



ONIX for Books

Acknowledgement Format Specification

Release 3.0, January 2015

The *ONIX for Books Product Information Format Specification* (DOI: [10.4400/jzhw](https://doi.org/10.4400/jzhw)) and the associated controlled vocabularies or codelists (DOI: [10.4400/akjh](https://doi.org/10.4400/akjh)) were originally compiled for EDItEUR by David Martin (ISNI: [0000 0004 5905 7204](https://orcid.org/0000-0004-5905-7204)), with recent changes by Graham Bell (ISNI: [0000 0004 2756 6266](https://orcid.org/0000-0004-2756-6266)). Graham Bell also compiled the accompanying *Implementation and Best Practice Guide* (DOI: [10.4400/zuim](https://doi.org/10.4400/zuim)). The schemas (RNG, XSD and DTD) which define the communication format for ONIX for Books Product Information messages were developed by Francis Cave. This *Acknowledgement Specification* and its accompanying schemas were compiled by Graham Bell.

Copyright © 2014 EDItEUR. All rights reserved. DOI: [10.4400/ewea](https://doi.org/10.4400/ewea)

EDItEUR (<http://www.editeur.org>, ISNI: [0000 0004 4914 9201](https://orcid.org/0000-0004-4914-9201)) is the international group which coordinates the development and promotion of standards for electronic commerce in the book and serials sectors. EDItEUR originally developed the ONIX for Books standard jointly with Book Industry Study Group (BISG), New York, and Book Industry Communication (BIC), London. Continued development of ONIX is now managed by a network of National Groups and an International Steering Committee coordinated by EDItEUR. At the time of writing there are ONIX for Books national groups in more than fifteen countries. More information and contact details for many of the national groups (including BIC and BISG) will be found on the [ONIX Maintenance and Support page](#) on the EDItEUR website. Contact EDItEUR by [e-mail](#).

TERMS AND CONDITIONS OF USE OF THE ONIX for Books ACKNOWLEDGEMENT FORMAT AND ASSOCIATED XML DTD AND XML SCHEMAS

All ONIX standards and documentation – including this document – are copyright materials, made available free of charge for general use. A full license agreement (DOI: [10.4400/nwqi](https://doi.org/10.4400/nwqi)) that governs their use is available on the EDItEUR website. In particular, if you use any of the ONIX for Books Acknowledgement Format schemas (RNG, XSD or DTD) ('the schemas'), you will be deemed to have accepted these terms and conditions:

1. You agree that you will not add to, delete from, amend, or copy for use outside of the schemas, any part thereof except for strictly internal use in your own organization;
2. You agree that if your business would benefit from future additions or amendments to or extracts of the schemas for any purpose that is not strictly internal to your own organization, you will in the first instance notify EDItEUR and allow EDItEUR to review and comment on your proposed use, in the interest of securing an orderly development of the schemas for the benefit of other users.

If you do not accept these terms, you must not use the schemas.

This *Acknowledgement Format Specification* is normative.

If you find any errors or discrepancies, or you have a question about this document, please contact EDItEUR by e-mail to info@editeur.org. If you wish to translate this *Acknowledgement Specification* into another language, or otherwise modify it in any way, please note that it is copyright material and – while EDItEUR wishes it to be as widely available as possible – it should not be translated, abridged or modified without permission. Please contact EDItEUR before embarking on a translation or abridgement.

Table of Contents

1. Introduction	4
Document history	4
2. ONIX for Books Acknowledgement message	5
High-level structure and conformance	5
Use of XML attributes	8
Using ONIX schemas for validation	10
Character sets and special characters	11
Using XHTML, HTML or XML within text fields	14
3. ONIX for Books Acknowledgement Message header	15
4. ONIX for Books Acknowledgement Product record	26
5. Appendix	31
A.1 Data element summary	31
A.2 Sample messages	33
A.3 List of all XML tags	42
A.4 Using the Acknowledgement message with ONIX 2.1	44
A.5 Diagrams of XML Structure	45

1. Introduction

This *Acknowledgement Format Specification* ('the *Acknowledgement Specification*') is a new addition to the ONIX for Books metadata framework, intended to complement Release 3.0 of the main *ONIX for Books Product Information Format Specification*. This *Acknowledgement Specification* also includes complete sample acknowledgements, an overview of the XML data elements and complete lists of XML tags used in an Acknowledgement message. Although it is the first release of the *Acknowledgement Specification*, this document is termed 'Release 3.0' for consistency with the main the main ONIX message specification..

Other documentation for Release 3.0 comprises an *Introduction to ONIX for Books 3.0*, the *ONIX for Books Codelists* (Issue 12 was the first issue intended for use with ONIX 3.0), the *ONIX for Books Implementation and Best Practice Guide* (the *Guide*), together with a small number of *How to...* guides on particular aspects of ONIX 3.0 usage, with detailed examples.

Although this issue of the *Acknowledgement Specification* is complete in terms of its coverage of the structure and data elements in the ONIX 3.0 Acknowledgement message, it may be subject to further revision, not only to correct any errors that are found during the early stages of implementation, but also to add material that may make it easier to use. The codelists – which form an important part of the standard – are also regularly updated. The earliest issue of the codelists that is usable with this *Acknowledgement Specification* is Issue 28. Any revisions will be notified through the ONIX for Books implementation listserv. If you are not already a member, you may wish to sign up through the [EDItEUR website](#).

The Acknowledgement message has been piloted throughout 2014. The differences between the pilot version and this final release are limited to the introduction of the <ProductIdentifier> composite and the <NoProduct/> data element. The latter aligns with the introduction of an identical element in ONIX for Books 3.0.2 and means that either one or more instances of the <Product> composite or an instance of <NoProduct/> *must* be included in an Acknowledgement. There are also minor changes to the associated codelists (especially [List 224](#)).

If you believe that you have found an error, or you have a question about this document, please contact EDItEUR either through the listserv or by e-mail to info@editeur.org.

Document history

14 Jan 2015	Initial release (as PDF and HTML).
4 Feb 2015	Added appendix Using the Acknowledgement message with ONIX 2.1.
25 Apr 2016	Removed appendix A.6 Initial codelists.

2. ONIX for Books Acknowledgement message

High-level structure and conformance

An ONIX for Books Product Information Format message ('an original ONIX message') is a standard data format based on XML, used primarily to convey information about books, e-books and book-related products between computer systems used by publishers, various intermediaries and retailers. The Acknowledgement message specified in this document is intended to be returned by the data recipient to the data sender, to acknowledge receipt and processing or to list errors encountered with the sender's ONIX message.

An ONIX for Books Acknowledgement message can be regarded as having four component parts: the start of message, whose format and content is dictated by the XML standard; a message header block; the body of the message providing information about the processing of data describing a number of products; and the end of message.

The start and the end of message are described in Sections X.1 and X.2.

The message header carries a number of mandatory data elements specifying the sender, the addressee (the sender of the original ONIX message) and date of acknowledgement message, and optionally stating various message-level status information relating to the original ONIX message. For further details, see the ONIX for Books Acknowledgement Message header section of this document.

The body of an ONIX for Books Acknowledgement Message is often effectively empty, consisting only of the <NoProduct/> empty data element. Alternatively, it may consist of one or more Product records, with no theoretical limit on the number of records. Each Product record consists of an identifier for the record (the <RecordReference> taken from the original ONIX message), plus record-level status information relating to the records in the original ONIX message. The content and format of the Product record are detailed in the ONIX for Books Acknowledgement Product record section of this document.

Some data elements within an ONIX for Books Acknowledgement Message take their content from code lists, controlled vocabularies, to ensure common understanding of the data where message creator and recipient need not be in direct contact, and where they may operate in quite different markets. These code lists also form an integral part of the specification of an ONIX for Books Acknowledgement message. Code lists are revised from time to time to add new codes. Old codes are never deleted, though they may be deprecated. The earliest release of the code lists that may be used with this specification is Issue 28.

An overriding requirement is that an ONIX for Books Acknowledgement Message must conform to the XML standard, *ie* it must be well-formed XML. It is also a requirement that Acknowledgement messages are valid according to the associated RNG and XSD schemas (which are equivalent).

As with previous ONIX for Books message specifications, the Acknowledgement message is also supported by formal definition in DTD format. The RNG and XSD formats differ from the DTD format in that they make extensive use of formal specifications of the code lists, whereas the DTD format makes almost no use of the code lists. Validation using the DTD is not sufficient and not recommended, as data element code values cannot be validated by this method. Implementors are free to choose which of the ONIX for Books schemas to use in validating the ONIX messages that they create or receive, but if the DTD format is chosen, implementors will need to find other methods for checking that all code values are valid.

Each schema is available in two separate 'flavors' with differing but equivalent XML markup, and any message must choose one or other markup flavor – either *Reference names* or *Short tags*. The former means that message filesizes are larger, but messages are more easily human-readable. Flavors must not be mixed within any one Acknowledgement message, and should match the flavor used in the original ONIX message to which the Acknowledgement is a response. Implementors may choose to implement either or both markup flavors, and best practice guidelines within any ONIX community may guide that choice.

All implementors are expected to check that *all* requirements of the ONIX for Books Acknowledgement specification are met, irrespective of whether or not these requirements are formally specified and enforced by any of the schemas. In particular, the specification defines requirements such as presence or absence of certain XML elements based on data values elsewhere in the message. These 'business rule' requirements cannot be specified in RNG, XSD or DTD schema languages. They may be encoded and enforced in an advanced schema language (eg Schematron) in the future.

Your attention is drawn to the terms and conditions of use which appear in the ONIX for Books schemas themselves, and on the [copyright](#) page of these Guidelines.

Notes	The DTD format is defined by the <i>XML Standard: W3C Recommendation Extensible Markup Language (XML) 1.0 (Fourth Edition)</i> – see http://www.w3.org/TR/2006/REC-xml-20060816/ for more details.
	The XML Schema Definition (XSD) format is defined by <i>W3C Recommendation XML Schema Part 1: Structures (Second Edition)</i> – see http://www.w3.org/TR/2004/REC-xmlschema-1-20041028/ for more details.
	The RELAX NG (RNG) format is defined by <i>ISO/IEC 19757-2:2008</i> , published by ISO, Geneva.

X.1 Start of message

The start of an ONIX for Books Acknowledgement message must consist of, as a minimum, two lines of XML as shown:

Example	<i>using Reference names</i>
	<pre><?xml version="1.0" encoding="UTF-8"?> <ONIXMessageAcknowledgement release="3.0" Upper case 'M' and 'A' xmlns="http://ns.editeur.org/onix/3.0/ acknowledgement/reference"></pre>
	<i>using Short tags</i>
	<pre><?xml version="1.0" encoding="UTF-8"?> <ONIXmessageacknowledgement release="3.0" Lower case 'm' and 'a' xmlns="http://ns.editeur.org/onix/3.0/ acknowledgement/short"></pre>

The character encoding in the XML declaration is optional, but strongly recommended. Further details of this are given in the [Character sets and special characters](#) section below.

The *xmlns* namespace attribute is also optional, but strongly recommended.

For the purposes of validation of an ONIX message against one of the schemas it may be necessary to insert some additional information on the second line, to include an explicit reference to the file location of the schema against which to validate the message. The precise XML that needs to be inserted will depend upon the schema format (XSD, RNG or DTD) and the tools being used for validation. Further details of this are given in the [Using Release 3.0 schemas for validation](#) section below.

X.2 End of message

The end of message 'trailer' must consist of a single line as shown:

Example	<i>using Reference names</i>
	<pre></ONIXMessageAcknowledgement></pre> Upper case 'M' and 'A'
	<i>using Short tags</i>
	<pre></ONIXmessageacknowledgement></pre> Lower case 'm' and 'a'

X.3 Layout of a complete message

In summary, the layout of a typical ONIX for Books Acknowledgement Message is like this:

Example	<i>using Reference names</i>	
	<pre><?xml version="1.0" encoding="UTF-8"?> <ONIXMessageAcknowledgement release="3.0" xmlns="http://ns.editeur.org/onix/3.0/ acknowledgement/reference"></pre>	Start of message – encoding and namespace are optional but recommended
	<pre><Header> <!-- message header data elements --> <!-- status information for message --> </Header></pre>	Header
	<pre><Product> <!-- record reference for product 1 --> <!-- status information for product 1 --> </Product></pre>	
	<pre><Product> <!-- record reference for product 2 --> <!-- status information for product 2 --></pre>	Body
	<pre></Product> <!-- further product records... --> <Product> <!-- record reference for product n --> <!-- status information for product n --> </Product></pre>	
	<pre></ONIXMessageAcknowledgement></pre>	End of message
	<i>using Short tags</i>	
	<pre><?xml version="1.0" encoding="UTF-8"?> <ONIXmessageacknowledgement release="3.0" xmlns="http://ns.editeur.org/onix/3.0/ acknowledgement/short"></pre>	Start of message – encoding and namespace are optional but recommended
	<pre><header> <!-- message header data elements --> <!-- status information for message --> </header></pre>	Header
	<pre><product> <!-- record reference for product 1 --> <!-- status information for product 1 --> </product></pre>	
	<pre><product> <!-- record reference for product 2 --> <!-- status information for product 2 --></pre>	Body
	<pre></product> <!-- further product records... --> <product> <!-- record reference for product n --> <!-- status information for product n --> </product></pre>	
	<pre></ONIXmessageacknowledgement></pre>	End of message

Note that ONIX messages can contain XML comments, introduced with '`<!--`' and terminated with '`-->`'. Comments may be helpful during development, when data may have to be checked 'by eye'. However, they are of no significant benefit in production, as they will be ignored by automated XML processing systems.

X.4 Empty XML elements

There is just one element in the ONIX for Books Acknowledgement format which is defined as an empty element in XML. All other elements are defined as carrying data content, and must not be sent as empty elements. If an element is mandatory, its content must be supplied, or the message will be invalid. If an element is optional, and there is no content for it, it must be omitted entirely. These rules are enforced in the RNG and XSD schemas, but cannot be enforced by the DTD.

The exceptional empty element is `<NoProduct/>` (or `<x507/>`). The sole valid use of this is within a message that contains no Product record-level status information.

Use of XML attributes

In all ONIX applications, a number of XML attributes may be used where applicable to carry information about the content of an associated element. The view which has been taken in the development of ONIX is that it is undesirable to use XML attributes to carry portions of the actual data content of the ONIX message. However, it is appropriate to use them to carry information which qualifies the data itself and its representation – metadata about metadata, as it were.

A number of general attributes are defined and may be used with *any* ONIX element, and they are not noted individually for each data element in the specification. However, while these are defined and valid for all elements within the Acknowledgement message, there is almost certainly no need to use them:

- `datestamp`
- `sourcename`
- `sourcetype`

Three further attributes may be used with a limited selection of data elements, as noted individually for each element in the specification:

- `dateformat`
- `language`
- `release`

Attributes are carried within an XML start tag. The attribute name is lower case, separated from the name of the element by a space, and the attribute value is placed in double quotes. If there are two or more attributes in a single tag, they too are separated by a space. Multiple attributes in a single tag may occur in any order.

The built-in XML attributes `xml:lang` and `xml:space` are not used in ONIX for Books Acknowledgement messages: these attributes are available by default in all XML applications, and cannot be prohibited technically, but should never be included in ONIX acknowledgements.

X.5 Datestamp attribute

Enables any data element or composite to carry the date or date and time when it was last updated or confirmed as correct, where it is significantly different from the overall `<AcknowledgementSentDateTime>` of the Acknowledgement message. If not supplied, there is no default value. If used on a composite, the *datestamp* indicates the most recent date when any individual data element within the composite was updated or confirmed as correct.

Format	<p>Permitted formats, where 'T' and 'Z' represent themselves (<i>ie</i> the letters T and Z), and where the symbol '±' represents either '+' or '-' to indicate a timezone offset from UTC.</p> <table> <tr> <td>YYYYMMDD</td><td>Date only</td></tr> <tr> <td>YYYYMMDDThhmm</td><td>Date and time (local time of sender)</td></tr> <tr> <td>YYYYMMDDThhmmZ</td><td>Universal time (UTC) †</td></tr> <tr> <td>YYYYMMDDThhmm±hhmm</td><td>With time zone offset from UTC †</td></tr> <tr> <td>YYYYMMDDThhmmss</td><td>Date and time (with seconds)</td></tr> <tr> <td>YYYYMMDDThhmmssZ</td><td>Universal time (with seconds)</td></tr> <tr> <td>YYYYMMDDThhmmss±hhmm</td><td>With time zone offset from UTC (with seconds)</td></tr> </table> <p>† indicates the <i>preferred</i> formats</p>	YYYYMMDD	Date only	YYYYMMDDThhmm	Date and time (local time of sender)	YYYYMMDDThhmmZ	Universal time (UTC) †	YYYYMMDDThhmm±hhmm	With time zone offset from UTC †	YYYYMMDDThhmmss	Date and time (with seconds)	YYYYMMDDThhmmssZ	Universal time (with seconds)	YYYYMMDDThhmmss±hhmm	With time zone offset from UTC (with seconds)
YYYYMMDD	Date only														
YYYYMMDDThhmm	Date and time (local time of sender)														
YYYYMMDDThhmmZ	Universal time (UTC) †														
YYYYMMDDThhmm±hhmm	With time zone offset from UTC †														
YYYYMMDDThhmmss	Date and time (with seconds)														
YYYYMMDDThhmmssZ	Universal time (with seconds)														
YYYYMMDDThhmmss±hhmm	With time zone offset from UTC (with seconds)														
Attribute name	<i>datestamp</i>														
Example	<code><MessageStatus datestamp="20100621">00</MessageStatus></code>														
Notes	The calendar date must use the Gregorian calendar, even if other dates within the acknowledgement message use a different calendar. For all practical purposes, UTC is the same as GMT.														

X.6 Sourcename attribute

Enables a data element or composite to carry the name of the source or authority for the data content. This is particularly useful when an acknowledgement is generated by a third-party or third-party software application not under the direct control of the sender of the acknowledgement. If not supplied, the data authority should be assumed to be the sender of the Acknowledgement message.

Format	Variable-length text, suggested maximum 50 characters
Attribute name	<i>sourcename</i>
Example	<code><a498 sourcename="XYZ Livres SA">01</a498></code> (XYZ is source of information)

X.7 Sourcetype attribute

Enables a data element or composite to carry a code indicating the type of source or authority for the data content. This is particularly useful when an acknowledgement is generated by a third-party or third-party software application not under the direct control of the sender of the acknowledgement. If not supplied, the data authority should be assumed to be the sender of the Acknowledgement message.

Format	Fixed-length, two digits
Code list	List 3
Attribute name	<i>sourcetype</i>
Example	<code><a498 sourcetype="11">01</a498></code> (Source of information is a retail bookseller)

X.8 Dateformat attribute

Used with the <Date> element to specify the format of the date. Each date element on which this attribute may be used specifies a default *dateformat* if the attribute is not supplied – for most date elements, this is format '00', YYYYMMDD. Note that the *dateformat* attribute is *not* used with <SentDateTime> and <AcknowledgementSentDateTime>.

Format	Fixed length, two digits
Code list	List 55
Attribute name	<i>dateformat</i>
Example	<code><Date dateformat="13">20100412T1425Z</Date></code> (format is YYYYMMDDThhmm with optional timezone information, date and time is 2:25pm UTC on 12 Apr 2010)

X.9 Language attribute

Enables the language of any text element to be specified when it is not the expected default language of the Acknowledgement message. The default language of the textual metadata is generally set by agreement between sender and recipient, and is separate from (though often identical to) the default language of the text used within the products described within the original ONIX message.

Many data elements that carry the *language* attribute are repeatable in order to allow parallel text to be provided in multiple languages. Data recipients able to support only a single language should select the repeat that carries the most appropriate *language* attribute.

Generally, a language also implies a particular script. (There are a very few languages that are commonly written in more than one script.)

Format	Fixed-length, three lower-case letters. Note that ISO 639 specifies that these codes should always be in lower-case
Code list	ISO 639-2/B List 74
Attribute name	<i>language</i>
Example	<code><AcknowledgementNote language="fre">Réponse retardée en raison de jour férié</AcknowledgementNote></code> (Text of note is in French)

X.10 Release attribute

Identifies the release of the ONIX format standard to which the Acknowledgement message conforms. Used only in the top-level element `<ONIXMessageAcknowledgement>` (short tag `<ONIXmessageacknowledgement>`), and is mandatory. The value will change with each new release, so that all messages will show explicitly the release to which they are intended to conform.

Format	must be "3.0" for this release
Attribute name	<i>release</i>
Example	<code><ONIXmessageacknowledgement release="3.0"></code>

Using ONIX schemas for validation

The main use of the ONIX for Books schemas is to automate checks on the validity of an ONIX message: does it use the right tags and the right code values in the right places? The schemas are available in DTD, XSD and RNG formats. The XSD and RNG forms are both much more expressive than the DTD form, enabling validation of code values, dates, quantities and link addresses where appropriate. The DTD is provided only for use with legacy validation tools.

For validation of ONIX messages against a Release 3.0 schema EDItEUR recommends that implementors adopt the following approach:

1. Decide which form of schema to use for validation purposes, XSD or RNG. This is largely a matter of personal preference, as the two are equivalent. The DTD format is not recommended;
2. Download from the EDItEUR website a copy of the selected schema and install this copy in an accessible location on a local server (eg on a corporate intranet or on a stand-alone PC);
3. Configure the software tools used for validation purposes to refer to the local copy of the schema.

When using an XSD or RNG schema to validate ONIX acknowledgement messages, a namespace declaration (the *xmlns* attribute) is required in the top-level `<ONIXMessageAcknowledgement>` start tag (see the [Start of message](#) section above). If you are using Reference names, the namespace URI must be "http://ns.editeur.org/onix/acknowledgement/3.0/reference". If you are using Short tags, it must be "http://ns.editeur.org/onix/acknowledgement/3.0/short". These namespace URIs are the same as are specified within the corresponding ONIX XSD and RNG schemas. Note that these URIs do *not* correspond to an actual Web address that is reachable by a browser. They are simply a device for creating an unambiguous reference to the authority for the defined ONIX namespaces.

Depending on the software tools used for validation, other changes to this start tag may be necessary. For example, with some validation software it may be necessary to identify within the ONIX message the location of the .xsd or .rng file to be used for validation. This location would normally be on a local disc or internal network. The *xsi:schemaLocation* attribute links the ONIX namespace URI to the .xsd location, and this location *would* correspond to a real address reachable by a browser.

Example	locating the XSD schema on an internal network
	<i>using Reference names</i>
	<pre><ONIXMessageAcknowledgement release="3.0" xmlns="http://ns.editeur.org/onix/acknowledgement/3.0/reference" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://ns.editeur.org/onix/acknowledgement/ 3.0/reference http://intranet/onix/reply/ ONIX_BookProductAcknowledgement_3.0_reference.xsd"></pre>
	<i>using Short tags</i>
	<pre><ONIXmessageacknowledgement release="3.0" xmlns="http://ns.editeur.org/onix/acknowledgement/3.0/short" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://ns.editeur.org/onix/acknowledgement/ 3.0/short http://intranet/onix/reply/ ONIX_BookProductAcknowledgement_3.0_short.xsd"></pre>

If using a DTD to validate an ONIX acknowledgement message, a suitable DOCTYPE declaration must be added.

Example	locating the DTD schema on a local disc
	<i>using Reference names</i>
	<pre><!DOCTYPE ONIXMessageAcknowledgement SYSTEM "./DTDs/ONIX_BookProductAcknowledgement_3.0_reference.dtd"> <ONIXMessageAcknowledgement release="3.0" xmlns="http://ns.editeur.org/onix/acknowledgement/3.0/reference"></pre>
	<i>using Short tags</i>
	<pre><!DOCTYPE ONIXmessageacknowledgement SYSTEM "./DTDs/ONIX_BookProductAcknowledgement_3.0_short.dtd"> <ONIXmessageacknowledgement release="3.0" xmlns="http://ns.editeur.org/onix/acknowledgement/3.0/short"></pre>

Any link to a local file copy of the relevant schema that has been added to an ONIX message for validation purposes should be removed prior to making the ONIX message available to supply chain partners. The message recipient will make their own arrangements for validation of incoming messages to suit their own internal systems.

Character sets and special characters

In principle, XML data files may include any [Unicode](#) character, and the default encoding is either UTF-8 or UTF-16. In practice, unless the XML declaration at the beginning of the ONIX Acknowledgement message includes an explicit 'encoding declaration', you should restrict the character set to the printable characters of ASCII (*ie* those characters whose Unicode numbers fall between 32 and 126 inclusive). This limited character set is unlikely to be adequate for anything except the most basic data in English, so it is expected that most – and outside of the English language markets, all – ONIX messages will include an encoding declaration or other special coding of non-ASCII characters described in the [Extended character sets and encoding declarations](#) section below.

For reference, here is a list of the basic character set for which no special coding is necessary, all of which can be found on a standard English computer keyboard:

- space character
- capital letters A–Z
- lower-case letters a–z
- digits 0–9
- punctuation ! " ' , - . : ; ?
- brackets () [] { }
- symbols # \$ % * + / = > \ @ _ ` | ~

The tab, line feed and carriage return characters are also allowed, but most XML software will treat them as spaces.

This set deliberately does *not* include the characters ‘&’ and ‘<’. These characters have special significance in all XML applications, and therefore cannot be used as text characters in any ONIX data elements. If you need to include either of these characters within a data element (for example in the name of an organization such as ‘AT&T’ that by convention uses ‘&’ rather than ‘and’), you must use the XML named entity reference form of expression in place of the ‘&’ or ‘<’:

Example	<i>entity</i>	<i>represents</i>
	&	&
	<	<

You may also use named entity references in place of the following characters:

Example	<i>entity</i>	<i>represents</i>
	"	"
	'	'
	>	>

ONIX for Books Acknowledgement markup (XML element and composite names, attribute names, and code values drawn from the ONIX codelists) is limited to this basic set of ASCII characters, plus the literal ‘&’ and ‘<’ symbols.

Note that the currency symbols ‘£’, ‘¥’, and ‘€’ and special symbols such as en and em rules, ellipses and bullets are not in the basic set, nor are there standard named entity references that may be used in their stead. Note also that some office applications insert so-called ‘smart’ apostrophes and quotation marks (single or double, open or closed), and these too are not in the basic set.

If your Acknowledgement message contains no characters beyond this basic set, then no special coding is necessary. However, without special precautions, if your Acknowledgement message contains even a single character beyond this basic set, there is a significant risk that a recipient will be unable to process your Acknowledgement because it contains invalid text. This is because XML without a character encoding declaration will be assumed to use the UTF-8 encoding – and even one stray extended character from another character encoding is likely to cause a UTF-8 text error.

X.13 Extended character sets and encoding declarations

If the basic ASCII character set is not adequate – and in most cases it will not be – there are two ways to incorporate non-ASCII characters into your ONIX data: either use Unicode *numerical character references*, or use an *encoding declaration*.

Numerical character references take the form defined in Section 4.1 of the XML 1.0 Recommendation. For example, the character ‘ž’ (z háček or z caron, used in Czech and some other languages) can be expressed as ‘ž’ or ‘ž’ where 382 and 17e are the Unicode character numbers in decimal and hexadecimal (base 16 numbers) respectively. In principle, any Unicode character can be included in ONIX data this way.

Example	<i>using a decimal character reference – è is character 232</i>
	<code><MessageStatusNote>Succ&#232;s complet</MessageStatusNote></code> (Succès complet)
	<i>using a hexadecimal character reference - è is character e8</i>
	<code><MessageStatusNote>Succ&#xe8;s complet</MessageStatusNote></code> (Succès complet)
Notes	Character references between and Ÿ (or between and Ÿ in hexadecimal) should never be used. It is a common error to use these numbers (eg ž or ž for 'ž'), since they are valid character numbers in the Windows-1252 character set, and they <i>appear</i> to work in web browsers, but they are strongly discouraged in all XML applications.

If you have a relatively small number of non-ASCII characters in your ONIX data, this approach using numerical character references might well be suitable. But beware: as above, in the absence of an encoding declaration, even a single character that is not in the basic ASCII set and not encoded into a numerical character reference can cause a text validation error.

If your Acknowledgement message is not primarily in English, or if you make extensive use of extended characters even in English text, then it is likely to be simpler to use a message-wide encoding declaration. This must be included in the XML declaration at the beginning of the ONIX message.

Example	<code><?xml version="1.0" encoding="ISO-8859-1"?></code> (message uses ISO Latin-1 character set and encoding)
	<code><?xml version="1.0" encoding="windows-1252"?></code> (message uses Windows-1252 character set and encoding)
	<code><?xml version="1.0" encoding="ISO-8859-15"?></code> (message uses ISO Latin-9 character set and encoding)

Declaring the ISO Latin-1 character set with ISO-8859-1 encoding allows characters and diacritics used in most Western European languages to be included in the ONIX message without any special encoding, as well as some other extended characters. However, note that Latin-1 does not include a '€' symbol (Latin-9 – also known as ISO-8859-15 – is an alternative character set that does), nor does it contain smart quotes (“ ” and ‘ ’), en and em dashes or the ellipsis character. Local circumstances may favor the use of other character sets and encodings, such as any of Parts 2 through 16 of ISO-8859, the common Windows-1252 (which is likely to be used by many older Windows-based applications in North America and Western Europe, and which does include smart quotes, en and em dashes and the ellipsis), or various Asian language encodings such as Shift-JIS.

Alternatively, a Unicode encoding like UTF-8 or -16 can include characters from any and all of these character sets. Despite the fact that UTF-8, UTF-16BE or UTF-16LE are the 'default' character encodings used in all XML applications, it is strongly recommended that they are declared explicitly when it is used in an ONIX for Books message. A byte order mark should not be included in UTF encodings. EDItEUR recommends the use of UTF-8 encoding in ONIX Acknowledgements that will be exchanged outside a particular national market.

Example	<code><?xml version="1.0" encoding="UTF-8"?></code> (UTF-8 encoding, Unicode character set)
---------	---

Using a suitable message-wide encoding means that no special encoding needs to be used for individual non-ASCII characters, making it straightforward to include metadata in any language. This applies equally to languages that use a Latin-based script, such as French above, and languages that use other scripts. Note that for Arabic and other scripts normally rendered right-to-left, the order of characters in the message follows the logical reading order for each script.

Whichever encoding is declared, ONIX implementors need to ensure the character set and encoding are controlled throughout the process of creating and sending the ONIX data, and receiving and processing Acknowledgements, so that data that is declared as being, say, ISO-8859-1, really is encoded as ISO-8859-1 and does not include any characters that are not present in the Latin-1 character set: simple cutting and pasting text from a variety of sources, for example, is likely to introduce character encoding inconsistencies (a character would be understood by the recipient as

some different character entirely) and may even result in completely invalid text. Software used to create XML with a rich character set needs careful configuration.

The two methods of incorporating non-ASCII characters can be combined: if your Acknowledgement uses the ISO Latin-1 character set and ISO-8859-1 encoding, it can include characters such as é, ø or £ ‘natively’. And although Latin-1 does not include the ‘ž’ character, it can still be included in your data using the numerical character reference ‘ž’ (or the hexadecimal equivalent ‘ž’). In contrast, if you use UTF-8, no numerical character references are necessary – effectively all characters can be included ‘natively’.

Implementors should note that it is not a requirement that recipients of ONIX messages should be able to handle correctly any character encodings other than ASCII, but supporting at least UTF-8 as an encoding is very strongly recommended. Equally, it is not a requirement for recipients to support any characters beyond the basic ASCII set, but in practice it is expected that recipients support the characters used in the commonly-used languages and scripts in their area of operation. For further technical guidance on character encodings in XML see Section 4.3.3 of the XML 1.0 Recommendation.

In early releases of the *ONIX for Books Product Information Format Specification* – up to release 2.1.4 – it was recommended that named HTML entity references be used in preference to numerical character references, and these were supported by the inclusion of special named entities such as ‘ö’, ‘…’ or ‘–’ in the ONIX for Books DTD. This recommendation no longer applies, and named entity references other than the five mandatory entities required for XML (&, <, >, ' and ") are supported in neither the main ONIX 3.0 message nor the Acknowledgement message. Special characters that are not available in the character set and encoding used for the Acknowledgement message may only be represented by numerical character references (*ie* an ellipsis may be included as ‘…’ or ‘…’, where in ONIX for Books release 2.1 for example, ‘…’ could have been used).

For textual data in an East Asian writing system which uses text glosses (for example Chinese or Japanese), Unicode’s interlinear annotation delimiters (numerical character references ࿹–࿻) may be used to delimit the gloss.

Example	<p><i>data in Japanese kanji script, including hiragana gloss</i></p> <pre><RecordStatusNote>これは&#xff9;村上春樹&#xfffa;むらかみはるき&#xffb;すべきですか？</RecordStatusNote> (Should this be Haruki Murakami?)</pre>
Notes	<p>This should typically be displayed as <small>村上春樹</small>これは村上春樹すべきですか？ Note that the interlinear annotation delimiters do not need to be included as numerical character references – they can be ‘native’ characters if the character encoding allows, but numerical references are used here for clarity because the characters are otherwise invisible. If a recipient application cannot properly process the interlinear annotations, then &#xff9; should be ignored, and &#xfffa; should be replaced with ‘ (’ and &#xffb; with ‘) ’ for display purposes.</p>

Using XHTML, HTML or XML within text fields

XHTML, HTML or XML mark-up cannot be used within any textual data elements in the ONIX for Books Acknowledgement message.

3. ONIX for Books Acknowledgement Message header

Header composite

A group of data elements which together constitute an acknowledgement message header. Mandatory in any ONIX for Books Acknowledgement message, and non-repeating.

The header carries information to identify the original ONIX message to which the acknowledgement is a response, as well as information identifying the acknowledgement and its sender. It also carries status information relating to the handling of the original ONIX message by the sender of the acknowledgement, and may carry a summary of the status of the Product records in the original ONIX message.

Reference name	<Header>
Short tag	<header>
Cardinality	1

Sender composite

A group of data elements which together specify the sender of an Acknowledgement message – *ie* the addressee or recipient of the original ONIX message. Mandatory in an ONIX for Books Acknowledgement message, and non-repeating.

Sender details may sometimes be drawn directly from Addressee details in the original ONIX message.

Reference name	<Sender>
Short tag	<sender>
Cardinality	1

Sender identifier composite

A group of data elements which together define an identifier of the sender of the acknowledgement. The composite is optional, and repeatable if more than one identifier of different types is sent, but *either* a <SenderName> *or* a <SenderIdentifier> *must* be included.

Reference name	<SenderIdentifier>
Short tag	<senderidentifier>
Cardinality	0...n

H.1 Sender identifier type

An ONIX code identifying a scheme from which an identifier in the <IDValue> element is taken. Mandatory in each occurrence of the <SenderIdentifier> composite, and non-repeating.

Format	Fixed length, two digits
Code list	List 44
Reference name	<SenderIDType>
Short tag	<m379>
Cardinality	1
Example	<m379>01</m379>

H.2 Identifier type name

A name which identifies a proprietary identifier scheme (*ie* a scheme which is not a standard and for which there is no individual ID type code). Must be included when, and only when, the code in the <SenderIDType> element indicates a proprietary scheme. Optional and non-repeating.

Format	Variable-length text, suggested maximum 50 characters
Reference name	<IDTypeName>
Short tag	<b233>
Cardinality	0...1
Attributes	<i>language</i>
Example	

H.3 Identifier value

An identifier of the type specified in the <SenderIDType> element. Mandatory in each occurrence of the <SenderIdentifier> composite, and non-repeating.

Format	According to the identifier type specified in <SenderIDType>
Reference name	<IDValue>
Short tag	<b244>
Cardinality	1
Example	

End of sender identifier composite

H.4 Sender name

The name of the acknowledgement sender organization, which should always be stated in a standard form agreed with the addressee. Optional and non-repeating, but *either* a <SenderName> element or a <SenderIdentifier> composite *must* be included.

Format	Variable-length text, suggested maximum 50 characters
Reference name	<SenderName>
Short tag	<x298>
Cardinality	0...1
Example	<SenderName>Hachette Livre</SenderName>

H.5 Sender contact

Free text giving the name, department, phone number, *etc* for a contact person in the sender organization who is responsible for the content of the message. Optional and non-repeating.

Format	Variable-length text, suggested maximum 300 characters
Reference name	<ContactName>
Short tag	<x299>
Cardinality	0...1
Example	<x299>Jacqueline Brown, tel. +44 20 7946 0921</x299>

H.6 Sender contact e-mail address

A text field giving the e-mail address for a contact person in the sender organization who is responsible for the content of the message. Optional and non-repeating.

Format	Variable-length text, suggested maximum 100 characters
Reference name	<EmailAddress>
Short tag	<j272>
Cardinality	0...1
Example	<j272>jacqueline.brown@hachettelivre.fr</j272>

End of sender composite

Addressee composite

A group of data elements which together specify the addressee of an ONIX for Books Acknowledgement message (which would be the sender of the original ONIX message). Optional in an ONIX for Books Acknowledgement message, and non-repeatable.

Reference name	<Addressee>
Short tag	<addressee>
Cardinality	0...1

Addressee identifier composite

A group of data elements which together define an identifier of the addressee. The composite is optional, and repeatable if more than one identifier of different types is sent, but *either* an <AddresseeName> or an <AddresseeIdentifier> *must* be included.

Reference name	<AddresseeIdentifier>
Short tag	<addresseeidentifier>
Cardinality	0...n

H.7 Addressee identifier type

An ONIX code identifying a scheme from which an identifier in the <IDValue> element is taken. Mandatory in each occurrence of the <AddresseeIdentifier> composite, and non-repeating.

Format	Fixed length, two digits
Code list	List 44
Reference name	<AddresseeIDType>
Short tag	<m380>
Cardinality	1
Example	<AddresseeIDType>01</AddresseeIDType> (ID is proprietary)

H.8 Identifier type name

A name which identifies a proprietary identifier scheme (*ie* a scheme which is not a standard and for which there is no individual ID type code). Must be used when, and only when, the code in the <AddresseeIDType> element indicates a proprietary scheme. Optional and non-repeating.

Format	Variable length text, suggested maximum 50 characters
Reference name	<IDTypeName>
Short tag	<b233>
Cardinality	0...1
Attributes	<i>language</i>
Example	<b233>BigRetailer Supplier ID</b233>

H.9 Identifier value

An identifier of the type specified in the <AddresseeIDType> element. Mandatory in each occurrence of the <AddresseeIdentifier> composite, and non-repeating.

Format	According to the identifier type specified in <AddresseeIDType>
Reference name	<IDValue>
Short tag	<b244>
Cardinality	1
Example	

End of addressee identifier composite

H.10 Addressee name

The name of the addressee organization, which should always be stated in a standard form agreed with the addressee. Optional and non-repeating, but *either* a <AddresseeName> element *or* a <AddresseeIdentifier> composite *must* be included.

Format	Variable-length text, suggested maximum 50 characters
Reference name	<AddresseeName>
Short tag	<x300>
Cardinality	0...1
Example	<x300>BiblioAggregator Ltd</x300>

H.11 Addressee contact

Free text giving the name, department, phone number *etc* for a contact person in the addressee organization to whom the message is to be directed. Optional and non-repeating.

Format	Variable-length text, suggested maximum 300 characters
Reference name	<ContactName>
Short tag	<x299>
Cardinality	0...1
Example	<ContactName>Mel Carter, tel. +44 1632 457890</ContactName>

H.12 Addressee contact e-mail address

A text field giving the e-mail address for a contact person in the addressee organization. Optional and non-repeating.

Format	Variable-length text, suggested maximum 100 characters
Reference name	<EmailAddress>
Short tag	<j272>
Cardinality	0...1
Example	<j272>carterm@aggregator.co.uk</j272>

End of addressee composite

H.13 Message sequence number

A monotonic sequence number of the message exchange in a series sent between trading partners, to enable the addressee to check against gaps and duplicates. This is the message number of the original ONIX message to which the Acknowledgement message is a response. Optional and non-repeating.

Format	Variable length positive integer
Reference name	<MessageNumber>
Short tag	<m180>
Cardinality	0...1
Example	<m180>1234</m180>

H.14 Message repeat number

A number which distinguishes any repeat transmissions of a particular message. If this element is used, the original is numbered 1 and repeats are numbered 2, 3 *etc.* This is the message repeat number of the original ONIX message to which the Acknowledgement is a response. Optional and non-repeating.

Format	Variable length positive integer
Reference name	<MessageRepeat>
Short tag	<m181>
Cardinality	0...1
Example	<m181>2</m181>

H.15 Message creation date/time

The date on which the original ONIX message (to which the Acknowledgement is a response) was sent. This must be taken exactly from the <SentDateTime> of the original message. The date (and optional time) format is based on ISO 8601. Mandatory and non-repeating.

Format	Permitted formats, where 'T' and 'Z' represent themselves (<i>ie</i> the letters T and Z), and where the symbol '±' represents either '+' or '-' to indicate a timezone offset from UTC. YYYYMMDD YYYYMMDDThhmm YYYYMMDDThhmmZ YYYYMMDDThhmm±hhmm YYYYMMDDThhmmss YYYYMMDDThhmmssZ YYYYMMDDThhmmss±hhmm Date only Date and time (local time of sender) Universal time (UTC) † With time zone offset from UTC † Date and time (with seconds) Universal time (with seconds) With time zone offset from UTC (with seconds) † indicates the <i>preferred</i> formats
Reference name	<SentDateTime>
Short tag	<x307>
Cardinality	1
Example	<x307>20100522T1230Z</x307> (12.30pm UTC, 22 May 2010)
Notes	For all practical purposes, UTC is the same as GMT.

H.16 Acknowledgement sequence number

A monotonic sequence number of the Acknowledgement messages returned in response to a single original ONIX message. There can be multiple distinct acknowledgements sent in response to a single ONIX message. If this element is used, the first acknowledgement is numbered 1 and subsequent (but distinct) acknowledgements are numbered 2, 3 *etc.* Optional and non-repeating.

Format	Variable-length positive integer
Reference name	<AcknowledgementNumber>
Short tag	<m485>
Cardinality	0...1
Example	<m485>1</m485>

H.17 Acknowledgement repeat number

A number which distinguishes any repeat transmissions of a particular acknowledgement. If this element is used, the original is numbered 1 and further transmissions of the same acknowledgement are numbered 2, 3 *etc.* Optional and non-repeating.

Format	Variable-length positive integer
Reference name	<AcknowledgementRepeat>
Short tag	<m486>
Cardinality	0...1
Example	<AcknowledgementRepeat>2</AcknowledgementRepeat>

H.18 Acknowledgement creation date/time

The date on which the Acknowledgement message was sent. The date and optional time format is identical to that used in [H.15 Message creation date/time](#) above. Mandatory and non-repeating.

Reference name	<AcknowledgementSentDateTime>
Short tag	<m487>
Cardinality	1
Example	<AcknowledgementSentDateTime>20150122T1245Z </AcknowledgementSentDateTime> (12.45pm UTC, 22 January 2015)

H.19 Acknowledgement note

Free text giving additional information about the acknowledgement itself (not about the status of the original ONIX message). Optional, and repeatable in order to provide a note in multiple languages. The *language* attribute is optional for a single instance of <AcknowledgementNote>, but must be included in each instance if <AcknowledgementNote> is repeated.

Format	Variable-length text, suggested maximum 500 characters
Reference name	<AcknowledgementNote>
Short tag	<m488>
Cardinality	0...n
Attributes	<i>language</i>
Example	<AcknowledgementNote>Reply delayed due to public holiday</AcknowledgementNote>

H.20 Status of message

An ONIX code indicating the receipt status of the original ONIX message to which the acknowledgement is a response. Mandatory and non-repeating. Some status codes may also allow or require an accompanying date in <MessageStatusDate>.

Format	Fixed-length text, 2 digits
Code list	List 221
Reference name	<MessageStatus>
Short tag	<m489>
Cardinality	1
Example	<m489>03</m489> (message processed in full)

Message status date composite

A group of data elements which together specify a date linked to the receipt status of the original ONIX message to which the acknowledgement is a response. Optional and repeatable.

Reference name	<MessageStatusDate>
Short tag	<messagestatusdate>
Cardinality	0...n

H.21 Message status date role

An ONIX code indicating the significance of the date, for example the expected date for processing the original ONIX message, or the expected date when processed data will be available to supply chain partners further downstream. Mandatory in each instance of the <MessageStatusDate> composite, and non-repeating.

Format	Fixed-length text, 2 digits
Code list	List 222
Reference name	<MessageStatusDateRole>
Short tag	<m490>
Cardinality	1
Example	<m490>01</m490> (Expected date to parse and ingest)

H.22 Date

The date specified in the <MessageStatusDateRole> field. Mandatory in each occurrence of the <MessageStatusDate> composite, and non-repeating. <Date> may carry a *dateformat* attribute that specifies the format of the date. If the attribute is missing, then the default format is YYYYMMDD.

Format	As specified by the value of the <i>dateformat</i> attribute, or the default YYYYMMDD
Reference name	<Date>
Short tag	<b306>
Cardinality	1
Attributes	<i>dateformat</i>
Example	<Date dateformat="13">20150122T2205Z</Date> (10:05pm UTC, 22 Jan 2015)

End of message status date composite

H.23 Message status note

Optional free text giving additional information about the status of the original ONIX message. This may be an error message, query or comment relating to the whole ONIX message (not to a particular Product record). Repeatable in order to provide a note in multiple languages. The *language* attribute is optional for a single instance of <MessageStatusNote>, but must be included in each instance if <MessageStatusNote> is repeated.

In most implementations, *either* a <MessageStatusNote> (which might be repeated in multiple languages) *or* one or more <MessageStatusDetail> composites should be provided. In general, the use of the <MessageStatusDetail> composite to provide more structured status information is strongly preferred. However, both <MessageStatusNote> and <MessageStatusDetail> may be omitted, for example when the original ONIX message has been received but not processed – in this circumstance, <MessageStatus> would be 00, there may be an expected date for processing the data, and none of <MessageStatusNote>, <MessageStatusDetail> or <RecordStatusSummary> are necessary.

For errors, queries or comments relating to particular Product records within the original ONIX message, see <RecordStatusNote> and <RecordStatusDetail> within the <Product> composite of the Acknowledgement message.

Format	Variable-length text, suggested maximum 500 characters
Reference name	<MessageStatusNote>
Short tag	<m491>
Cardinality	0...n
Attributes	<i>language</i>
Example	<MessageStatusNote>Queued for processing – previous message missing</MessageStatusNote>

Message status detail composite

A group of data elements which together provide detailed information about the status of the original ONIX message. Optional, and repeatable to provide multiple detailed status messages. Each message may be an error message arising from the processing of the original ONIX message by the sender of the Acknowledgement message, a query or a comment relating to the whole ONIX message (not to a particular Product record).

For errors, queries or comments relating to particular Product records within the original ONIX message, see <RecordStatusNote> and <RecordStatusDetail> within the <Product> composite of the Acknowledgement message.

Reference name	<MessageStatusDetail>
Short tag	<messagestatusdetail>
Cardinality	0...n

H.24 Message status detail code type

A mandatory, non-repeating ONIX code indicating whether <StatusDetailCode> uses a proprietary error or message code, or whether it uses a standard ONIX error or message code.

Format	Fixed-length, two digits
Code list	List 223
Reference name	<StatusDetailCodeType>
Short tag	<a492>
Cardinality	1
Example	<a492>01</a492> (Proprietary)

H.25 Message status detail code type name

A name for the code scheme from which the value of <StatusDetailCode> is taken. Must be included when, and only when, <StatusDetailCodeType> indicates that the scheme is proprietary.

The name might typically be the name of the application, software library or database that receives or processes the original ONIX message, or a named list of proprietary error codes or messages.

Format	Variable-length text, suggested maximum 50 characters
Reference name	<StatusDetailCodeTypeName>
Short tag	<a493>
Cardinality	0...1
Attributes	<i>language</i>
Example	<StatusDetailCodeTypeName>BowkerErrorNo</StatusDetailCodeTypeName>

H.26 Message status detail type

A mandatory and non-repeating ONIX code indicating the type or severity of the status detail information – for example whether the Message status detail composite is merely for information (*eg* guidance on how the message might be improved to meet best practice), or whether it indicates a severe error that prevents further processing of the message.

Format	Fixed-length, single ASCII character
Code list	List 224
Reference name	<StatusDetailType>
Short tag	<a494>
Cardinality	1
Example	<a494>F</a494> (Severe error that prevents further processing)

H.27 Message status detail code

ONIX code or proprietary application-specific code indicating the message or error type – typically an error code returned by the application or software library routine parsing or ingesting the original ONIX message. Optional, but if omitted then <StatusDetailText> must be provided – and whenever possible, inclusion of both is strongly recommended.

Format	Variable-length text, suggested maximum 20 characters
Reference name	<StatusDetailCode>
Short tag	<a495>
Cardinality	0...1
Example	<a495>E0012</a495> (proprietary error code)

H.28 Message status detail text

Text of the message or describing the error – typically an error message returned by the application or software library routine parsing or ingesting the original ONIX message. Optional, and repeatable to provide the message in multiple languages. The language attribute is optional for a single instance of <StatusDetailText>, but must be included in each instance if <MessageStatusNote> is repeated.

If <StatusDetailText> is omitted, then <StatusDetailCode> must be provided – and whenever possible, inclusion of both is strongly recommended.

Format	Variable-length text, suggested maximum 500 characters
Reference name	<StatusDetailText>
Short tag	<a496>
Cardinality	0...n
Attributes	<i>language</i>
Example	<StatusDetailText><MessageNumber> and <SentDateTime> conflict, order of update messages cannot be reliably determined</StatusDetailText> (proprietary error message)
Notes	Note that < and & characters in the error text must be escaped

H.29 Message status detail XPath

A pointer indicating a position within the original ONIX message to which the message status message relates, for example the location associated with a parsing error or with a particular information message or query. Optional, and repeatable if the message relates to multiple locations within the original ONIX message.

Format	Variable-length text conforming to the XPath specification
Reference name	<StatusDetailXPath>
Short tag	<a497>
Cardinality	0...n
Example	<a497>/ONIXMessage/Header/SentDateTime</a497>

End of message status detail composite

Record status summary composite

Optional and repeatable group of data elements which, when provided, summarises the number of Product records in the original ONIX message that have been processed, and their status.

<RecordStatusSummary> should be omitted when <MessageStatus> indicates that the original ONIX message has been received but not yet processed (code 00). If processing has begun, one or more <RecordStatusSummary> composites must be present, one for each of the distinct record statuses encountered. Each summary composite reports the count of records with each particular status. Each

Product record processed from the original ONIX message should be included in the count for the relevant <RecordStatus>, but statuses for which the count is zero should be omitted.

For example, if an ONIX message containing 16 Product records has been fully processed by the data recipient, and one of those Product records contains an error that prevents any part of the data for that product being ingested, then there should be two or more repeats of the <RecordStatusSummary>. One <RecordStatusSummary> should report 15 records without errors and another <RecordStatusSummary> should report 1 record rejected. The total number of records reported in repeats of <RecordStatusSummary> within the Acknowledgement message should be 16.

Note that it is possible to report *partial* processing of the ONIX message, by summarising the status of the records processed to date.

Reference name	<RecordStatusSummary>
Short tag	<recordstatussummary>
Cardinality	0...n

H.30 Record status

An ONIX code for the recipient's (the sender of the Acknowledgement message) processing status of a group of Product records from the original ONIX message. Mandatory within each instance of the <RecordStatusSummary> composite, and non-repeating.

Format	Variable-length text conforming to the XPath specification
Codelist	List 226
Reference name	<RecordStatus>
Short tag	<a498>
Cardinality	1
Example	<RecordStatus>00</RecordStatus> (Record without any info, query or error messages)

H.31 Record status count

The number of Product records from the original ONIX message which have the status indicated in <RecordStatus>. Mandatory within each instance of the <RecordStatusSummary> composite, and non-repeating.

Format	Variable-length positive integer
Reference name	<NumberOfRecords>
Short tag	<m499>
Cardinality	1
Example	<NumberOfRecords>15</NumberOfRecords>

End of record status summary composite

End of header composite

4. ONIX for Books Acknowledgement Product record

Every ONIX Acknowledgement message must contain either one or more <Product> composites or a single <NoProduct/> empty element.

Product composite

In the ONIX for Books Acknowledgement message, a <Product> composite represents the detailed status related to a single Product record in the original ONIX message to which the Acknowledgement is a response. The Acknowledgement's <Product> composite consists of an identifier for the original Product record, a status for that record, and a number of informational comments, queries and error messages.

Reference name	<Product>
Short tag	<product>
Cardinality	0...n

P.1 Record reference

Mandatory and non-repeating record reference taken exactly from the original ONIX message and identifying the original Product record to which the Acknowledgement <Product> composite refers.

Format	Variable-length, alphanumeric, suggested maximum length 100 characters
Reference name	<RecordReference>
Short tag	<a001>
Cardinality	1
Example	<a001>com.xyzpublishers.onix.9780001234567</a001>

P.2 Status of Product record

An ONIX code indicating the receipt and processing status of the Product record from the original ONIX message. Mandatory and non-repeating.

Format	Fixed-length, two digits
Code list	List 226
Reference name	<RecordStatus>
Short tag	<a498>
Cardinality	1
Example	<a498>03</a498> (record rejected)

Product identifier composite

An optional and repeatable group of data elements which together define an identifier of the product in accordance with a specified scheme. In many implementations of the Acknowledgement message where the requirement is simply to provide an acknowledgement and any error messages, this composite is unnecessary, because clear identification of the particular product record from the original ONIX message is provided by the mandatory <RecordReference>. However, <ProductIdentifier> provides a mechanism for standard and proprietary product identifiers (for example those assigned by the sender of the Acknowledgement message) to be included as part of the acknowledgement and passed back to the sender of the original ONIX message.

Reference name	<ProductIdentifier>
Short tag	<productidentifier>
Cardinality	1...n

P.3 Product identifier type code

An ONIX code identifying the scheme from which the identifier in the <IDValue> element is taken. Mandatory in each occurrence of the <ProductIdentifier> composite, and non-repeating.

Format	Fixed-length text, two digits
Code list	List 5
Reference name	<ProductIDType>
Short tag	<b221>
Cardinality	1
Example	<ProductIDType>03</ProductIDType> (GTIN-13)

P.4 Identifier type name

A name which identifies a proprietary identifier scheme (*ie* a scheme which is not a standard and for which there is no individual ID type code). Must be used when, and only when, the code in the <ProductIDType> element indicates a proprietary scheme, *eg* a wholesaler's own code. Optional and non-repeating.

Format	Variable-length text, suggested maximum length 50 characters
Reference name	<IDTypeName>
Short tag	<b233>
Cardinality	0...1
Attributes	<i>language</i>
Example	<IDTypeName>KNV</IDTypeName>

P.5 Identifier value

An identifier of the type specified in the <ProductIDType> element. Mandatory in each occurrence of the <ProductIdentifier> composite, and non-repeating.

Format	According to the identifier type specified in <ProductIDType>
Reference name	<IDValue>
Short tag	<b244>
Cardinality	1
Example	<b244>9780300117264</b244>

End of product identifier composite

P.6 Product record status note

Optional free text giving additional information about the status of the Product record from the original ONIX message. This may be an error message, query or comment relating to the Product record (not to the whole ONIX message). Repeatable in order to provide a note in multiple languages. The *language* attribute is optional for a single instance of <RecordStatusNote>, but must be included in each instance if <RecordStatusNote> is repeated.

In most implementations, *either* a <RecordStatusNote> (which might be repeated in multiple languages) *or* one or more <RecordStatusDetail> composites should be provided. In general, the use of the <RecordStatusDetail> composite to provide more structured status information is strongly preferred. Rarely – if for example the Acknowledgement is being used solely to pass the ONIX recipient's newly-assigned identifier back to the sender of the original ONIX message – both <RecordStatusNote> and <RecordStatusDetail> may be omitted.

For errors, queries or comments relating to the original ONIX message as a whole, see <MessageStatusNote> and <MessageStatusDetail> within the <Header> composite of the Acknowledgement message.

Format	Variable-length, suggested maximum 500 characters
Reference name	<RecordStatusNote>
Short tag	<a500>
Cardinality	0...n
Attributes	<i>language</i>
Example	<a500>Record rejected as no suitable identifiers are provided</a500>

Product record status detail composite

A group of data elements which together provide detailed information about the status of the Product record from the original ONIX message. Optional, and repeatable to provide multiple detailed status messages. Each status message may be an error message arising from the processing of the Product record by the sender of the Acknowledgement message, a query or a comment relating to the product Record (not to the original ONIX message as a whole).

For errors, queries or comments relating to the original ONIX message as a whole, see <MessageStatusNote> and <MessageStatusDetail> within the <Header> composite of the Acknowledgement message.

Reference name	<RecordStatusDetail>
Short tag	<recordstatusdetail>
Cardinality	0...n

P.7 Product record status detail code type

A mandatory, non-repeating ONIX code indicating whether <StatusDetailCode> uses a proprietary error or message code, or whether it uses a standard ONIX error or message code.

Format	Fixed-length, two digits
Code list	List 223
Reference name	<StatusDetailCodeType>
Short tag	<a492>
Cardinality	1
Example	<a492>01</a492> (Proprietary)

P.8 Product record status detail code type name

A name for the code scheme from which the value of <StatusDetailCode> is taken. Must be included when, and only when, <StatusDetailCodeType> indicates that the scheme is proprietary.

The name might typically be the name of the application, software library or database that receives or processes the original ONIX message, or a named list of proprietary error codes or messages.

Format	Variable-length text, suggested maximum 50 characters
Reference name	<StatusDetailCodeTypeName>
Short tag	<a493>
Cardinality	0...1
Attributes	<i>language</i>
Example	<StatusDetailCodeTypeName>BowkerErrorNo</StatusDetailCodeTypeName>

P.9 Product record status detail type

A mandatory and non-repeating ONIX code indicating the type or severity of the status detail information – for example whether the Record status detail composite is merely for information (eg guidance on how the Product record might be improved to meet best practice), or whether it indicates a severe error that prevents further processing of the record.

Format	Fixed-length, single ASCII character
Code list	List 224
Reference name	<StatusDetailType>
Short tag	<a494>
Cardinality	1
Example	<a494>F</a494> (Severe error)

P.10 Product record status detail code

ONIX code or proprietary application-specific code indicating the error, warning or information message type – typically an error code returned by the application or software library routine parsing or ingesting the original ONIX Product record. Optional, but if omitted then <StatusDetailText> *must* be provided – and whenever possible, inclusion of both is strongly recommended.

Format	Variable-length text, suggested maximum 20 characters
Reference name	<StatusDetailCode>
Short tag	<a495>
Cardinality	0...1
Example	<a495>E0012</a495> (Proprietary error code)

P.11 Product record status detail text

Text describing the error, warning or information message – typically an error message returned by the application or software library routine parsing or ingesting the original ONIX Product record. Optional, and repeatable to provide the message in multiple languages. The *language* attribute is optional for a single instance of <StatusDetailText>, but must be included in each instance if <StatusDetailText> is repeated.

If <StatusDetailText> is omitted, then <StatusDetailCode> must be provided – and whenever possible, inclusion of both is strongly recommended.

Format	Variable-length text, suggested maximum length 500 characters
Reference name	<StatusDetailText>
Short tag	<a496>
Cardinality	0...n
Attributes	<i>language</i>
Example	<StatusDetailText><ProductForm> ED & <Measure> are inconsistent</StatusDetailText> (proprietary error message)
Notes	Note that < and & characters in the error text must be escaped

P.12 Product record status detail XPath

A pointer indicating a position within the original ONIX message to which the Product record status message relates, for example the location associated with a parsing error or with a particular information message or query. Optional, and repeatable if the message relates to multiple locations within the original ONIX message.

Format	Variable-length text conforming to the XPath specification
Reference name	<StatusDetailXPath>
Short tag	<a497>
Cardinality	0...n
Example	<a497>/ONIXMessage/Product[4]/DescriptiveDetail/Measure[1]</a497>

End of record status detail composite

End of product composite

P.13 “No product” indicator

An empty element that provides a positive indication that an Acknowledgement message does not carry any Product records. Intended to be used in acknowledgements providing confirmation that there have been no errors or issues encountered. Optional and non-repeating, but must be used in an Acknowledgement message that contains no <Product> composites.

Format	XML empty element
Reference name	<NoProduct/>
Short tag	<x507/>
Cardinality	0...1
Example	<x507/>

5. Appendix

A.1 Data element summary

The summary table shows all the data elements which may occur in the Acknowledgement message, in the order they must occur, with both Reference names and the equivalent Short tags.

Data elements are linked back to their definitions. Leader dots and indentation of Reference names indicate the nesting structure of the composites and data elements.

The rightmost column indicates the cardinality of each data element – whether the element is optional (0...1), optional and repeatable (0...n), mandatory (1) or mandatory and repeatable (1...n).

Where data element content is controlled by a code list, the list number is shown in the adjacent List/Attr column. This column also indicates data elements that may carry a *dateformat*, *language* or *release* attribute, using the abbreviations *df*, *la* and *rl*. Note that all elements and composites may carry the *timestamp*, *sourcename* and *sourcetype* attributes.

Data element summary				
#	Reference name	Short tag	List/Attr	Card
Message				
	<ONIXMessageAcknowledgement>	<ONIXmessage acknowledgement>	rl	1
Message header				
	.<Header>	<header>		1
	..<Sender>	<sender>		1
	...<SenderIdentifier>	<senderidentifier>		0...n
H.1<SenderIDType>	<m379>	44	1
H.2<IDTypeName>	<b233>	la	0...1
H.3<IDValue>	<b244>		1
H.4	...<SenderName>	<x298>		0...1
H.5	...<ContactName>	<x299>		0...1
H.6	...<EmailAddress>	<j272>		0...1
	..<Addressee>	<addressee>		0...n
	...<AddresseeIdentifier>	<addresseeidentifier>		0...n
H.7<AddresseeIDType>	<m380>	44	1
H.8<IDTypeName>	<b233>	la	0...1
H.9<IDValue>	<b244>		1
H.10	...<AddresseeName>	<x300>		0...1
H.11	...<ContactName>	<x299>		0...1
H.12	...<EmailAddress>	<j272>		0...1
H.13	..<MessageNumber>	<m180>		0...1
H.14	..<MessageRepeat>	<m181>		0...1
H.15	..<SentDateTime>	<x307>		1
H.16	..<AcknowledgementNumber>	<m485>		0...1
H.17	..<AcknowledgementRepeat>	<m486>		0...1
H.18	..<AcknowledgementSentDateTime>	<m487>		1
H.19	..<AcknowledgementNote>	<m488>	la	0...n

H.20	..<MessageStatus>	<m489>	221	1
	..<MessageStatusDate>	<messagestatusdate>		0...n
H.21	...<MessageStatusDateRole>	<m490>	222	1
H.22	...<Date>	<b306>	df	1
H.23	..<MessageStatusNote>	<m491>	la	0...n
	..<MessageStatusDetail>	<messagestatusdetail>		0...n
H.24	...<StatusDetailCodeType>	<a492>	223	1
H.25	...<StatusDetailCodeTypeName>	<a493>	la	0...1
H.26	...<StatusDetailType>	<a494>	224	1
H.27	...<StatusDetailCode>	<a495>		0...1
H.28	...<StatusDetailText>	<a496>	la	0...n
H.29	...<StatusDetailXPath>	<a497>		0...n
	..<RecordStatusSummary>	<recordstatussummary>		0...n
H.30	...<RecordStatus>	<a498>	226	1
H.31	...<NumberOfRecords>	<m499>		1
Product record				
	.<Product>	<product>		0...n
P.1	..<RecordReference>	<a001>		1
P.2	..<RecordStatus>	<a498>	226	1
	..<ProductIdentifier>	<productidentifier>		0...n
P.3	...<ProductIDType>	<b221>	5	1
P.4	...<IDTypeName>	<b233>	la	0...n
P.5	...<IDValue>	<b244>		1
P.6	..<RecordStatusNote>	<a500>	la	0...1
	..<RecordStatusDetail>	<recordstatusdetail>		0...n
P.7	...<StatusDetailCodeType>	<a492>	223	1
P.8	...<StatusDetailCodeTypeName>	<a493>	la	0...1
P.9	...<StatusDetailType>	<a494>	224	1
P.10	...<StatusDetailCode>	<a495>		0...1
P.11	...<StatusDetailText>	<a496>	la	0...n
P.12	...<StatusDetailXPath>	<a497>		0...n
P.13	.<NoProduct>	<x507/>		0...1

A.2 Sample messages

- [Simple Acknowledgement noting receipt of an ONIX message](#)
- [Acknowledgement indicating that all records in an ONIX message processed without issues](#)
- [Acknowledgement indicating an error in one record in an ONIX message](#)
- [Acknowledgement noting partial processing of a large ONIX message](#)

There are two versions of each sample message, using Reference names and Short tags, and their content is identical. Throughout the examples, data in red indicates values taken from the ONIX code lists, text in blue indicates free text or other content not taken from code lists. Lines are numbered and annotated for convenience – an actual ONIX Acknowledgement message would contain only the XML in the middle column of the table. The examples are indented for clarity, but a real message should (ideally) not be indented.

Note that the message is sent using the UTF-8 character encoding, so non-ASCII characters in textual data elements need no special encoding.

Simple Acknowledgement noting receipt of an ONIX message

#	Using Reference names	Commentary
1	<code><?xml version="1.0" encoding="UTF-8"?></code>	File may contain extended characters without need for special coding
2	<code><ONIXMessageAcknowledgement release="3.0" xmlns="http://ns.editeur.org/onix/3.0/acknowledgement/reference"></code>	Includes optional namespace
3	<code><Header></code>	
4	<code><Sender></code>	The recipient of the original ONIX message
5	<code><SenderName>Waterstones</SenderName></code>	
6	<code></Sender></code>	
7	<code><Addressee></code>	The sender of the original ONIX message
8	<code><AddresseeName>Publisher GmbH</AddresseeName></code>	
9	<code></Addressee></code>	
10	<code><MessageNumber>571</MessageNumber></code>	Message number and sent date
11	<code><SentDateTime>20130327T1510Z</SentDateTime></code>	...of the original ONIX message
12	<code><AcknowledgementNumber>1</AcknowledgementNumber></code>	First acknowledgement
13	<code><AcknowledgementSentDateTime>20130327T1805Z</AcknowledgementSentDateTime></code>	
14	<code><MessageStatus>00</MessageStatus></code>	Message received
15	<code><MessageStatusDate></code>	
16	<code><MessageStatusDateRole>01</MessageStatusDateRole></code>	Expected to be processed
17	<code><Date dateformat="00">20130328</Date></code>	...tomorrow
18	<code></MessageStatusDate></code>	
19	<code></Header></code>	
20	<code><NoProduct/></code>	
21	<code></ONIXMessageAcknowledgement></code>	

#	<i>using Short tags</i>	<i>Commentary</i>
1	<?xml version="1.0" encoding="UTF-8"?>	File may contain extended characters without need for special coding
2	<ONIXmessageacknowledgement release="3.0" xmlns="http://ns.editeur.org/onix/3.0/acknowledgement/short">	Includes optional namespace
3	<header>	
4	<sender>	The recipient of the original ONIX message
5	<x298>Waterstones</x298>	
6	</sender>	
7	<addressee>	The sender of the original ONIX message
8	<x300>Publisher GmbH</x300>	
9	</addressee>	
10	<m180>571</m180>	Message number and sent date
11	<x307>20130327T1510Z</x307>	...of the original ONIX message
12	<m485>1</m485>	First acknowledgement
13	<m487>20130327T1805Z</m487>	
14	<m489>00</m489>	Message received
15	<messagestatusdate>	
16	<m490>01</m490>	Expected to be processed
17	<b306 dateformat="00">20130328</b306>	...tomorrow
18	</messagestatusdate>	
19	</header>	
20	<x507/>	
21	</ONIXmessageacknowledgement>	

Acknowledgement indicating that all records in original ONIX message processed without issues, data will be available to customers in four days time

#	<i>Using Reference names</i>	<i>Commentary</i>
1	<?xml version="1.0" encoding="UTF-8"?>	
2	<ONIXMessageAcknowledgement release="3.0" xmlns="http://ns.editeur.org/onix/3.0/acknowledgement/reference">	
3	<Header>	
4	<Sender>	The recipient of the original ONIX message
5	<SenderIdIdentifier>	
6	<SenderIdType>06</SenderIdType>	Identifier is a GLN
7	<IDValue>5030670165841</IDValue>	
8	</SenderIdIdentifier>	
9	<SenderName>Waterstones</SenderName>	
10	<EmailAddress>onix.qa@waterstones.com</EmailAddress>	
11	</Sender>	

12	<Addressee>	The sender of the original ONIX message
13	<AddresseeName>Publisher GmbH </AddresseeName>	
14	</Addressee>	
15	<MessageNumber>571</MessageNumber>	Message number and sent date
16	<SentDateTime>20130327T1510Z</SentDateTime>	...of the original ONIX message
17	<AcknowledgementNumber>2 </AcknowledgementNumber>	Second acknowledgement to that ONIX message (first may have been simple acknowledgement of receipt)
18	<AcknowledgementSentDateTime>20130328T1345Z </AcknowledgementSentDateTime>	
19	<MessageStatus>03</MessageStatus>	Message processed in full
20	<MessageStatusDate>	
21	<MessageStatusDateRole>02 </MessageStatusDateRole>	Date data expected to be available to downstream partners
22	<Date>20130401</Date>	Default date format is YYYYMMDD
23	</MessageStatusDate>	
24	<RecordStatusSummary>	
25	<RecordStatus>00</RecordStatus>	37 records processed
26	<NumberOfRecords>37</NumberOfRecords>	...without issues
27	</RecordStatusSummary>	
28	</Header>	
29	<NoProduct/>	No Product record-specific status messages
30	</ONIXMessageAcknowledgement>	
#	using Short tags	Commentary
1	<?xml version="1.0" encoding="UTF-8"?>	
2	<ONIXmessageacknowledgement release="3.0" xmlns="http://ns.editeur.org/onix/3.0/acknowledgement/short">	
3	<header>	
4	<sender>	The recipient of the original ONIX message
5	<senderidentifier>	
6	<m379>06</m379> <b244>5030670165841</b244>	Identifier is a GLN
8	</senderidentifier>	
9	<x298>Waterstones</x298>	
10	<j272>onix.qa@waterstones.com</j272>	
11	</sender>	
12	<addressee>	The sender of the original ONIX message
13	<x300>Publisher GmbH</x300>	
14	</addressee>	

15	<m180>571</m180>	Message number and sent date
16	<x307>20130327T1510Z</x307>	...of the original ONIX message
17	<m485>2</m485>	Second acknowledgement to that ONIX message (first may have been simple acknowledgement of receipt)
18	<m487>20130328T1345Z</m487>	
19	<m489>03</m489>	Message processed in full
20	<messagestatusdate>	
21	<m490>02</m490>	Date data expected to be available to downstream partners
22	<b306>20130401</b306>	Default date format is YYYYMMDD
23	</messagestatusdate>	
24	<recordstatussummary>	
25	<a498>00</a498>	37 records processed
26	<m499>37</m499>	...without issues
27	</recordstatussummary>	
28	</header>	
29	<x507/>	No Product record-specific status messages
30	</ONIXmessageacknowledgement>	

Acknowledgement indicating an error in one record in an ONIX (it has a non-fatal internal contradiction, and is sent too early)

#	Using Reference names	Commentary
1	<?xml version="1.0" encoding="UTF-8"?>	
2	<ONIXMessageAcknowledgement release="3.0" xmlns="http://ns.editeur.org/onix/3.0/acknowledgement/reference">	
3	<Header>	
4	<Sender>	
5	<SenderIdentifier>	
6	<SenderIDType>06</SenderIDType>	Identifier is a GLN
7	<IDValue>5030670165841</IDValue>	
8	</SenderIdentifier>	
9	<SenderName>Waterstones</SenderName>	
10	<EmailAddress>onix.qa@waterstones.com</EmailAddress>	
11	</Sender>	
12	<Addressee>	The sender of the original ONIX message
13	<AddresseeName>Publisher GmbH</AddresseeName>	
14	</Addressee>	
15	<MessageNumber>571</MessageNumber>	Message number and sent date

16	<SentDateTime>20130327T1510Z</SentDateTime>	...of the original ONIX message
17	<AcknowledgementNumber>2</AcknowledgementNumber>	Second acknowledgement to that ONIX message
18	<AcknowledgementSentDateTime>20130328T1345Z</AcknowledgementSentDateTime>	
19	<MessageStatus>03</MessageStatus>	Message processed in full
20	<MessageStatusDate>	
21	<MessageStatusDateRole>02</MessageStatusDateRole>	Date data expected to be available to downstream partners
22	<Date>20130401</Date>	
23	</MessageStatusDate>	
24	<RecordStatusSummary>	
25	<RecordStatus>00</RecordStatus>	36 records processed
26	<NumberOfRecords>36</NumberOfRecords>	...without issues
27	</RecordStatusSummary>	
28	<RecordStatusSummary>	
29	<RecordStatus>02</RecordStatus>	1 record processed
30	<NumberOfRecords>1</NumberOfRecords>	without error(s)
31	</RecordStatusSummary>	
32	</Header>	
33	<Product>	
34	<RecordReference>de.com.publisher.36036</RecordReference>	
35	<RecordStatus>02</RecordStatus>	
36	<RecordStatusDetail>	
37	<StatusDetailCodeType>02</StatusDetailCodeType>	Code in <StatusDetailCode> taken from List 225
38	<StatusDetailType>E</StatusDetailType>	Error
39	<StatusDetailCode>000</StatusDetailCode>	Unknown error
40	<StatusDetailText>Inclusion of diameter measurement not compatible with <ProductForm></StatusDetailText>	Error description
41	<StatusDetailXPath>/ONIXMessage/Product[RecordReference='de.com.publisher.36036']/DescriptiveDetail/ProductForm</StatusDetailXPath>	XPaths of conflicting data elements
42	<StatusDetailXPath>/ONIXMessage/Product[RecordReference='de.com.publisher.36036']/DescriptiveDetail/Measure/MeasureType[.='12']</StatusDetailXPath>	
43	</RecordStatusDetail>	
44	<RecordStatusDetail>	
45	<StatusDetailCodeType>01</StatusDetailCodeType>	Code in <StatusDetailCode> is proprietary
46	<StatusDetailCodeTypeName>Phoenix Ingest</StatusDetailCodeTypeName>	
47	<StatusDetailType>W</StatusDetailType>	Warning
48	<StatusDetailCode>Q105</StatusDetailCode>	

49	<StatusDetailText language="eng">Do not normally list new products in catalog more than two years in advance.</StatusDetailText>	Bilingual warning text
50	<StatusDetailText language="ger">Wir normalerweise nicht neue Produkte im Katalog mehr als zwei Jahre Liste im Voraus.</StatusDetailText>	
51	<StatusDetailXPath>/ONIXMessage/Product[27]/PublishingDetail/PublishingDate[1]/Date</StatusDetailXPath>	
52	</RecordStatusDetail>	
53	</Product>	
54	</ONIXMessageAcknowledgement>	
#	using Short tags	Commentary
1	<?xml version="1.0" encoding="UTF-8"?>	
2	<ONIXmessageacknowledgement release="3.0" xmlns="http://ns.editeur.org/onix/3.0/acknowledgement/short">	
3	<header>	
4	<sender>	
5	<senderidentifier>	
6	<m379>06</m379> <b244>5030670165841</b244>	Identifier is a GLN
8	</senderidentifier>	
9	<x298>Waterstones</x298>	
10	<j272>onix.qa@waterstones.com</j272>	
11	</sender>	
12	<addressee>	The sender of the original ONIX message
13	<x300>Publisher GmbH</x300>	
14	</addressee>	
15	<m180>571</m180>	Message number and sent date
16	<x307>20130327T1510Z</x307>	...of the original ONIX message
17	<m485>2</m485>	Second acknowledgement to that ONIX message
18	<m487>20130328T1345Z</m487>	
19	<m489>03</m489>	Message processed in full
20	<messagestatusdate>	
21	<m490>02</m490>	Date data expected to be available to downstream partners
22	<b306>20130401</b306>	
23	</messagestatusdate>	
24	<recordstatussummary>	
25	<a498>00</a498>	36 records processed
26	<m499>36</m499>	without issues
27	</recordstatussummary>	
28	<recordstatussummary>	
29	<a498>02</a498>	One record processed

30	<code><m499>1</m499></code>	with error(s)
31	<code></recordstatussummary></code>	
32	<code></header></code>	
33	<code><product></code>	
34	<code><a001>de.com.publisher.36036</a001></code>	
35	<code><a498>02</a498></code>	
36	<code><recordstatusdetail></code>	
37	<code><a492>02</a492></code>	Code in <StatusDetailCode> taken from List 225
38	<code><a494>E</a494></code>	Error
39	<code><a495>000</a495></code>	Unknown error
40	<code><a496>Inclusion of diameter measurement not compatible with &lt;ProductForm></a496></code>	Error description
41	<code><a497>/ONIXMessage/Product[RecordReference='de.com.publisher.36036']/DescriptiveDetail/ProductForm</a497></code>	XPaths of conflicting data elements
42	<code><a497>/ONIXMessage/Product[RecordReference='de.com.publisher.36036']/DescriptiveDetail/Measure/MeasureType[.='12']</a497></code>	
43	<code></recordstatusdetail></code>	
44	<code><recordstatusdetail></code>	
45	<code><a492>01</a492></code>	Code in <StatusDetailCode> is proprietary
46	<code><a493>Phoenix Ingest</a493></code>	
47	<code><a494>W</a494></code>	Warning
48	<code><a495>Q105</a495></code>	
49	<code><a496 language="eng">Do not normally list new products in catalog more than two years in advance.</a496></code>	Bilingual warning text
50	<code><a496 language="ger">Wir normalerweise nicht neue Produkte im Katalog mehr als zwei Jahre Liste im Voraus.</a496></code>	
51	<code><a497>/ONIXMessage/Product[27]/PublishingDetail/PublishingDate[1]/Date</a497></code>	
52	<code></recordstatusdetail></code>	
53	<code></product></code>	
54	<code></ONIXmessageacknowledgement></code>	

In the example above, <StatusDetailXPath> shows two different styles of XPath predicate, one based on a query (eg Product[RecordReference='de.com.publisher.36036']) and one based on the order of records in the original ONIX record (eg Product[27]). The former is perhaps more robust but the latter is simpler. The simple style is recommended unless it cannot be guaranteed that the order of records and elements has not been altered by some intermediate processing between sender and recipient of the original ONIX message.

Note also that the XPath should use Reference or Short tags according to the flavor used in the original ONIX message, and need not necessarily match the choice of tag flavor used in the Acknowledgement message (Reference names are used in the XPath within the Short tags example above). However, Acknowledgements should typically use the same tag flavor as the original message, and thus the flavor used in the XPath would normally agree with the flavor used in the Acknowledgement as a whole.

Acknowledgement noting partial processing of a large ONIX message – processing of third tranche of records		
#	<i>Using Reference names</i>	<i>Commentary</i>
1	<?xml version="1.0" encoding="UTF-8"?>	
2	<ONIXMessageAcknowledgement release="3.0" xmlns="http://ns.editeur.org/onix/3.0/acknowledgement/reference">	
3	<Header>	
4	<Sender>	The recipient of the original ONIX message
5	<SenderName>Waterstones</SenderName>	
6	</Sender>	
7	<Addressee>	The sender of the original ONIX message
8	<AddresseeName>Publisher GmbH</AddresseeName>	
9	</Addressee>	
10	<MessageNumber>584</MessageNumber>	Message number and sent date
11	<SentDateTime>20130415T1235Z</SentDateTime>	...of the original ONIX message
12	<AcknowledgementNumber>4</AcknowledgementNumber>	Fourth acknowledgement
13	<AcknowledgementSentDateTime>20130416T0635Z</AcknowledgementSentDateTime>	
14	<MessageStatus>02</MessageStatus>	Message part-processed (there is at least another tranche still to be processed)
15	<RecordStatusSummary>	
16	<RecordStatus>00</RecordStatus>	75 records processed (in this tranche)
17	<NumberOfRecords>75</NumberOfRecords>	...without issues
18	</RecordStatusSummary>	
19	<RecordStatusSummary>	
20	<RecordStatus>09</RecordStatus>	Status of 150 records in previous tranche(s)
21	<NumberOfRecords>150</NumberOfRecords>	...reported in earlier acknowledgement(s)
22	</RecordStatusSummary>	
23	</Header>	
24	<NoProduct/>	No Product record-specific status messages (for this tranche)
25	</ONIXMessageAcknowledgement>	
#	<i>using Short tags</i>	<i>Commentary</i>
1	<?xml version="1.0" encoding="UTF-8"?>	
2	<ONIXmessageacknowledgement release="3.0" xmlns="http://ns.editeur.org/onix/3.0/acknowledgement/short">	
3	<header>	

4	<sender>	The recipient of the original ONIX message
5	<x298>Waterstones</x298>	
6	</sender>	
7	<addressee>	The sender of the original ONIX message
8	<x300>Publisher GmbH</x300>	
9	</addressee>	
10	<m180>584</m180>	Message number and sent date
11	<x307>20130415T1235Z</x307>	...of the original ONIX message
12	<m485>4</m485>	Fourth acknowledgement
13	<m487>20130416T0635Z</m487>	
14	<m489>02</m489>	Message part-processed (there is at least another tranche still to be processed)
15	<recordstatussummary>	
16	<m498>00</m498>	75 records processed (in this tranche)
17	<m499>75</m499>	...without issues
18	</recordstatussummary>	
19	<recordstatussummary>	
20	<m498>09</m498>	Status of 150 records in previous tranche(s)
21	<m499>150</m499>	...reported in earlier acknowledgement(s)
22	</recordstatussummary>	
23	</header>	
24	<x507/>	No Product record-specific status messages (for this tranche)
25	</ONIXmessageacknowledgement>	

In the example above, there *may* have been issues with Product records processed in the first or second tranche, but they would have been detailed in earlier Acknowledgement messages.

A.3 List of all XML tags

The tables show equivalent Reference names and Short tags, first in Reference name order, then in Short tag order. Note that in any one ONIX Acknowledgement message, Reference names and Short tags cannot be mixed.

Complete list of tag names		
<i>Reference name</i>	<i>equivalent Short tag</i>	<i>Notes</i>
<AcknowledgementNote>	<m488>	
<AcknowledgementNumber>	<m485>	
<AcknowledgementRepeat>	<m486>	
<AcknowledgementSentDateTime>	<m487>	
<Addressee>	<addressee>	
<AddresseeIdentifier>	<addresseeidentifier>	
<AddresseeIDType>	<m380>	
<AddresseeName>	<x300>	
<ContactName>	<x299>	
<Date>	<b306>	
<EmailAddress>	<j272>	
<Header>	<header>	
<IDTypeName>	<b233>	
<IDValue>	<b244>	
<MessageNumber>	<m180>	
<MessageRepeat>	<m181>	
<MessageStatus>	<m489>	
<MessageStatusDate>	<messagestatusdate>	
<MessageStatusDateRole>	<m490>	
<MessageStatusDetail>	<messagestatusdetail>	
<MessageStausNote>	<m491>	
<NoProduct>	<x507>	
<NumberOfRecords>	<m499>	
<ONIXMessageAcknowledgement>	<ONIXmessageacknowledgement>	
<Product>	<product>	
<ProductIdentifier>	<productidentifier>	
<ProductIDType>	<b221>	
<RecordReference>	<a001>	
<RecordStatus>	<a498>	
<RecordStatusDetail>	<recordstatusdetail>	
<RecordStatusNote>	<a500>	
<RecordStatusSummary>	<recordstatussummary>	
<Sender>	<sender>	
<SenderIdentifier>	<senderidentifier>	
<SenderIDType>	<m379>	
<SenderName>	<x298>	
<SentDateTime>	<x307>	
<StatusDetailCode>	<a495>	
<StatusDetailCodeType>	<a492>	
<StatusDetailCodeTypeName>	<a493>	
<StatusDetailText>	<a496>	

<StatusDetailType>	<a494>	
<StatusDetailXPath>	<a497>	
<i>equivalent Reference name</i>	<i>Short tag</i>	<i>Notes</i>
<RecordReference>	<a001>	
<StatusDetailCodeType>	<a492>	
<StatusDetailCodeTypeName>	<a493>	
<StatusDetailType>	<a494>	
<StatusDetailCode>	<a495>	
<StatusDetailText>	<a496>	
<StatusDetailXPath>	<a497>	
<RecordStatus>	<a498>	
<RecordStatusNote>	<a500>	
<ProductIDType>	<b221>	
<IDTypeName>	<b233>	
<IDValue>	<b244>	
<Date>	<b306>	
<EmailAddress>	<j272>	
<MessageNumber>	<m180>	
<MessageRepeat>	<m181>	
<SenderIDType>	<m379>	
<AddresseeIDType>	<m380>	
<AcknowledgementNumber>	<m485>	
<AcknowledgementRepeat>	<m486>	
<AcknowledgementSentDateTime>	<m487>	
<AcknowledgementNote>	<m488>	
<MessageStatus>	<m489>	
<MessageStatusDateRole>	<m490>	
<MessageStatusNote>	<m491>	
<NumberOfRecords>	<m499>	
<SenderName>	<x298>	
<ContactName>	<x299>	
<AddresseeName>	<x300>	
<SentDateTime>	<x307>	
<NoProduct>	<x507>	
<Addressee>	<addressee>	
<AddresseeIdentifier>	<addresseeidentifier>	
<Header>	<header>	
<MessageStatusDate>	<messagestatusdate>	
<MessageStatusDetail>	<messagestatusdetail>	
<ONIXMessageAcknowledgement>	<ONIXmessageacknowledgement>	
<Product>	<product>	
<ProductIdentifier>	<productidentifier>	
<RecordStatusDetail>	<recordstatusdetail>	
<RecordStatusSummary>	<recordstatussummary>	
<Sender>	<sender>	
<SenderIdentifier>	<senderidentifier>	

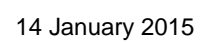
A.4 Using the Acknowledgement message with ONIX 2.1

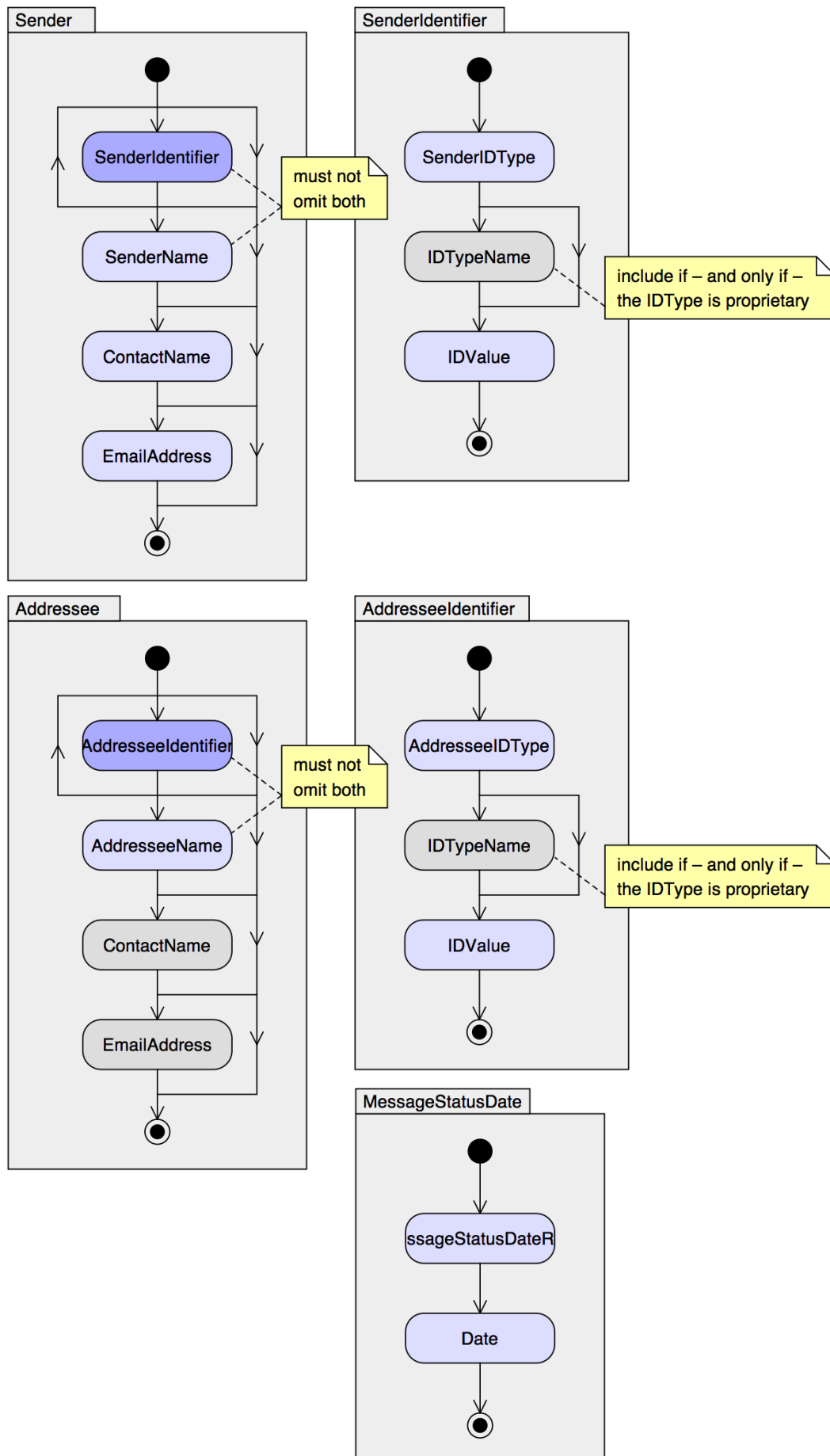
The ONIX Acknowledgement message is designed primarily for use with ONIX 3.0. It can be used to respond to receipt of an ONIX 2.1 message, but such use is not specifically supported.

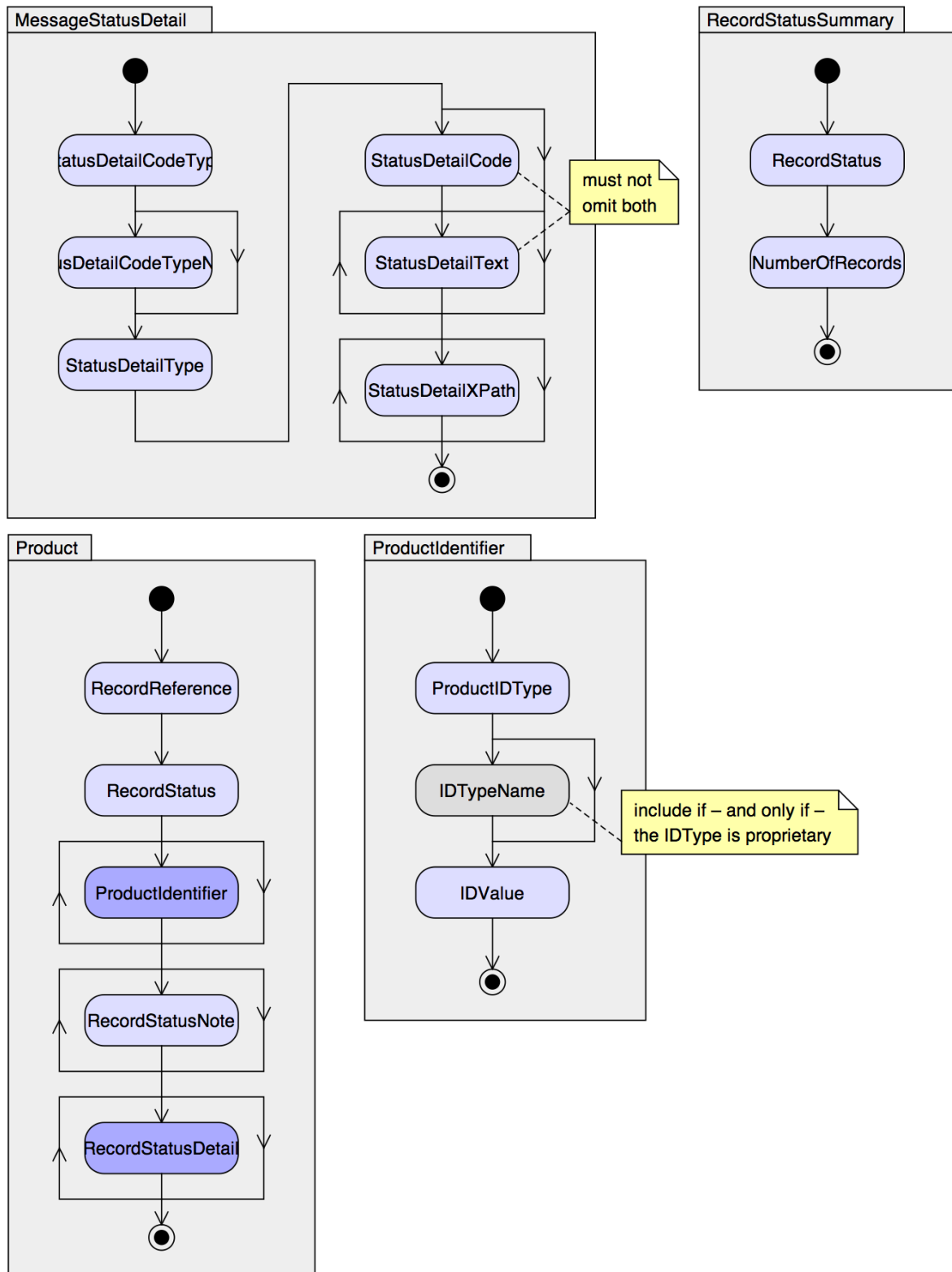
The Acknowledgement message uses a few data elements whose content is typically derived directly from related data elements in an original ONIX 3.0 message – for example any <MessageNumber> in an acknowledgement must be identical to <MessageNumber> in the original message to which the acknowledgement is a response. These data elements have clear equivalents in ONIX 2.1 too. Most are identically specified, but a few are named or specified slightly differently.

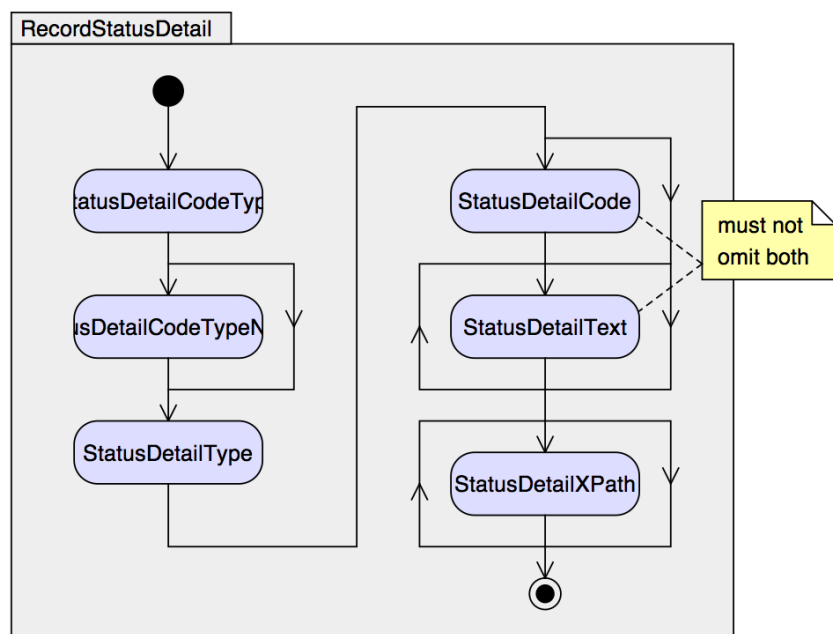
Acknowledgement tag equivalents in ONIX 2.1

<i>Reference name</i>	<i>Short tag</i>	<i>Notes</i>
<SentDateTime>	<x307>	The value for <SentDateTime> in the Acknowledgement header should be taken from the <SentDate> or <m182> element in the ONIX 2.1 message. If the <SentDate> includes a time (<i>ie</i> the data format is YYYYMMDDhhmm) then the format must be modified to YYYYMMDDThhmm (note the literal 'T') when the time is used in the Acknowledgement.
<SenderName>	<x298>	The value for <SenderName> in the Acknowledgement header should be taken from the <ToCompany> or <m178> element in the ONIX 2.1 message.
<AddresseeName>	<x300>	The value for <AddresseeName> in the Acknowledgement header should be taken from the <FromCompany> or <m174> element in the ONIX 2.1 message.
<ContactName>	<x299>	The value used for the sender or addressee <ContactName> in the Acknowledgement header should be taken from the <FromPerson> (<m175>), or <ToPerson> (<m179>) elements in the ONIX 2.1 message, as appropriate.









ONIX for Books Acknowledgement Format Specification Release 3.0 © [EDItEUR](#) 2014