EDItEUR

ONIX for Books

Product Information Message

Application Note: Pricing in ONIX 3.0, part 1

The section of ONIX that defines the price of a particular product is – for many users of the standard – the most obvious single reason for adopting ONIX. The standard is designed to support unambiguous communication of *product* metadata through the supply chain. While ONIX is used widely for exchange of purely bibliographic data, it is expressly designed for *commercial* use cases.

But pricing of products is an area that has changed almost beyond recognition since ONIX was created in the late 1990s. Then, a product had a price. Perhaps a handful of prices, each relevant to a particular 'market'. Prices – in particular, the prices of digital products – are considerably more complex. Thus the ONIX pricing model has evolved, even though in some instances, the internal order processing systems used across the industry have not kept pace.

Basic pricing

The basic commercial model used widely across the industry is itself an unusual one. Where many industries have business-to-business (wholesale) prices that are separate from their retail (business-to-consumer) prices, in many countries books are sold with a 'trade discount'. That is, the business-to-business price is not stated explicitly, but linked mathematically to the stated retail price via a discount percentage. This may be due to the fact that many countries have or had a 'fixed retail price' law or agreement for book sales, or to the desire to set different business-to-business prices for different business customers.

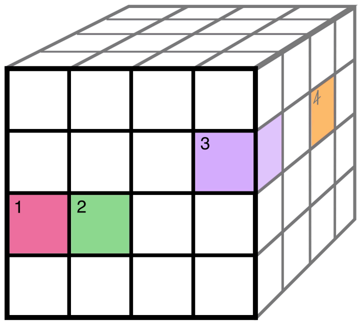
So for ONIX, this means that we have several ways of stating a price:

- Recommended retail price (with an optional discount for b2b transactions – the discount is in effect the retailer's gross margin)
- Fixed retail price (with an optional discount for b2b transactions)
- Wholesale price (with an optional fixed or recommended retail price – the difference between the wholesale and retail prices is the retailer's gross margin)
- With e-books, there is another common price type – agency prices. From the point of view of the retailer, this is a fixed price, and instead of a discount on b2b transactions, the retailer earns a *commission* on sales.

Each of these price types can be state either with or without the relevant retail-level taxes. In ONIX, these different types are stated explicitly in the <PriceType> data element.

The US book trade uses RRPs without tax (*ie* there are no tax details in the ONIX, as the sales tax is only added at the store checkout – and it varies from state to state and city to city). France – where fixed price laws apply – mostly uses FRPs including tax. The UK book trade uses RRPs including tax. But it also occasionally quotes RRPs *without* tax for some types of customer (for example when selling into the schools market).

Now price type and country are not the only two variables. A single product can have many prices – each with a unique combination of type, country, currency, date (since prices can vary through time), customer type and so on.



One way to visualize this is that all the prices for a product can be assembled into a cube. Each small cell in the big cube is one price. The sides of the cube represent different countries, different price or customer types, different dates and so on. You could say that the vertical side of the cube represents the countries, so all US prices are on the bottom 'layer', all Canadian prices on the second layer, all Mexican prices on the third layer and so on. Meanwhile the horizontal axis might be the price type – RRPs without tax in the leftmost slice, special schools prices in the next slice across and so on. The cell highlighted in pink (#1) is thus the retail price in Canada and the adjacent green cell (#2) is a special schools price in Canada. The purple cell (#3) could be the current Mexican agency price, and – if the front-to-back axis represents time – the orange cell (#4) could represent the Mexican agency price in three months time.

Thus every cell is a price – say '7.99' or '8.45'– and the *position* of that cell in the big cube controls how, where, when and to whom that price applies.

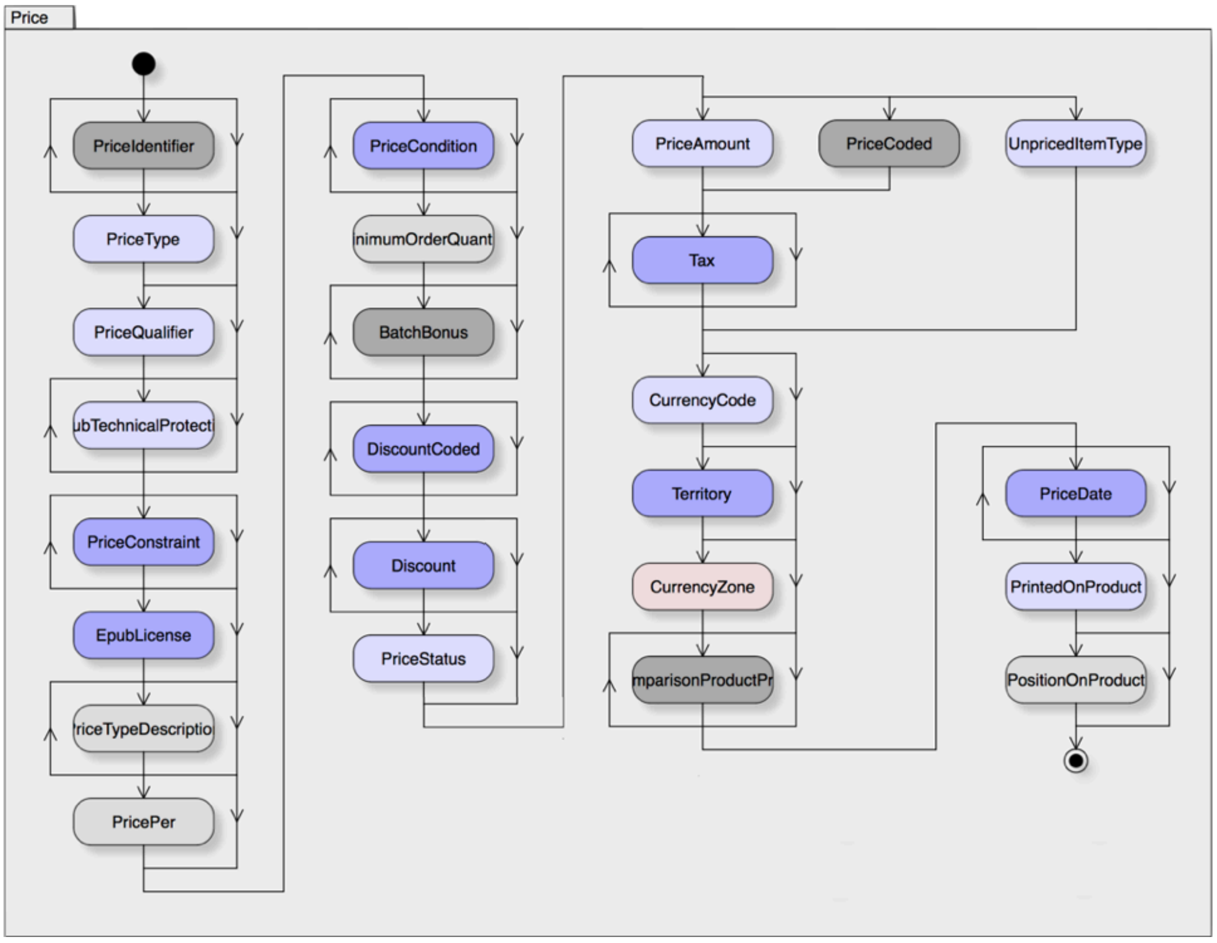
What may be harder to visualize is that the cube is not just three dimensional – it's really a 'hypercube' with up to seven or eight dimensions – but in the ONIX this doesn't matter. Each cell in the big cube is described individually in one <Price> composite, and so the big cube is just a long list of repeated <Price> composites. Each composite contains its own 'position' within the big cube.

It is not unusual to have a list of five, ten or twenty different <Price> composites. Any one <Price> composite contains the actual money amount – or perhaps a special ONIX element showing the item is free of charge or a price has not yet been set, or an Apple-style price tier. And the <Price> composite also contains the cell's unique position in the big cube based on a combination of:

- Price type and price qualifier (which control the terms of sale and the end customer type)
- Territory (country or countries where the price is valid)
- Currency (the denomination of the price)
- Dates when the price is valid
- Price conditions (*eg this* price is only valid if the end customer already purchased *that*)
- Price constraints and licensing (for e-books where the usage constraints vary according to how much the end customer pays)
- Minimum order quantity

Given the whole range of prices, the retailer can then work out the single price that is relevant in a particular transaction.

The <Price> composite in ONIX



As you can see from the diagram above which lists all the ONIX data elements that can occur within the <Price> composite, a single <Price> is a large and complex structure. (Note that darker blue and grey ovals are themselves composites which have their own internal structure, and the one pink element, CurrencyZone, is deprecated and should not be used.)

But a typical <Price> composite is not as complex as the diagram implies – in many circumstances, the only elements needed are the <PriceType>, <PriceStatus>, the price amount itself and possibly the accompanying tax detail, currency and – if the price applies in only some of the countries covered by 'for sale' sales rights – a <Territory>.

Price type is the mandatory indicator that specifies what *kind* of price – recommended retail, fixed retail, wholesale or agency (these four are the most common across most countries, though there are many other types). Note that in a strict sense, the <PriceType> is optional, but it may only be omitted if a <DefaultPriceType> is provided in the ONIX message header – and including defaults in the header are generally not good practice.

Price status is often omitted, and while it *is* optional, good practice is to include it so that the ONIX data recipient can tell whether the price is a firm offer or simply a provisional price. Provisional prices are only appropriate several weeks or months before publication. Obviously, a potential purchaser is more likely to place an order against a firm price than against a provisional price that might be altered later, so setting a firm price at least four months before publication should be a priority.

Price amount should be quoted to the right number of decimal places – *eg* 9.95 if the currency is Dollars, Pounds, Euros *etc*, but 995 (without any decimals) if the currency is Yen or South Korean Won). Note that even if it is conventional to use a comma as a decimal point (for example, 7,95 € would be common in France), only a period can be used in the ONIX XML – the recipient can *display* the price in the usual format, but in the <PriceAmount> it must be 7.95.

Note also that a Price amount that is 0.00, just zero or blank is not allowed. Free of charge products, or metadata provided before a particular price is set, must use <UnpricedItemType> instead of <PriceAmount>[1].

Tax details must only be included if the Price type indicates that the price amount includes tax. It is good practice to include all the elements within the <Tax> composite, even though in simple cases only one or other of <TaxRatePercent> or <TaxAmount> is mandatory. Note that <TaxAmount> is also mandatory when there are multiple repeats of the <Tax> composite (*ie* when different components of a product are taxed at different rates).

The **Currency code** – like the Price type – should be treated as mandatory (a default can be provided, but this is not a good practice).

The **Territory** can be very simple – if tax details are provided, then it is very likely that the price applies in only a single country, so the territory would consist of just a single <CountriesIncluded> tag.

So a simple price set in US Dollars, relevant to the US and all other countries outside the EU may look like this:

```
<Price>
    <PriceType>01</PriceType>                   <!-- RRP excluding tax -->
    <PriceStatus>02</PriceStatus>               <!-- firm -->
    <PriceAmount>7.95</PriceAmount>
    <CurrencyCode>USD</CurrencyCode>
    <Territory>
        <RegionsIncluded>WORLD</RegionsIncluded>  <!-- everywhere except the EU -->
        <CountriesExcluded>AT BE BG CY CZ DE DK EE ES FI FR GB GR HR HU IE IT LT
        LU LV MT NL PL PT RO SE SI SK</CountriesExcluded>   <!-- see footnote 2 -->
    </Territory>
</Price>
```

[1] The <Price> composite is repeatable, so *other* prices can be 'not free' in the usual way. It is also possible to use <UnpricedItemType> instead of <Price>, when the product is always free or when no prices have been set at all.
[2] This simple example ignores some subtleties in <Territory>. In particular, a country code such as FR indicates only metropolitan France (inc Corsica) and ignores overseas *departements* and territories (*eg* Guadeloupe, Réunion) which are part of the EU.

This book may be not for sale within the EU, or if it *is* for sale, this price should be accompanied by at least one (more likely several) further <Price> composites covering the various EU countries. A simple price for France, for example, may look like this:

```
<Price>
    <PriceType>04</PriceType>                    <!-- FRP including tax -->
    <PriceStatus>02</PriceStatus>                <!-- firm -->
    <PriceAmount>6.95</PriceAmount>
    <Tax>
        <TaxType>01</TaxType>
        <TaxRateCode>R</TaxRateCode>
        <TaxRatePercent>5.5</TaxRatePercent>     <!-- note 5.5% × 6.59 = 0.36 -->
        <TaxableAmount>6.59</TaxableAmount>      <!-- and 6.59 + 0.36  = 6.95 -->
        <TaxAmount>0.36</TaxAmount>
    </Tax>
    <CurrencyCode>EUR</CurrencyCode>
    <Territory>
        <CountriesIncluded>FR</CountriesIncluded>
    </Territory>
</Price>
```

Price changes

Upcoming price changes can be signalled to data aggregators, wholesalers and retailers in advance in ONIX by including both the current price and the future price. Dates can be attached to each to signal the last date the current price is valid, and the first date the future price is effective. The following could be added to the current price:

```
<PriceDate>
    <PriceDateRole>15</PriceDateRole>            <!-- price ends 28th Feb -->
    <Date>20180228</Date>
</PriceDate>
```

and this to the future price:

```
<PriceDate>
    <PriceDateRole>14</PriceDateRole>            <!-- price effective 1st Mar -->
    <Date>20180301</Date>
</PriceDate>
```

Of course, the new price applies to new stock ordered from the publisher or distributor from the effective date onwards. In many cases, there may still be existing stock in the supply chain that remains at the old price for some time after the price change.

Multiple tax rates

For some products with multiple components – for example (in the UK) many print and audio or print and e-book bundles – the different components of the product are taxed at different rates. While the print component carries reduced rate VAT (zero %), the audio or e-book component is taxed at the standard rate (20%). In this case, the publisher must decide the proportions of the price for each component [3], and calculate the tax appropriately.

In the ONIX, this requires two separate <Tax> composites. Here's a concrete example:

```
<PriceAmount>9.95</PriceAmount>              <!-- total recommended retail price -->
<Tax>
    <TaxType>01</TaxType>
    <TaxRateCode>S</TaxRateCode>
    <TaxRatePercent>20</TaxRatePercent>
    <TaxableAmount>5.85</TaxableAmount>          <!-- base price of component A -->
    <TaxAmount>1.17</TaxAmount>                  <!-- tax on price of component -->
</Tax>
<Tax>
    <TaxType>01</TaxType>
    <TaxRateCode>Z</TaxRateCode>
    <TaxRatePercent>0</TaxRatePercent>
    <TaxableAmount>2.93</TaxableAmount>>         <!-- base price of component A -->
    <TaxAmount>0.00</TaxAmount>                  <!-- tax on price of component -->
</Tax>
```

[3] For example by basing the proportions on the relative manufacturing costs, or linking the proportions to the retail prices of the components when sold individually – consult a tax advisor.

Notice that the <PriceAmount> is equal to the sum of the two taxable amounts and the two tax amounts – this must always be true.

Common errors with simple prices

The single most common issue is to see **<Tax> details included when <PriceType> indicates the price is exclusive of tax** (types 01, 03, 41 and so on). This cannot be interpreted with any reliability – is the tax intended to be *added* to the <PriceAmount> to give the final consumer price, or is the tax intended to be *subtracted* from the <PriceAmount> to give the actual exc-tax price? The <Tax> composite should *only* be used if the <PriceType> indicates an inc-tax price.

In simple cases, it is perfectly okay to do the opposite – to omit tax details when the price type is inc-tax. But this does imply the retailer needs to work out the tax details, and in complex cases involving different tax rates on different components within the product, it isn't possible. For example with a print/e-book or print/audio combination, the retailer would need to know the relative values of each component in order to calculate the relevant taxes. The ONIX should contain the full tax details (in two or more <Tax> composites, or provide the relevant details in <ProductClassification> (the latter is particularly useful in international trade).

The **use of a zero in <PriceAmount>** is also very common. This has *never* been the correct way to specify free-of-charge products, but the ONIX DTD which is often used for data validation cannot detect such errors – so many have mistakenly assumed that zero prices are okay. Recent ONIX schema files (XSD and RNG) expressly invalidate any zero price amount. Free of charge products, and those for which a price has not been set, should use <UnpricedItemType> instead of <PriceAmount>. And because all the other data elements can still be used – territory, dates and so on – it's simple to say that a product costing $7.99 in the US is free in Canada, or is free in the US after a certain date.

When tax is included, it is good practice to list all the elements within the <Tax> composite. However, this does mean there's some potential for mathematical errors. In essence, these should both be true:

```
        <TaxableAmount> + <TaxRatePercent> × <TaxableAmount> ÷ 100 = <TaxAmount>
        <TaxableAmount> + <TaxAmount> = <PriceAmount>
```

Throughout most of Europe, books are subject to Value Added Tax. The rate of tax in percent varies from country to country. And in most countries, there is a special 'reduced rate' VAT on physical books, rather than the 'standard rate' applied to most other goods and services (including e-books). A typical example would be Germany, where the standard rate is (currently) 19% with a reduced rate of 7%. Somewhat confusingly, the reduced rate in the UK is zero percent. This leads to the belief that 'books don't carry VAT'. This is incorrect – they carry VAT at 0%, which means that UK book prices should usually use <PriceType> 02 (RRP inc tax), <TaxableAmount> and <PriceAmount> are identical, and <TaxRatePercent> and <TaxAmount> are both zero.

Where there are multiple prices, it's vital that data recipients can deduce which *one* price is applicable, based on the various dimensions of the pricing cube. Specifying a price of 7.99 without a currency is obvious nonsense. A price without a territory can often cause problems [4] – particularly if it includes tax.

Do not confuse a firm price status (which is the opposite of a provisional price) with the concept of 'firm sale' (which is the opposite of 'sale or return').

Dates in ONIX are inclusive. So the end date for an old price and a start date for a new price are not the same date – they are one day apart.

EDItEUR
22/12/2017

[4] Technically, if the price territory is missing, then the price is applicable throughout the market territory – or if there is no market defined, throughout the <SalesRights> territories of types 01 and 02. *If this is what is intended*, then omitting <Territory> from <Price> is okay. But note, any tax details almost inevitably make a price specific to a single country, so <Tax> without <Territory> is almost always an error.