NOTES
6	Version: changed from "3.0.1" to "IATA:FuelInvoiceV3.1.0"	Length of the Version element modified from AN10 to AN50.
7	IssuingOrganizationID	New optional element
8	ReceivingOrganizationID	New optional element
9	TestFlag	New optional element
22	InvoiceDeliveryCountryCode	New optional element
153	SubItemInvoiceUOMFactor	New optional element
182	SubItemTaxInvoiceUOMFactor	New optional element
211	SubTaxInvoiceUOMFactor	New optional element

Table 2 - Fuel Invoice Standard: 3.2.0 changes compared to version 3.1.0

ROW	ELEMENT NAME	NOTES
1	OpenTag – encoding did not match specification so changed to UTF-8 from UTF-16	
6	Version: changed from "3.1.0" to "IATA:FuelInvoiceV3.2.0"	
115	DensityInformation	New optional element

Table 3 - Fuel Invoice Standard: 3.2.1 changes compared to version 3.2.0

ROW	ELEMENT NAME	NOTES
1	OpenTag – imports changed to IATA file naming standard. Version attribute added to conform to IATA standards	
6	Version: changed from "3.2.0" to "IATA:FuelInvoiceV3.2.1"	

Currency Conversion: Examples

Multi-currency support is one of the main innovations in version 3.0.0 of the Fuel Invoice Standard. New optional currency conversion elements are provided at various levels:

- InvoiceCurrencyConversion
- SubInvoiceCurrencyConversion
- SubItemCurrencyConversion
- SubItemTaxCurrencyConversion
- SubTaxCurrencyConversion

The structure of a CurrencyConversion element is always :

- ConversionMechanism (Multiply or Divide)
- CurrencyFrom with attribute FactorFrom
- CurrencyTo with attribute FactorTo
- ExchangeRate

ConversionMechanism

The most common case is where the ConversionMechanish is Multiply – meaning that the amount in CurrencyFrom needs to be multiplied by the ExchangeRate to calculate the amount in CurrencyTo.

Example:

Conversion from EUR to GBP where 1 EUR : 1GBP = 0.67779
would be modeled as follows:

```
<CurrencyConversion>
        <ConversionMechanism>M</ConversionMechanism>
        <CurrencyFrom FactorFrom=”1”>EUR</CurrencyFrom>
        <CurrencyTo FactorTo=”1”>GBP</CurrencyTo>
        <ExchangeRate>0.677799</ExchangeRate>
</CurrencyConversion>
```

An ExchangeRate may be used inversely by setting the ConversionMechanism to Divide. In this case the amount in CurrencyFrom needs to be divided by the ExchangeRate to calculate the amount in CurrencyTo.

Example:

Conversion from EUR to GBP where 1 GBP : 1 EUR = 1.47536
would be modeled as follows:

```
<CurrencyConversion>
        <ConversionMechanism>D</ConversionMechanism>
        <CurrencyFrom FactorFrom=”1”>EUR</CurrencyFrom>
        <CurrencyTo FactorTo=”1”>GBP</CurrencyTo>
        <ExchangeRate>1.47536</ExchangeRate>
</CurrencyConversion>
```

FactorTo and FactorFrom

In some cases, where the value of one currency is very low compared to the other currency, a multiplication factor may be added.

Example:

Conversion from CLP to USD where 100 CLP : 1 USD = 0.190585
would be modeled as follows:

```
<CurrencyConversion>
        <ConversionMechanism>M</ConversionMechanism>
        <CurrencyFrom FactorFrom=”100”>CLP</CurrencyFrom>
        <CurrencyTo FactorTo=”1”>USD</CurrencyTo>
        <ExchangeRate>0.190585</ExchangeRate>
</CurrencyConversion>
```

Although less common, a multiplication factor other than 1 could also be provided for the CurrencyTo.

Example:

Conversion from USD to CLP where 1 USD : 100 CLP = 5.24700
would be modeled as follows:

```
<CurrencyConversion>
        <ConversionMechanism>M</ConversionMechanism>
        <CurrencyFrom FactorFrom=”1”>USD</CurrencyFrom>
        <CurrencyTo FactorTo=”100”>CLP</CurrencyTo>
        <ExchangeRate>5.24700</ExchangeRate>
</CurrencyConversion>
```

Currency Conversion formulas

To convert an amount in CurrencyFrom to an amount in CurrencyTo, the following rules apply:

If the ConversionMechanism is multiply then use the following formula:

amount in CurrencyTo = (amount in CurrencyFrom / FactorFrom) x (ExchangeRate x FactorTo)

(2) If the ConversionMechanism is divide then use the following formula:

amount in CurrencyTo = (amount in CurrencyFrom / FactorFrom) / (ExchangeRate / FactorTo)

Examples of how to use the currency conversion formulas:

Assume that 1 CLP : 1 USD = 0.00190585 and consequently 1 USD : 1 CLP = 524.70

Conversion from CLP to USD where 100 CLP : 1 USD = 0.190585
could be modeled using the Multiply conversion mechanism as follows:

```
<CurrencyConversion>
        <ConversionMechanism>M</ConversionMechanism>
        <CurrencyFrom FactorFrom=”100”>CLP</CurrencyFrom>
        <CurrencyTo FactorTo=”1”>USD</CurrencyTo>
        <ExchangeRate>0.190585</ExchangeRate>
</CurrencyConversion>
```

To convert 500 CLP to USD:

(500 CLP / 100) x (0.190585 x 1) = 0.95 USD

Conversion from CLP to USD where 1 USD : 100 CLP = 5.24700
could be modeled using the Divide conversion mechanism as follows:

```
<CurrencyConversion>
        <ConversionMechanism>D</ConversionMechanism>
        <CurrencyFrom FactorFrom=”100”>CLP</CurrencyFrom>
        <CurrencyTo FactorTo=”1”>USD</CurrencyTo>
        <ExchangeRate>5,24700</ExchangeRate>
</CurrencyConversion>
```

To convert 500 CLP to USD:

(500 CLP / 100) / (5.24700 / 1) = 0.95 USD

Use of Pricing and Invoice UOM Factors: Examples

SubItemPricingUOMFactor indicates to how many units of SubItemPricingUOM the pricing unit price (SubItemPricingUnitRate) is applicable.

SubItemInvoiceUOMFactor indicates to how many units of SubItemInvoiceUOM the invoice unit price is applicable (SubItemInvoiceUnitRate).

Both UOMFactors should be understood as a tool to enable a presentation of the SubItemInvoiceUnitRate and SubItemPricingUnitRate in the restricted format (numeric 18,6).

Examples:

1- Charge at 3.5 EUR per HectoLitre, and the delivery is in Litres:

Solution 1:

```
<SubItemPricingUnitRateType>UR</SubItemPricingUnitRateType>
```

<SubItemPricingUnitRate>3.50000</SubItemPricingUnitRate>

<SubItemPricingUOM>HL</SubItemPricingUOM>

<SubItemPricingUOMFactor>1.000000</SubItemPricingUOMFactor>

<SubItemPricingCurrencyCode>EUR</SubItemPricingCurrencyCode>

Solution 2:

<SubItemPricingUnitRateType>UR</SubItemPricingUnitRateType>

<SubItemPricingUnitRate>3.50000</SubItemPricingUnitRate>

<SubItemPricingUOM>LT</SubItemPricingUOM>

<SubItemPricingUOMFactor>100.000000</SubItemPricingUOMFactor>

<SubItemPricingCurrencyCode>EUR</SubItemPricingCurrencyCode>

Both solutions are acceptable in the standard.

2- Representation of flat fee:

SubItemInvoiceQty should be populated with the number of times that the flat fee is applied (typically: 1).

```
<SubItemProduct>
        <SubItemProductID SubItemProductIDQualifier="FEE">HUF</SubItemProductID>
        <SubItemDescription>INTO-PLANE HOOKUP FEE</SubItemDescription>
        <SubItemPricingUnitRateType>FF</SubItemPricingUnitRateType>
        <SubItemPricingUnitRate>17.000000</SubItemPricingUnitRate>
        <SubItemPricingUOM>EA</SubItemPricingUOM>
        <SubItemPricingUOMFactor>1.000000</SubItemPricingUOMFactor>
        <SubItemPricingCurrencyCode>EUR</SubItemPricingCurrencyCode>
        <SubItemPricingAmount>17.00</SubItemPricingAmount>
        <SubItemInvoiceUOM>EA</SubItemInvoiceUOM>
        <SubItemQuantity>
                <SubItemInvoiceQuantity>1.00</SubItemInvoiceQuantity>
                <SubItemQuantityType>IN</SubItemQuantityType>
                <SubItemQuantityFlag>GR</SubItemQuantityFlag>
        </SubItemQuantity>
        <SubItemInvoiceUnitRate>17.00</SubItemInvoiceUnitRate>
        <SubItemInvoiceCurrency>EUR</SubItemInvoiceCurrency>
        <SubItemInvoiceAmount>17.00</SubItemInvoiceAmount>
</SubItemProduct>
```