



IT- og Telestyrelsen

Ministeriet for Videnskab
Teknologi og Udvikling

OIOUBL Guideline

UBL 2.0 Catalogue prices and quantity

OIOUBL Pris og mængde i kataloger

G40

Version 1.1



This release is protected by Creative Commons License, Naming 2.5 

Colophon

Contact:

Danish National IT and Telecom Agency

E-mail: oioubl@itst.dk

OIOUBL Version 2.01

April 2007

Ministry of Science, Technology and Innovation

National IT and Telecom Agency

Data Standardization Office

Holsteinsgade 63

DK-2100 Copenhagen Ø

Phone +45 3545 0000

Fax +45 3545 0010

<http://www.itst.dk>

itst@itst.dk

Copyrights for this release in accordance with Creative Common, Naming 2.5: 

Permission is granted to:

- *produce processed works based on this document*
- *reproduce and make the document available to the public*
- *use the document for commercial purposes*
provided that the Danish National IT & Telecom Agency be clearly referenced as the source of this release.

Further information about these rights is available at

<http://creativecommons.org/licenses/by/2.5/deed.da>.

Contents

1. Preface.....	4
1.1 Purpose of this document.....	4
1.2 General Points.....	5
2. Relevant UBL Classes and Elements.....	6
2.1 DK names and cardinality.....	6
2.1.1 CatalogueLine.....	7
2.1.2 RequiredItemLocationQuantity.....	8
2.1.3 Item.....	9
3. Description.....	10
3.1 Relationship between the Price and Quantity elements.....	10
3.2 Base Unit.....	10
3.3 Delivery Unit.....	11
3.4 OrderableUnit.....	11
3.5 PackSizeNumeric.....	13
4. Examples.....	14
4.1 Converting BaseUnit to OrderableUnit.....	14
4.2 Quantity dependent pricing.....	16
4.3 Period dependent pricing.....	17
4.4 Specifying customer dependent prices.....	18
4.4.1 Specifying the Customer on the Catalogue Line.....	18
4.4.2 Region dependent Pricing.....	20
4.5 Specifying List Prices.....	22
5. Relevant code lists.....	24
6. Terms and abbreviations.....	25

1. Preface

These guidelines form of a series describing the purpose and use of the business documents that comprise the Danish localization of UBL 2.0, known as OIOUBL.

As well as guidelines describing the use of commonly used elements, a separate guideline has been prepared for each business document.

1.1 Purpose of this document

This guideline specifies the use of classes and elements that deal with prices and quantities in Catalogue documents.

Special focus is given to:

- The elements that are required when specifying prices and quantities and how they interrelate
- How these elements may be used to illustrate different price/quantity relationships
- How to specify different prices for different customers

The following OIOUBL documents are relevant to this guideline:

Document	Description
UBL-Catalogue-2	The Catalogue document is always used for creating new catalogues and it may also be used for updating existing catalogues. Note that each catalogue line entry is overwritten when updating with the Catalogue document. While it is possible to create a catalogue line without a price, in this guideline classes and elements dealing with prices are assumed. For a more detailed description, see the OIOUBL Catalogue guideline (Ref. G03).
UBL-CataloguePricingUpdate-2	This document is used for updating prices and quantities in existing catalogues. Note that the entire RequiredItemLocationQuantity class is overwritten when updating with the CataloguePricingUpdate document. For a more detailed description, please see the OIOUBL Guideline Catalogue Pricing Update (Ref. G07).
UBL-CatalogueItemSpecificationUpdate-2	This document is used for updating the specifications of items in existing Catalogues (such as package sizes). Note that the entire Item class is overwritten when updating with the CatalogueItemSpecificationUpdate document. For a more detailed description, please see the OIOUBL Guideline Catalogue Item Specification Update (Ref. G06).

1.2 General Points

Encrypted delivery should be used when a CPR number is used as an ID.

Price and quantity specification are also described in other OIOUBL documents, such as Order and Invoice (See separate guide line about prices Ref. G25). It is therefore important to consider prices and quantities in the scope of the entire purchasing process, in order to be able to transfer the prices and quantities specified in a Catalogue to an Order, which can be matched against a resulting Invoice.

For a more detailed description of other areas of the Catalogues, see the respective guidelines. For example:

OIOUBL Guideline Catalogue Item Descriptions and Categorization (Ref. G38)

OIOUBL Catalogue Identification, Versioning and Validity Periods (Ref. G37)

OIOUBL Guideline Catalogue Parties (Ref. G39)

2. Relevant UBL Classes and Elements

The elements that are relevant for the specification of prices and quantities are defined:

- a) directly under *CatalogueLine*,
- b) together in the *RequiredItemLocationQuantity* class under *CatalogueLine*, or
- c) in the *Item* class.

Elements under *CatalogueLine* can only be created and updated by use of the Catalogue document. The elements within a *CatalogueLine* are:

- *OrderableIndicator*
- *OrderableUnit*
- *ContentUnitQuantity*
- *OrderQuantityIncrementNumeric*
- *MinimumOrderQuantity*
- *MaximumOrderQuantity*

Elements within the *RequiredItemLocationQuantity* class may also be updated with the CataloguePricingUpdate document. These elements are:

- *MinimumQuantity*
- *MaximumQuantity*
- *Price*
- *DeliveryUnit*

Elements under *Item* may also be updated with the *CatalogueItemSpecificationUpdate* document. These are:

- *PackQuantity*
- *PackSizeNumeric*

2.1 DK names and cardinality

The table below lists the elements and their names in Danish, as well as the cardinality.

Directly under *CatalogueLine* are elements identifying the orderable units. The following are relevant for defining prices and quantities:

2.1.1 CatalogueLine

UK-name	DK-name	Use	Remarks
Catalogue/CatalogueLine/Note	Note	0..n	May be a text description of the contents for a catalogue line. For example, "1 box of 12 each" or "1 pcs. of 5 KG each", etc.
Catalogue/CatalogueLine/OrderableIndicator	KanBestillesIndikator	1	The default value is "true", meaning the item is orderable. The "false" value is used if the goods are not orderable.
Catalogue/CatalogueLine/OrderableUnit	BestillingsEnhed	0..1 (or 1)*	*If the value of OrderableIndicator is true (see above), then OrderableUnit is mandatory and must be specified. The orderable unit is the units of measurement that will be supplied to the Customer for orders specifying one unit. The value is defined using a units of measure code. For example, CS, PK, etc.
Catalogue/CatalogueLine/ContentUnitQuantity	BestillingsEnhedsMængde	1	Specifies the quantity that comprises an orderable unit. For example, if the OrderableUnit is "CS" (case), and the case contains 12 bottles, then ContentUnitQuantity should be "12", and the unit "bottles" is specified in the unitCode attribute (as described below).
Catalogue/CatalogueLine/ContentUnitQuantity@unitCode			This value must be a valid Unit of Measure code. Following the previous example, if the ContentUnitQuantity is "12 bottles", "12" is entered in ContentUnitQuantity, and the unitCode is set to "BO", as in... <cbc:ContentUnitQuantity unitCode="BO">12</cbc:ContentUnitQuantity>
Catalogue/CatalogueLine/OrderQuantityIncrementNumeric	BestillingsMængdeSpring	0..1	The OrderQuantityIncrementNumeric specifies the incremental quantities that must be ordered of a given Item. For example, if the OrderableUnit is 1 bottle (BO), but a box of 12 bottles must be purchased, the OrderQuantityIncrementNumeric is specified as "12", meaning It is only possible to buy 12, 24, 36 bottles, etc.
Catalogue/CatalogueLine/MinimumOrderQuantity	MinimumBestillingsMængde	0..1	Specifies the minimum order quantity for an item. For example, if the MinimumOrderQuantity is 12 (and the OrderQuantityIncrementNumeric is 1), the possible order quantities are 12, 13, 14 units, etc. The unit is specified in the unitCode attribute, as described below.
Catalogue/CatalogueLine/MinimumOrderQuantity@unitCode			Specifies the unit for the minimum order quantity. The value must be a valid units of measurement code, for example, "BO" for bottle.
Catalogue/CatalogueLine/MaximumOrderQuantity	MaksimumBestillingsMængde	0..1	Specifies the maximum order quantity for an item.
Catalogue/CatalogueLine/MaximumOrderQuantity@unitCode			Specifies the unit for the maximum order quantity. The value must be a valid units of measurement code, for example, "BO" for bottle.

2.1.2 RequiredItemLocationQuantity

RequiredItemLocationQuantity defines the class that describes the supplier's base units, and their delivery units. The following elements and sub-classes under *RequiredItemLocationQuantity* are used for prices and quantities:

UK-name	DK-name	Use	Remarks
Catalogue/CatalogueLine/RequiredItemLocationQuantity/MinimumQuantity	MinimumsMængde	0..1	Specifies the minimum quantity that the price applies to. This makes it possible to differentiate prices based on the purchased quantities. The unit is specified in the unitCode attribute, as described below.
Catalogue/CatalogueLine/RequiredItemLocationQuantity/MinimumQuantity@unitCode			Specifies the units of measurement for the minimum quantity. The value must be a valid code. For example, "BO" for bottles.
Catalogue/CatalogueLine/RequiredItemLocationQuantity/MaximumQuantity	MaksimumsMængde	0..1	Specifies the maximum quantity that the price applies to. This also makes it possible to differentiate prices based on the purchased quantities. The unit is specified in the unitCode attribute, as described below.
Catalogue/CatalogueLine/RequiredItemLocationQuantity/MaximumQuantity@unitCode			Specifies the units of measurement for the maximum quantity. The value must be a valid code. For example, "BO" for bottles.
Catalogue/CatalogueLine/RequiredItemLocationQuantity/Price	Pris	0..1	The Price class contains information about the pricing of an item.
Catalogue/CatalogueLine/RequiredItemLocationQuantity/Price/PriceAmount	PrisBeløb	1	Specifies the price (exclusive of tax but including any discounts and charges) for a base unit (BaseQuantity) of an item. Amounts should use a full stop/period as the decimal point separator.
Catalogue/CatalogueLine/RequiredItemLocationQuantity/Price/PriceAmount@currencyID			The currency that applies to the price, expressed using a code list. For example "DKK", "EUR", etc.
Catalogue/CatalogueLine/RequiredItemLocationQuantity/Price/BaseQuantity	BeregningsGrundlagMængde	1	The base quantity used by the price. For example, if the price specified in PriceAmount is DKK 65.00 for one bottle of wine, the BaseQuantity should be "1". The unit is specified in the unitCode attribute, as described below.
Catalogue/CatalogueLine/RequiredItemLocationQuantity/Price/BaseQuantity@unitCode			The units of measurement for the BaseQuantity. The value must be a valid code. For example, "BO" for bottle.
Catalogue/CatalogueLine/RequiredItemLocationQuantity/Price/PriceTypeCode	PrisTypeKode	0..1	Code specifying the price type, such as "CA" for catalogue. For more details see section 6 below.
Catalogue/CatalogueLine/RequiredItemLocationQuantity/Price/PriceTypeCode/@listAgencyID			In the attribute listAgencyID the value "6" specifies that UN/ECE is maintaining the codelist
Catalogue/CatalogueLine/RequiredItemLocationQuantity/Price/PriceTypeCode/@listID			Reference to the used codelist: UN/ECE 5387
Catalogue/CatalogueLine/RequiredItemLocationQuantity/Price/OrderableUnitFactorRate	OrdreAntalMængdeRate	1	The calculation factor from the units of measurement in BaseQuantity to the units of measurement in OrderableUnit. This has the default value of "1", meaning the base unit and the orderable unit are the same. For more detail, see the following sections.
Catalogue/CatalogueLine/RequiredItemLocationQuantity/DeliveryUnit	LeveringsEnhed	0..n	The packing sizes for units in which an item is deliverable. If the items are deliverable in different packing sizes, several Delivery Units may be specified.
Catalogue/CatalogueLine/RequiredItemLocationQuantity/	BatchMængde	1	The quantity of the ordered item that makes a

mLocationQuantity/DeliveryUnit/BatchQuantity			batch in relation to delivery. For example, if delivery can be made in cases, the BatchQuantity would be specified as "1" and cases would be specified in the unitCode attribute, as described below.
Catalogue/CatalogueLine/RequiredItemLocationQuantity/DeliveryUnit/BatchQuantity@unitCode			Specifies the units of measurement for the BatchQuantity. The value must be a valid code. For example, "CS" for case.
Catalogue/CatalogueLine/RequiredItemLocationQuantity/DeliveryUnit/ConsumerUnitQuantity	ForbrugsEnhedsMængde	0..1	Specifies the consumable units of measurement that comprise the BatchQuantity. A consumer quantity should be specified with respect to the BaseQuantity (the units of measurements that the supplier can deliver to customer).
Catalogue/CatalogueLine/RequiredItemLocationQuantity/DeliveryUnit/ConsumerUnitQuantity@unitCode			Specifies the units of measurement for the ConsumerUnitQuantity. The value must be a valid code. For example, "BO" for bottle.

2.1.3 Item

Under the *Item* class (in the Catalogue document) are packing specifications related to quantities supplied:

UK-name	DK-name	Use	Remarks
Catalogue/CatalogueLine/Item/PackQuantity	PakkeMængde	0..1	The quantity of the supplied packaging of the item. This may contain a number of individual units as described in PackSizeNumeric (below). For example, if the item is supplied as a case of 12, then the PackQuantity would be "1" and unitCode would be for cases.
Catalogue/CatalogueLine/Item/PackQuantity@unitCode			Specifies the units of measurement for PackQuantity. The value must be a valid code. For example, "CS" for case.
Catalogue/CatalogueLine/Item/PackSizeNumeric	PakkeStørrelse	0..1	The number of units in one pack of the Item. Using the previous example this would be specified as "12".

Price discounts may be specified on a Catalogue line using the *Price* within the *AllowanceCharge* class. See the general guideline, OIOUBL Guideline Discounts and Charges (Ref. G17) for a more detailed description of how to define discounts.

3. Description

The following section describes the different classes and elements related to prices and quantities.

3.1 Relationship between the Price and Quantity elements

The below figure describes the overall relationship between *Price* and *Quantity* classes and elements.

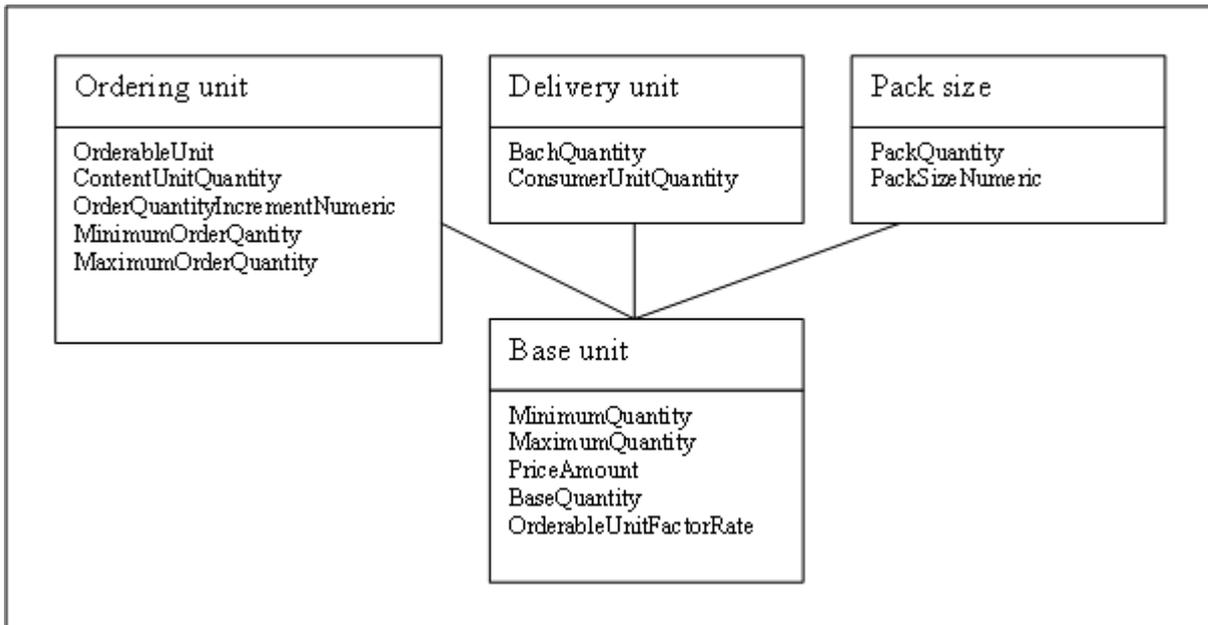


Figure 1: Relationships between Price and Quantity

The common point of reference is the base unit. This is established as the supplier's base units (*BaseQuantity*) and base price (*Price*). These are defined in the *RequiredItemLocationQuantity* class together with the information on delivery units. The *OrderableUnits* are specified based on the base units.

Note that it is also possible to define different pack sizes for a given item that may not be related to pricing. The pack sizes are specified in the *Item* class.

3.2 Base Unit

When working with prices and quantities, the *Price* class is the foundation for all other classes and fields. The example below shows how the *Price* class may be entered:

```
<cac:RequiredItemLocationQuantity>
  <cbc:MinimumQuantity unitCode="B0">1</cbc:MinimumQuantity>
  <cbc:MaximumQuantity unitCode="B0">100</cbc:MaximumQuantity>
  <cac:Price>
    <cbc:PriceAmount currencyID="DKK">60.00</cbc:PriceAmount>
    <cbc:BaseQuantity unitCode="B0">1</cbc:BaseQuantity>
    <cbc:OrderableUnitFactorRate>12</cbc:OrderableUnitFactorRate>
  </cac:Price>
</cac:RequiredItemLocationQuantity>
```

```

        <cac:ValidityPeriod>
            <cbc:StartDate>2006-08-01</cbc:StartDate>
            <cbc:EndDate>2007-07-31</cbc:EndDate>
        </cac:ValidityPeriod>
    </cac:Price>
</cac:RequiredItemLocationQuantity>

```

Figure 2: Example of the Price class

In this example, the *MinimumQuantity* and the *MaximumQuantity* specify that the price applies to purchases of between 1 and 100 bottles of the item. The item is one bottle at a price of DKK 60.00. The *BaseQuantity* defines the supplier's base unit, that is, the units of measurement that the supplier's goods are sold by. With this item, each orderable unit (*OrderableUnitFactorRate*) actually contains 12 bottles. Furthermore, the example shows that the price is valid from August 1st, 2006 to July 31st, 2007.

3.3 Delivery Unit

The example below shows how the delivery unit information may be entered:

```

<cac:RequiredItemLocationQuantity>
    <cac:DeliveryUnit>
        <cbc:BatchQuantity unitCode="CS">1</cbc:BatchQuantity>
        <cbc:ConsumerUnitQuantity unitCode="BO">12</cbc:ConsumerUnitQuantity>
    </cac:DeliveryUnit>
    <cac:DeliveryUnit>
        <cbc:BatchQuantity unitCode="PF">1</cbc:BatchQuantity>
        <cbc:ConsumerUnitQuantity unitCode="BO">120</cbc:ConsumerUnitQuantity>
    </cac:DeliveryUnit>
</cac:RequiredItemLocationQuantity>

```

Figure 3: Example of DeliveryUnit

Information about the delivery units (*DeliveryUnit*) is specified in the *RequiredItemLocationQuantity* class.

The *DeliveryUnit* class contains information to the buyer about the available delivery units of an item. The *BatchQuantity* in the example shows that the item may be delivered in cases (code "CS") and pallets (code "PF").

ConsumerUnitQuantity specifies how many units the batch consists of. In this example, a case consists of 12 bottles (code "BO"), and a pallet consists of 120 bottles (code "BO").

The *ConsumerUnitQuantity* should be defined in terms of the *BaseQuantity*. In fact, in many cases they will be the same.

3.4 OrderableUnit

Within each catalogue line is information used for ordering the item specified, as shown in the

example below:

```
<cac:CatalogueLine>
  <cbc:ID>12457812-FR123</cbc:ID>
  <cbc:ActionCode listAgencyID="320" listID="urn:oioubl:odelist:catalogueactioncode-1.1">Add</cbc:ActionCode>
  <cbc:Note>12 flasker pr. kasse</cbc:Note>
  <cbc:OrderableIndicator>true</cbc:OrderableIndicator>
  <cbc:OrderableUnit>CS</cbc:OrderableUnit>
  <cbc:ContentUnitQuantity unitCode="BO">12</cbc:ContentUnitQuantity>
  <cbc:OrderQuantityIncrementNumeric>1</cbc:OrderQuantityIncrementNumeric>
  <cbc:MinimumOrderQuantity unitCode="CS">1</cbc:MinimumOrderQuantity>
  <cbc:MaximumOrderQuantity unitCode="CS">10</cbc:MaximumOrderQuantity>
  ...
</cac:CatalogueLine>
```

Figure 4: Example of OrderableUnit

The *ID* uniquely identifies the item in question. The "Add" for an *ActionCode* specifies that the item should be added, either to an existing or new catalogue. For a more detailed description of ID, and item identification see the OIOUBL Guideline Catalogue Identification, Versioning and Validity Periods (Ref. G37).

In the *Note* element, an optional text may be defined for the catalogue line. In the example it contains a description of the orderable unit.

The *OrderableIndicator* indicates whether an item can be ordered or not. The default value is "true", meaning the item is orderable. A value of "false" means that the item cannot be ordered. This may be the case when, for example, catalogues that are frequently being updated. It may be convenient for the supplier to be able to notify the customer when an item is temporarily out of stock, or to inform about seasonal items, without having to delete the item from the catalogue.

If the *OrderableIndicator* is "true", then the orderable unit **must** be defined. The orderable unit is specified in *OrderableUnit*, using a valid units of measure code. For example "CS" for case, as shown in the example. The orderable unit is the unit that the customer will receive when ordering a quantity of "1" from the catalogue.

A direct relation exists between *OrderableUnit*, *PriceAmount*, *BaseQuantity*, and *OrderableUnitFactorRate* (as described in the above in section 4.1 "Relations between the Price and Quantity fields"). This means that:

$$BaseQuantity * OrderableUnitFactorRate = the\ quantity\ specified\ by\ OrderableUnit$$

Accordingly, if the *BaseQuantity* is "1" (and units are "BO" for bottles), then the *OrderableUnitFactorRate* is "12", and the *OrderableUnit* is "CS". This means the order quantity "CS" in *OrderableUnit* is "1 case" (which contains 12 bottles).

The price for the orderable unit is calculated likewise:

$$PriceAmount / BaseQuantity * (BaseQuantity * OrderableUnitFactorRate) = The\ price\ of\ one\ orderable\ unit$$

This expression can be reduced to:

$$PriceAmount * OrderableUnitFactorRate = Price\ of\ one\ orderable\ unit$$

For example, if *PriceAmount* is "65.00" (DKK), *BaseQuantity* is "1", and *OrderableUnitFactorRate* is "12", then the price is 780.00 (DKK) for a case of 12 bottles, and this is the orderable unit.

ContentUnitQuantity defines what comprises an *OrderableUnit*. In this example, the case ("CS") consists of 12 bottles ("BO").

OrderQuantityIncrementNumeric should be stated in relation to the *OrderableUnit*. In the example, the orderable unit is 1 case (of 12 bottles), any order quantity increments will be stated in cases.

In another situation where the base unit and the orderable unit may both specify 1 bottle, and the item is still only sold in cases of 12 bottles, there is the option to specify a *MinimumOrderQuantity* and an *OrderQuantityIncrementNumeric*. The difference is that if the minimum order quantity is set to 12 bottles, it means a minimum of 12 bottles must be ordered, but that purchases of 13, 14 or 15 bottles are also possible. If the *OrderQuantityIncrementNumeric* is then set to 12, the item must be ordered in quantities of 12, 24, 36 bottles, etc.

If the *MaximumOrderQuantity* element is specified then a customer can only order a limited number of units.



Note the difference between *MinimumOrderQuantity* and *MaximumOrderQuantity*, which are related to the quantities orderable, and *MinimumQuantity* and *MaximumQuantity*, which are related to the price for a given quantity.

3.5 PackSizeNumeric

Another two elements in the Item class relate to the specification of units. These are the *PackQuantity* and the *PackSizeNumeric*.

```
<cac:Item>
...
<cbc:PackQuantity unitCode="CS">1</cbc:PackQuantity>
<cbc:PackSizeNumeric>12</cbc:PackSizeNumeric>
...
</cac:Item>
```

Figur 5: Example of PackQuantity and PackSizeNumeric

PackQuantity is an expression of the packing, in this example it is "1 case (CS)". *PackSizeNumeric* specifies how many units are in a package, in this example, "12".

The specification of packing is only defined where the item is described. But it has an impact on other quantity specifications.

PackSizeNumeric has the following relation to *Price/BaseQuantity*:

$$BaseQuantity * PackSizeNumeric = the\ quantity\ expressed\ using\ PackQuantity@unitCode$$

Where *PackQuantity* is the quantity that the package actually contains.

4. Examples

This section contains different examples of how to use the price and quantity elements, as well as demonstrating some of the relationships between them.

4.1 Converting BaseUnit to OrderableUnit

In some business sectors the orderable unit and the billing unit of an item are not identical. For example, oil is ordered in barrels but settled in liters, meat may be ordered as joints or carcasses and settled by weight, and a length measure of steel is settled in kilograms. In the OIOUBL catalogue documents it is possible to manage these differences.

This first example describes how a single pack of pork chops weighing 2 kilos is specified as being available for ordering. Each pack containing 12 chops. The supplier's base unit is kilos, and the price of one kilo is DKK 50.00.

```
<cac:CatalogueLine>
  <cbc:ID>78945612-4545</cbc:ID>
  <cbc:ActionCode listAgencyID="320" listID="urn:oioubl:codelist:catalogueactioncode-1.1">Add</cbc:ActionCode>
  <cbc:Note>12 chops per package</cbc:Note>
  <cbc:OrderableIndicator>true</cbc:OrderableIndicator>
  <cbc:OrderableUnit>PK</cbc:OrderableUnit>
  <cbc:ContentUnitQuantity unitCode="EA">12</cbc:ContentUnitQuantity>
  <cac:RequiredItemLocationQuantity>
    <cbc:LeadTimeMeasure unitCode="DAY">1</cbc:LeadTimeMeasure>
    <cac:Price>
      <cbc:PriceAmount currencyID="DKK">50.00</cbc:PriceAmount>
      <cbc:BaseQuantity unitCode="KGM">1</cbc:BaseQuantity>
      <cbc:OrderableUnitFactorRate>2</cbc:OrderableUnitFactorRate>
    </cac:Price>
    <cac:DeliveryUnit>
      <cbc:BatchQuantity unitCode="PK">1</cbc:BatchQuantity>
      <cbc:ConsumerUnitQuantity unitCode="KGM">2</cbc:ConsumerUnitQuantity>
    </cac:DeliveryUnit>
    ...
  </cac:RequiredItemLocationQuantity>
  ...
</cac:CatalogueLine>
```

Figure 6: Example of unit conversion

In this example, *OrderableUnit* specifies that the basic unit for ordering is a single pack, and *ContentUnitQuantity* details that each pack contains 12 pieces.

Alternatively, if it is not possible to specify the exact number pork chops in each pack, the *ContentUnitQuantity* may be specified as weight (for example, 2 KGM), and the supplier can provide a text description of the contents in the *Note* field, such as "10 – 12 pcs. per pack".

PriceAmount and *BaseQuantity* specify that the price is DKK 50.00 for one kilo. As one package is 2 kilos, the supplier specifies in *OrderableUnitFactorRate* which factor the base unit should be multiplied with to calculate the orderable unit. In this example it is "2".

The price of one package may be calculated as:

$$\text{PriceAmount} / \text{BaseQuantity} * (\text{BaseQuantity} * \text{OrderableUnitFactorRate}) = \text{The price of one OrderableUnit}$$

In the example above, this is...

$$50.00 / 1 * (1 * 2) = \text{DKK } 100.00 \text{ per pack.}$$

In another example, the catalogue specifies orders may be placed for one barrel of oil (containing 750 litres) and payment is based on the supplier's base unit (which is based on liters).

```
<cac:CatalogueLine>
  <cbc:ID>22334455-999</cbc:ID>
  <cbc:ActionCode llistAgencyID="320" listID="urn:oioubl:odelist:catalogueactioncode-1.1">Add
</cbc:ActionCode>
  <cbc:Note>One barrel of oil containing 750 liters</cbc:Note>
  <cbc:OrderableIndicator>true</cbc:OrderableIndicator>
  <cbc:OrderableUnit>BLL</cbc:OrderableUnit>
  <cbc:ContentUnitQuantity unitCode="LTR">750</cbc:ContentUnitQuantity>
  <cac:RequiredItemLocationQuantity>
    <cbc:LeadTimeMeasure unitCode="DAY">3</cbc:LeadTimeMeasure>
    <cac:Price>
      <cbc:PriceAmount currencyID="DKK">480.00</cbc:PriceAmount>
      <cbc:BaseQuantity unitCode="LTR">1000</cbc:BaseQuantity>
      <cbc:OrderableUnitFactorRate>0.75</cbc:OrderableUnitFactorRate>
    </cac:Price>
    <cac:DeliveryUnit>
      <cbc:BatchQuantity unitCode="BLL">1</cbc:BatchQuantity>
      <cbc:ConsumerUnitQuantity unitCode="LTR">750</cbc:ConsumerUnitQuantity>
    </cac:DeliveryUnit>
    ...
  </cac:RequiredItemLocationQuantity>
  ...
</cac:CatalogueLine>
```

Figure 7: Second example of unit conversion

In the fragment shown in Figure 7, the units of measurement code, "BLL" in *OrderableUnit* means that the orderable unit is one barrel, and the *ContentUnitQuantity* specifies that the barrel contains 750 litres (LTR).

The supplier's base price (*PriceAmount*) is DKK 480.00 for 1000 litres (the *BaseQuantity*). Because the supplier does not sell the oil in kilolitres, but in barrels of 750 litres, the supplier must specify the conversion factor (*OrderableUnitFactorRate*) that should be applied to convert the supplier's base unit of kilolitres to the orderable unit of barrels containing 750 litres. So in this case, the *OrderableUnitFactorRate* is 0.75. (1000 litres * 0.75 = 750 litres ≈ 1 barrel).

The price of one barrel of oil is calculated by multiplying the supplier's base price (*PriceAmount*) by the *OrderableUnitFactorRate*, i.e. DKK 480.00 * 0.75 = DKK 360.00 per barrel.

4.2 Quantity dependent pricing

In OIOUBL-2.0 it is possible to specify different price and quantity combinations for catalogue items. This means, for example, that a supplier can specify one price if the customer buys from 1 to 100 units of an item, and a different price for purchases of more than 100 units. This is sometimes known as setting price-breaks.

The different prices are handled by defining a new *RequiredItemLocationQuantity* class for each price/quantity combination. In the following example, the document fragment states that if the customer purchases from 1 to 60 bottles the price is DKK 65.00 per bottle, and for purchases between 61 and 120 bottles the price is DKK 60.00 per bottle.

```

...
<cac:RequiredItemLocationQuantity>
  <cbc:LeadTimeMeasure unitCode="DAY">3</cbc:LeadTimeMeasure>
  <cbc:MinimumQuantity unitCode="BO">1</cbc:MinimumQuantity>
  <cbc:MaximumQuantity unitCode="BO">60</cbc:MaximumQuantity>
  <cac:Price>
    <cbc:PriceAmount currencyID="DKK">65.00</cbc:PriceAmount>
    <cbc:BaseQuantity unitCode="BO">1</cbc:BaseQuantity>
    <cbc:OrderableUnitFactorRate>12</cbc:OrderableUnitFactorRate>
    <cac:ValidityPeriod>
      <cbc:StartDate>2006-08-01</cbc:StartDate>
      <cbc:EndDate>2007-07-31</cbc:EndDate>
    </cac:ValidityPeriod>
  </cac:Price>
  ...
</cac:RequiredItemLocationQuantity>
<cac:RequiredItemLocationQuantity>
  <cbc:LeadTimeMeasure unitCode="DAY">3</cbc:LeadTimeMeasure>
  <cbc:MinimumQuantity unitCode="BO">61</cbc:MinimumQuantity>
  <cbc:MaximumQuantity unitCode="BO">120</cbc:MaximumQuantity>
  <cac:Price>
    <cbc:PriceAmount currencyID="DKK">60.00</cbc:PriceAmount>
    <cbc:BaseQuantity unitCode="BO">1</cbc:BaseQuantity>
    <cbc:OrderableUnitFactorRate>12</cbc:OrderableUnitFactorRate>
    <cac:ValidityPeriod>
      <cbc:StartDate>2006-08-01</cbc:StartDate>
      <cbc:EndDate>2007-07-31</cbc:EndDate>
    </cac:ValidityPeriod>
  </cac:Price>
  ...
</cac:RequiredItemLocationQuantity>

```

Figure 8: Example of different price/quantity combinations

The *MinimumQuantity* and *MaximumQuantity* specify the range within which the *PriceAmount* for the *RequiredItemLocationQuantity* applies. In the first, the range is 1 to 60 bottles, and in the second the range is 61 to 120 bottles.

If no upper limit is required, for example, when the price of DKK 60.00 applies to all quantities greater than 60 bottles, the *MaximumQuantity* should be omitted.

Note that the *OrderableUnitFactorRate* in this example is 12. This means that the *OrderableUnit* will be in groups of 12 bottles. In other words, customers must order complete cases (*OrderableUnit* of "CS"). The results will be in quantities of 12, 24, 36 bottles, etc. Accordingly, the order increment does not correspond to the range specified for the various price/quantities. So in this situation it is not actually possible to order 61 bottles.

4.3 Period dependent pricing

In many situations a supplier will use a price update to change the price of catalogue items. But it is also possible to create period dependent prices, such as a planned price rise for a seasonal item or discounts for a sale period.

Period dependent pricing can normally only be applied after prior agreement with the Customer, as the receiver's catalogue must be prepared to handle the price change.

The method is very similar to the method described for quantity dependent prices. A *RequiredItemLocationQuantity* is repeated for each new combination. The difference is that the validity period is changed, and not the minimum/maximum range.

In the following example (Figure 9.), the price of 1 unit of an item is DKK 25.00 from January 1st, 2006 to May 31st, 2006. From June 1st, 2006 the price rises to DKK 30.00, and this price is valid until December 31st, 2006.

```
...
<cac:RequiredItemLocationQuantity>
  <cbc:LeadTimeMeasure unitCode="DAY">3</cbc:LeadTimeMeasure>
  <cac:Price>
    <cbc:PriceAmount currencyID="DKK">25.00</cbc:PriceAmount>
    <cbc:BaseQuantity unitCode="EA">1</cbc:BaseQuantity>
    <cbc:OrderableUnitFactorRate>1</cbc:OrderableUnitFactorRate>
    <cac:ValidityPeriod>
      <cbc:StartDate>2006-01-01</cbc:StartDate>
      <cbc:EndDate>2006-05-31</cbc:EndDate>
    </cac:ValidityPeriod>
  </cac:Price>
  ...
</cac:RequiredItemLocationQuantity>
<cac:RequiredItemLocationQuantity>
  <cbc:LeadTimeMeasure unitCode="DAY">3</cbc:LeadTimeMeasure>
  <cac:Price>
    <cbc:PriceAmount currencyID="DKK">30.00</cbc:PriceAmount>
```

```

    <cbc:BaseQuantity unitCode="EA">1</cbc:BaseQuantity>
    <cbc:OrderableUnitFactorRate>1</cbc:OrderableUnitFactorRate>
    <cac:ValidityPeriod>
      <cbc:StartDate>2006-06-01</cbc:StartDate>
      <cbc:EndDate>2006-12-31</cbc:EndDate>
    </cac:ValidityPeriod>
  </cac:Price>
  ...
</cac:RequiredItemLocationQuantity>
...

```

Figure 9: Example of period dependent pricing

Within the *RequiredItemLocationQuantity*, the *ValidityPeriod StartDate* and *EndDate* is defined for the appropriate *PriceAmount*.



There can be no overlap between the specified validity periods, as this would create an ambiguous pricing for the item.

4.4 Specifying customer dependent prices

In OIOUBL the same catalogue may specify different prices depending on who the potential customer for the item is. This may be the situation where a supplier has agreed a special price (or other special terms) with one customer, while the same item is sold to other customers at a different price (or terms).

There are two options. The first option is to attach a specific customer (*ContractorCustomerParty*) to the catalogue line. This may be the case where the catalogue is sent to a marketplace that acts as a hub. Another option is to geographically delimit which customers a given price applies to.

The customer-dependent prices should only be used when a prior agreement exists between the catalogue provider and receiver.

4.4.1 Specifying the Customer on the Catalogue Line

When working with portals or marketplaces, it may be necessary to set up a relationship between a customer and the price of an individual line item in a catalogue.

In such a situation, a customer (*ContractorCustomerParty*) may be specified on the catalogue line, as shown in Figure 10 below.

```

<cac:CatalogueLine>
  <cbc:ID>NM-457896432</cbc:ID>
  <cbc:ActionCode istAgencyID="320" listID="urn:oioubl:odelist:catalogueactioncode-1.1">Add</cbc:ActionCode>
  <cbc:OrderableIndicator>true</cbc:OrderableIndicator>
  <cbc:OrderableUnit>EA</cbc:OrderableUnit>
  <cbc:ContentUnitQuantity unitCode="EA">1</cbc:ContentUnitQuantity>
  <cac:ContractorCustomerParty>

```

```

<cac:Party>
  <cbc:EndpointID schemeAgencyID="9" schemeDataURI="urn:oioubl:scheme:
endpointid-1.0" schemeID="GLN">5798000416604</cbc:EndpointID>
  <cac:PartyIdentification>
    <cbc:ID schemeAgencyID="9" schemeID="GLN">5798000416604</cbc:ID>
  </cac:PartyIdentification>
  <cac:PartyName>
    <cbc:Name>Den Lille Skole</cbc:Name>
  </cac:PartyName>
  <cac:PostalAddress>
    <cbc:AddressFormatCode listAgencyID="320" listID="urn:oioubl:
codelist:addressformatcode-1.1">StructuredDK</cbc:AddressFormatCode>
    <cbc:StreetName>Fredericiavej</cbc:StreetName>
    <cbc:BuildingNumber>10</cbc:BuildingNumber>
    <cbc:CityName>Helsingør</cbc:CityName>
    <cbc:PostalZone>3000</cbc:PostalZone>
    <cac:Country>
      <cbc:IdentificationCode>DK</cbc:IdentificationCode>
    </cac:Country>
  </cac:PostalAddress>
  <cac:PartyLegalEntity>
    </cbc:CompanyID schemeID="DK:CVR">DK65656565</cbc:CompanyID>
  </cac:PartyLegalEntity>
  <cac:Contact>
    <cbc:ID>9000123456</cbc:ID>
  </cac:Contact>
</cac:Party>
</cac:ContractorCustomerParty>
<cac:RequiredItemLocationQuantity>
  <cbc:LeadTimeMeasure unitCode="DAY">3</cbc:LeadTimeMeasure>
  <cbc:HazardousRiskIndicator>>false</cbc:HazardousRiskIndicator>
  <cac:Price>
    <cbc:PriceAmount currencyID="DKK">799.00</cbc:PriceAmount>
    <cbc:BaseQuantity unitCode="EA">1</cbc:BaseQuantity>
    <cbc:OrderableUnitFactorRate>1</cbc:OrderableUnitFactorRate>
  </cac:Price>
  ...
</cac:RequiredItemLocationQuantity>
  ...
</cac:CatalogueLine>

```

Figure 10: Example of a specific customer for a catalogue line

In the example "Den Lille Skole" is defined as the customer (*ContractorCustomerParty*). This means that the price (*PriceAmount*) and the other information (such as *OrderableUnit* and *LeadTimeMeasure*, etc) entered on the subsequent catalogue line are specific to this customer.

If the same supplier sells the same item to another customer on different terms, another catalogue line is created. The new customer in question is defined as the *ContractorCustomerParty*, and the remaining elements are specified according to the agreement with this customer.

The same item ID (*SellersItemIdentification/ID*) can apply to all lines, specified using the *Item* class, as shown in the example:

```
<cac:Item>
  <cbc:Description>Nokia Mobile telephone - Type ABC</cbc:Description>
  <cbc:PackQuantity unitCode="EA">1</cbc:PackQuantity>
  <cbc:PackSizeNumeric>1</cbc:PackSizeNumeric>
  <cac:SellersItemIdentification>
    <cbc:ID>87067606</cbc:ID>
  </cac:SellersItemIdentification>
  ...
</cac:Item>
```

Figure 11: Example of Item ID

Note that each *CatalogueLine/ID* must be unique. This means that, even if the seller and the item are the same on the catalogue lines, it is not the same line. Therefore the ID must be different for each line, to distinguish between the two. For more details, see OIOUBL Catalogue Identification, Versioning and Validity Periods (Ref. G37).

4.4.2 Region dependent Pricing

There is also the option of differentiating the prices according to a specific region.

In this case, using the *RequiredItemLocationQuantity* class, one or more regions (or addresses) may be specified (using the *ApplicableTerritoryAddress*) to which the price is applicable.

For example, a supplier may sell the same item to two high schools, but at different prices. A high school in Aalborg may be able to buy the item at DKK 399.00, whilst the high school in Århus will pay DKK 449.00 for the same item.

If a high school in Brønderslev were able to buy the item at the same price as the high school in Aalborg, another *ApplicableTerritoryAddress* would be included in the *RequiredItemLocationQuantity*, for which the price is also DKK 399.00.

The place or region where the price is applicable can be identified in many different ways. For example, it may be:

- an address (using *StreetName*, *BuildingNumber*, *CityName* and *PostalZone*),
- identified from an address register (using just the *ID*),
- it may be an extended region, such as a city or postal zone (using *CityName* or *PostalZone*),
- an entire country (using *Country/IdentificationCode*).

The address format (*AddressFormatCode*) should be specified as "StructuredDK" (See the OIOUBL Guideline Addresses, Ref. G36). Figure 12 is an example of regional dependent pricing.

```

<cac:CatalogueLine>
  ...
  <cac:RequiredItemLocationQuantity>
    <cbc:LeadTimeMeasure unitCode="DAY">3</cbc:LeadTimeMeasure>
    <cac:ApplicableTerritoryAddress>
      <cbc:AddressFormatCode listAgencyID="320" listID="urn:oioubl:codelist:
addressformatcode-1.1">StructuredDK</cbc:AddressFormatCode>
      <cbc:StreetName>Skolevangen</cbc:StreetName>
      <cbc:BuildingNumber>12</cbc:BuildingNumber>
      <cbc:CityName>Aalborg</cbc:CityName>
      <cbc:PostalZone>9000</cbc:PostalZone>
      <cac:Country>
        <cbc:IdentificationCode>DK</cbc:IdentificationCode>
      </cac:Country>
    </cac:ApplicableTerritoryAddress>
    <cac:Price>
      <cbc:PriceAmount currencyID="DKK">399.00</cbc:PriceAmount>
      <cbc:BaseQuantity unitCode="EA">1</cbc:BaseQuantity>
    </cac:Price>
    <cac:DeliveryUnit>
      <cbc:BatchQuantity unitCode="EA">1</cbc:BatchQuantity>
      <cbc:ConsumerUnitQuantity unitCode="EA">1</cbc:ConsumerUnitQuantity>
    </cac:DeliveryUnit>
  </cac:RequiredItemLocationQuantity>
  <cac:RequiredItemLocationQuantity>
    <cbc:LeadTimeMeasure unitCode="DAY">3</cbc:LeadTimeMeasure>
    <cac:ApplicableTerritoryAddress>
      <cbc:AddressFormatCode listAgencyID="320" listID="urn:oioubl:codelist:
addressformatcode-1.1">StructuredDK</cbc:AddressFormatCode>
      <cbc:StreetName>Gymnasiegade</cbc:StreetName>
      <cbc:BuildingNumber>1</cbc:BuildingNumber>
      <cbc:CityName>Århus</cbc:CityName>
      <cbc:PostalZone>8000</cbc:PostalZone>
      <cac:Country>
        <cbc:IdentificationCode>DK</cbc:IdentificationCode>
      </cac:Country>
    </cac:ApplicableTerritoryAddress>
    <cac:Price>
      <cbc:PriceAmount currencyID="DKK">449.00</cbc:PriceAmount>
      <cbc:BaseQuantity unitCode="EA">1</cbc:BaseQuantity>
      <cbc:OrderableUnitFactorRate>1</cbc:OrderableUnitFactorRate>
    </cac:Price>
    <cac:DeliveryUnit>
      <cbc:BatchQuantity unitCode="EA">1</cbc:BatchQuantity>

```

```

        <cbc:ConsumerUnitQuantity unitCode="EA">1</cbc:ConsumerUnitQuantity>
    </cac:DeliveryUnit>
</cac:RequiredItemLocationQuantity>
...
</cac:CatalogueLine>

```

Figure 12: Example of regional dependent pricing

Alternatively, it is possible to define a region using free text in the *AddressLine* element. In this case, the address format (*AddressFormatCode*) is specified as "Unstructured".

4.5 Specifying List Prices

The supplier has the option to specify a list price as a supplement to the purchase price, which makes it possible for the customer to compare the two prices.

As prices can only be specified in one place in the catalogue (under *RequiredItemLocationQuantity/Price/PriceAmount*) a code must be used to distinguish the purchase price from the list price. This is specified by *RequiredItemLocationQuantity/Price/PriceTypeCode*.

If a line item needs such a supplementary list price, the *RequiredItemLocationQuantity* class is repeated. The purchase price is specified in the first *RequiredItemLocationQuantity*. If no code is specified in *PriceTypeCode*, the price is considered to be the purchase price. That is, purchase price is the default type. The list price is then specified in the second *RequiredItemLocationQuantity*. Here the *PriceTypeCode* is set to the code for the type of list price (as per code list K14).

```

<cac:RequiredItemLocationQuantity>
    <cbc:LeadTimeMeasure unitCode="DAY">3</cbc:LeadTimeMeasure>
    <cac:Price>
        <cbc:PriceAmount currencyID="DKK">25.00</cbc:PriceAmount>
        <cbc:BaseQuantity unitCode="EA">1</cbc:BaseQuantity>
        <cbc:OrderableUnitFactorRate>1</cbc:OrderableUnitFactorRate>
    </cac:Price>
    ...
</cac:RequiredItemLocationQuantity>
<cac:RequiredItemLocationQuantity>
    <cbc:LeadTimeMeasure unitCode="DAY">3</cbc:LeadTimeMeasure>
    <cac:Price>
        <cbc:PriceAmount currencyID="DKK">30.00</cbc:PriceAmount>
        <cbc:BaseQuantity unitCode="EA">1</cbc:BaseQuantity>
        <cbc:PriceTypeCode listAgencyID="6" listID="UN/ECE 5387>DR</cbc:PriceTypeCode>
        <cbc:OrderableUnitFactorRate>1</cbc:OrderableUnitFactorRate>
        <cac:PriceList>
            <cbc:ID>12345</cbc:ID>
        </cac:PriceList>
    </cac:Price>
</cac:RequiredItemLocationQuantity>

```

Figure 13: Example of specifying a list price

In addition to the specification of list price, a price list may be referenced in the *PriceList* class, the reference being the *ID*.

5. Relevant code lists

Code list:	Agency:	Urn:	Example value:
ProfileID	320	urn:oiubl:id:profileid-1.1	Catalogue-CatAdv-1.0
EndpointID	320	urn:oiubl:scheme:endpointid-1.1	GLN
PartyIdentification/ID	320	urn:oiubl:scheme:partyidentificationid-1.1	DK:CVR
CurrencyCode	6	ISO 4217 Alpha	DKK, EUR
UnitOfMeasureCode	6	UN/ECE rec 20	PK, EA
PriceTypeCode	6	UN/ECE 5387	CAT, DR
CatalogueActionCode	320	urn:oiubl:codelist:catalogueactioncode-1.1	Add, Delete, Update
AddressFormatCode	320	urn:oiubl:codelist:addressformatcode-1.1	StructuredLax

6. Terms and abbreviations

Listed below are the most important terms and abbreviations:

Term:	Explanation:
Document level	Elements at document level are found directly under the root element (the top element) in the XML structure. elements at the document level apply to the whole document.
Line level	Elements at line level, unlike elements at the document level, only apply to a specific transaction line
Class	A class is a collection of elements. For example, the Price class contains elements such as PriceAmount, BaseQuantity, etc.
Element	An element is an information entity in an XML structure. For example, the PriceAmount is the element containing the price in an invoice line.
Attributes	In an XML element, it is possible to specify a property as an attribute, e. g. the attribute unitCode in which the unit for a quantity may be specified, as in the example: <cbc:BaseQuantity unitCode="BO">1</cbc:BaseQuantity> Attributterne benyttes også til at angive kodelister f.eks. listID="urn:oiubl:codelist:addressformatcode-1.1"