XLIFF Version 1.2+

Edited by Rodolfo Raya
Bryan Schnabel
Tektronix

<bryan.s.schnabel@tektronix.com>

$Id: XLIFF-core.xml,v 1.2 2008-1-Feb admin Exp $

http://docs.oasis-open.org/xliff/xliff-core/xliff-core.html

http://docs.oasis-open.org/xliff/v1.2/os/xliff-core.pdf

http://docs.oasis-open.org/xliff/v1.2/cs02/xliff-core.html

http://docs.oasis-open.org/xliff/v1.2/cs02/xliff-core.pdf

http://docs.oasis-open.org/xliff/xliff-core/xliff-core.html

http://docs.oasis-open.org/xliff/xliff-core/xliff-core.pdf

OASIS XLIFF TC [http://www.oasis-open.org/committees/]

Related Work

This specification replaces or supersedes:

- XLIFF 1.2 OASIS Standard

Declared XML Namespaces

urn:oasis:names:tc:xliff:document:1.2+

Status

This document was last revised or approved by the XLIFF TC on the above date. The level of approval is also listed above. Check the current location noted above for possible later revisions of this document. This document is updated periodically on no particular schedule.

Technical Committee members should send comments on this specification to the Technical Committee's email list. Others should send comments to the Technical Committee by using the "Send A Comment" button on the Technical Committee' s web page at www.oasis-open.org/committees/xliff

Technical Committee members should send comments on this specification to the Technical Committee's email list. Others should send comments to the Technical Committee by using the "Send A Comment" button on the Technical Committee's web page at http://www.oasis-open.org/committees/xliff.

For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the Technical Committee web page (http://www.oasis-open.org/committees/xliffipr.php).

Notices

1 February 2011

Abstract

This document defines the XML Localization Interchange File Format (XLIFF). The purpose of this vocabulary is to store localizable data and carry it from one step of the localization process to the other, while allowing interoperability between tools.

Table of Contents

Introduction

XLIFF is the XML Localization Interchange File Format designed by a group of software providers, localization service providers, and localization tools providers. It is intended to give any software provider a single interchange file format that can be understood by any localization provider. It is loosely based on the OpenTag version 1.2 specification and borrows from the TMX 1.2 specification. However, it is different enough from either one to be its own format.

Transitional and Strict

XLIFF is specified in two "flavors". Indicate which of these variants you are using by selecting the appropriate schema. The schema may be specified in the XLIFF document itself or in an OASIS catalog. The namespace is the same for both variants. Thus, if you want to validate the document, the tool used knows which variant you are using. Each variant has its own schema that defines which elements and attributes are allowed in certain circumstances.

As newer versions of XLIFF are approved, sometimes changes are made that render some elements, attributes or constructs in older versions obsolete. Obsolete items are deprecated and should not be

used even though they are allowed. The XLIFF specification details which items are deprecated and what new constructs to use.

- *Transitional* - Applications that produce older versions of XLIFF may still use deprecated items. Deprecated elements and attributes are allowed. Non-XLIFF items are validated only to ensure they are well-formed. Use this variant to validate XLIFF documents that you read.

  ```
  xsi:schemaLocation='urn:oasis:names:tc:xliff:document:1.2 xliff-core-1.2-transitional.xsd'
  ```

- *Strict* - All deprecated elements and attributes are not allowed. Obsolete items from previous versions of XLIFF are deprecated and should not be used when writing new XLIFF documents. In order for XLIFF documents with extensions to validate, the parser MUST find the schema for namespaced elements and attributes, and elements and attributes MUST be valid. Use this variant to validate XLIFF documents that you create.

  ```
  xsi:schemaLocation='urn:oasis:names:tc:xliff:document:1.2 xliff-core-1.2-strict.xsd'
  ```

General Structure

XLIFF is an XML application, as such it begins with an XML declaration. After the XML declaration comes the XLIFF document itself, enclosed within the `<xliff>` [17] element. An XLIFF document is composed of one or more sections, each enclosed within a `<file>` [17] element. The `<file>` [17] element consists of a `<header>` [17] element, which contains metadata about the `<file>` [17] , and a `<body>` [23] element, which contains the extracted translatable data from the `<file>` [17]. The translatable data within `<trans-unit>` [24] elements is organized into `<source>` [25] and `<target>` [25] paired elements. These `<trans-unit>` [24] elements can be grouped recursively in `<group>` [23] elements.

In addition, XLIFF provides the ability to maintain information about the processing of the file via the `<phase>` [20] element. Possible translations for a specific `<source>` [25] element can be generated from any number of MT (Machine Translation) and CAT (Computer Assisted Translation) systems and stored near the `<source>` [25] in `<alt-trans>` [26] elements. Context for a `<source>` [25] that could be used by a translator or a TM (Translation Memory) system is provided by the `<context>` [22] element. Binary data can be made available via the `<bin-unit>` [27], which may also be translated and contain an associated `<trans-unit>` [24] .

It is strongly recommended that content within the `<file> [17]` element be uniformly bilingual. In other words, each `<source> [25]` and `<target> [25]` element that is a child of `<trans-unit> [24]` is of the same language as the source-language and target-language attributes of the `<file> [17]` element, respectively. The `xml:lang [69]` attribute should not be used in those elements. The exception is that `<source> [25]` and `<target> [25]` elements that are children of `<alt-trans> [26]` may contain an `xml:lang [69]` attribute of a different language than that of the `source-language [62]` and `target-language [65]` attributes of the `<file> [17]` element.

```
<xliff version='1.2' xmlns='urn:oasis:names:tc:xliff:document:1.2'> <file orig
```

The complete tree structure is available in Appendix A.

Header

The XLIFF `<header>` [17] contains metadata about the file and the localization process. It contains the `<skl>` [18], `<phase-group>` [20], `<glossary>` [19], `<reference>` [19] `<count-group>` [21] `<tool>` [65], `<prop-group>` [22], and `<note>` [19] elements. The `<skl>` [18] element contains either a skeleton file of the file submitted for localization or a hypertext link to that file.

The <phase-group> [20] element contains information about each processing phase used in localizing the file; references to these phases are stored along with the translations. The <glossary> [19] and <reference> [19] elements may contain hypertext links to a glossary and reference file, respectively, or the actual glossary and reference data that can be used in the localization process.

The <count-group> [21] element is a grouping element of count information of the entire file. The <prop-group> [22] element (deprecated) contains tool-specific information used in combining the data with the skeleton file or storing the data in a repository. The <note> [19] element contains instructions for the localization process. The <count-group> [21], <prop-group> [22], and <note> [19] elements can also appear in the body of the file.

Body

The XLIFF <body> [23] contains the structure and the localizable content from the file. It contains the <group> [23], <trans-unit> [24] and <bin-unit> [27] elements. The structure is described using the <group> [23], <trans-unit> [24], <bin-unit> [27] elements. The <group> [23] element is a general purpose structural element used in describing the hierarchy of the file; it can contain other <group> [23] elements as children as well as <trans-unit> [24] and <bin-unit> [27] elements.

The <trans-unit> [24] and <bin-unit> [27] elements contain the translatable portions of the document. The <trans-unit> [24] element contains the text to be translated, the translations, and other related information. The <bin-unit> [27] contains binary data that may or may not need to be translated; it also can contain translated versions of the binary object as well as other related information.

In the <trans-unit> [24] element the text to be translated is contained in a <source> [25] element. This element may contain inline elements that either remove the codes from the source (<g> [28], <x/> [29], <bx/> [29], <ex/> [30]) or that mask off codes left inline (<bpt> [31], <ept> [31], <sub> [32], <it> [31], <ph> [30]). The translated text is contained in a <target> [25] element that has the same inline codes available to it as does the <source> [25] element. Translation matches generated by a TM or MT or entered by a translator may be provided in an <alt-trans> [26] element, which also contains the <source> [25] and <target> [25] elements.

At every structural level contextual information for the localization process can be provided by the <context-group> [21] named group element, count information by the <count-group> [21] named group element, and tool-specific information by the <prop-group> [22] named group element (deprecated).

Named Groups

XLIFF allows grouping of certain elements into named groups. A named group is simply a grouping element with a name attribute. These named groups can be interspersed throughout the file with information designed for specific purposes. Using XML processing instructions different actions can be performed with specific named groups. The named group elements are <context-group> [21], <count-group> [21] and <prop-group> [22] (deprecated).

The <count-group> [21] element contains counts of words, translations, dialogs, or anything else that may need to be counted in the file. A different named group could be stored by the client, translator, reviewer, and localization engineer. Processing instructions could inform a system which of these <count-group> [21] to update during the localization process.

The <prop-group> [22] element contains tool specific data that can be used in creating the translated file, storing the translations, and any other specific task. Processing instructions can indicate to the tools which named <prop-group> [22] to use when updating the repository or combining the localized data with the skeleton file to create a translated file. Note that the <prop-group> [22] has been deprecated since version 1.1.

Inline Elements

The content of the <source> [25] and the <target> [25] elements can include one or more inline elements (also called "content markup"). Those elements are used to represent codes that reside within the source or target text, for example the formatting codes to mark a section of a sentence in bold.

There are three different types of inline elements:

1. Elements that have a content, and for which this content is the actual native code of the original data (escaped for XML if necessary). These elements are: <bpt> [31], <ept> [31], <it> [31], and <ph> [30].

2. Elements that are empty and act as placeholders for a native code that is either in the Skeleton file or generated automatically. These elements are: <g> [28], <bx/> [29], <ex/> [30], and <x/> [29].

3. The <sub> [32] element, which can be inside <bpt> [31], <ept> [31], <it> [31], and <ph> [30] to delimit a translatable run of text within a native inline code, for example the value of an ALT attribute in a <IMG> element in HTML.

The first two types of inline elements can be classified into three main categories depending on their function, and regardless the method they use to hold the native codes:

A) Codes that either begin or end an instruction, and whose beginning and ending functions both appear within a translation unit. For example, an instruction to begin embolden for a range of words which is then followed in the same translation unit by an instruction to end bold formatting. The elements that can handle such cases are: <bpt> [31], <ept> [31], <g> [28], <bx/> [29], and <ex/> [30].

B) Codes that either begin or end an instruction, but whose beginning and ending functions are not both contained within a single translation unit. For example, an instruction to embolden text may apply to the first of three sentences in a paragraph contained within a single translation unit, but the instruction to turn off bolding may only appear at the end of the third sentence. Its beginning instruction is present in the first translation unit, while its closing tag is present in the third translation unit. The elements that can handle such cases are: <it> [31] and <x/> [29].

C) Codes that represent self-contained functions that do not require explicit ending instructions. Images or cross-reference tokens are examples of these standalone codes. The elements that can handle such cases are: <ph> [30] and <x/> [29].

The guidelines for using the inline elements are as follows:

- Use <bpt> [31] or <bx/> [29] for opening each code that has a corresponding closing code in the content. Use <bpt> [31] to mask the code and <bx/> [29] to replace the code. The <bpt> [31] and <bx/> [29] elements should be followed by a matching <ept> [31] or <ex/> [30] element, respectively, within the same translation unit. These paired elements are matched by setting their rid [61] attributes to the same value. For example: <bpt id='2' rid='1'>xx</bpt> ... <ept id='3' rid='1'>xx</ept> and <bx id='4' rid='2'/> ... <ex id='5' rid='2'/>. If the rid [61] attribute is not present (in a 1.0 document for example), the attribute id [46] is used to match both tags. For example: <bpt id='5'>xx</bpt> ... <ept id='5'>xx</ept>.

- Use <ept> [31] or <ex/> [30] for closing each code that has a corresponding opening code in the content. Use <ept> [31] to mask the code and <ex/> [30] to replace the code. The <ept> [31] and <ex/> [30] elements should be preceded by a matching <bpt> [31] and <bx/> [29] element, respectively. These paired elements are matched by setting their rid [61] attributes to the same value. For example: <bpt id='2' rid='1'>xx</bpt> ... <ept id='3' rid='1'>xx</ept> and <bx id='4' rid='2'/> ... <ex id='5' rid='2'/>. If the rid [61] attribute

is not present (in a 1.0 document for example), the attribute `id [46]` is used to match both tags. For example: `<bpt id='5'>xx</bpt> ... <ept id='5'>xx</ept>`.

- Use `<g> [28]` to replace any inline code of the original document that has a beginning and an end and can be moved within its parent structural element.
- Use `<ph> [30]` or `<x/> [29]` for standalone codes. Use `<ph> [30]` to mask the code and `<x/> [29]` to replace the code. Standalone codes are codes that are not opening or closing of a pair, for example empty elements in XML.
- Use `<it> [31]` for opening or closing each code that has no corresponding closing or opening code in the source element. In some cases, because of the segmentation, you may have opening and closing codes that have no corresponding closing or opening codes within the same translation unit. Use `<it> [31]` to encapsulate those codes. The `<it> [31]` element has a mandatory `pos [54]` attribute that should be set to "`begin`" or "`end`" depending on whether the isolated code is an opening or a closing code.
- Use the `xid [68]` attribute of the `<bx/> [29]`, `<ex/> [30]` and `<x/> [29]` elements to relate a `<trans-unit> [24]` or `<bin-unit> [27]` that contains the content of that replaced code.
- At the time of this document's authoring, TMX 14b does not support <g> [28] and <x/> [29] inline tag equivalents. Therefore, if interchange of translation memory data with TMX is required, use <bpt> [31] and <ept> [31] tags instead of <g> [28] and `<ph> [30]` tags instead of <x/> [29].

As XLIFF inline elements are closely related to TMX inline elements, further examples of usage of these tags may be found in their specification's Content Markup section [http://www.lisa.org/tmx/tmx.htm#SectionContentMarkup].

Inline elements are normally treated as being transparent with regard to lexical processing such as segmentation or word tokenisation. If the inline element also represents a lexical function, such as implying spacial characteristics or a string of characters or symbols, then the `equiv-text` [43] attribute must be used to denote any such lexical characteristics.

For example:

```
<p>This HTML break element<br/>is not followed by a white space character</p>
```

is represented in an XLIFF document as:

```
<source>This HTML break element<x id="x1" ctype="x-html-br" equiv-text=" "/>is n
by a white space character.</source>
```

Extensibility

At times, it may be useful to extend the set of information available in an XLIFF document by inserting constructs defined in various other XML vocabularies. You can add non-XLIFF elements, as well as attributes and attribute values. Adding elements and attributes use the namespace mechanism [XML Names [77]]. Adding attribute values generally involves preceding the value by an "`x-`" (e.g. `<context context-type='x-for-engineers'>`).

Although XLIFF offers this extensibility mechanism, in order to avoid a nimiety of information and increase interoperability between tools, it is strongly recommended to use XLIFF capabilities whenever possible, rather than to create non-standard user-defined elements or attributes.

Adding Elements

XLIFF provides several extension points in the following elements: <alt-trans> [26], <bin-unit> [27], <group> [23], <header> [17], <tool> [65], <trans-unit> [24], and <xliff> [17].

Several non-XLIFF elements can be used at each extension point. The content of each element can be any valid XML content (empty content, PCDATA, mixed content, and so forth).

For example, the following XLIFF code shows how to add user-defined elements (in bold) within an XLIFF document:

```
<xliff version='1.2'
 xmlns='urn:oasis:names:tc:xliff:document:1.2'
 xmlns:sup='http://www.ChaucerState.ac.pg/Frm/XLFSup-v1'>
<file original='passus-1.doc' source-language='enm' datatype='plaintext'>
<group>
<sup:SourceInfo>
<sup:Book>Piers Plowman, Passus 1</sup:Book>

<sup:Author>William Langland</sup:Author>
</sup:SourceInfo>
 <sup:WorkInfo Task='transcription' Context='Middle-English:1360'/>
<trans-unit id='1'>
<source xml:lang='enm'>What this mountaigne bymeneth</source>
<target xml:lang='en'>What this mountain means</target>

<sup:Reference Type='strophe'>1-a</sup:Reference>
</trans-unit>
<trans-unit id='2'>
<source xml:lang='enm'>and the merke dale</source>
<target xml:lang='en'>and the dark dale</target>
<sup:Reference Type='strophe'>1-b</sup:Reference>

</trans-unit>
<trans-unit id='3'>
<source xml:lang='enm'>And the feld ful of folk</source>
<target xml:lang='en'>And the field full of folk</target>
<sup:Reference Type='strophe'>2-a</sup:Reference>
</trans-unit>

<trans-unit id='4'>
<source xml:lang='enm'>I shal yow faire shewe.</source>
<target xml:lang='en'>I fairly will show.</target>
<sup:Reference Type='strophe'>2-b</sup:Reference>
</trans-unit>
</group>

</file>
</xliff>
```

The non-XLIFF elements used in the example above would be defined as the following:

```
<xsd:schema targetNamespace="XLFSup-v1"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:sup="http://www.ChaucerState.ac.pg/Frm/XLFSup-v1"
 elementFormDefault="qualified" attributeFormDefault="unqualified">
```

```
<xsd:element name="SourceInfo">
<xsd:complexType>

 <xsd:sequence maxOccurs="unbounded">

<xsd:element name="Book" type="xsd:string"/>
<xsd:element name="Author" type="xsd:string"/>
</xsd:sequence>

 </xsd:complexType>
</xsd:element>
<xsd:element name="WorkInfo">
<xsd:complexType>
<xsd:attribute name="Task" type="xsd:string"/>
<xsd:attribute name="Context" type="xsd:string"/>

</xsd:complexType>
</xsd:element>
<xsd:element name="Reference">
<xsd:complexType>
<xsd:simpleContent>
<xsd:extension base="xsd:string">Struct_InLine
<xsd:attribute name="Type" type="xsd:string"/>
</xsd:extension>
</xsd:simpleContent>

</xsd:complexType>
</xsd:element>
</xsd:schema>
```

It is not possible to add non-XLIFF elements in either the `<source>` [25] or `<target>` [25] elements. However, the `<mrk>` [32] element can be used to markup sections of the text with user-defined values assigned to the `mtype` [50] attribute. You can also add non-XLIFF attributes to most of the inline elements used in `<source>` [25] and `<target>` [25] .

Adding Attributes

Attributes of a namespace different than XLIFF can be included in several XLIFF elements.

The following elements allow non-XLIFF attributes: `<alt-trans>` [26], `<bin-source>` [27], `<bin-target>` [28], `<bin-unit>` [27], `<bpt>` [31], `<bx/>` [29], `<ept>` [31], `<ex/>` [30], `<file>` [17], `<g>` [28], `<group>` [23], `<it>` [31], `<mrk>` [32], `<ph>` [30], `<seg-source>` [28], `<source>` [25], `<target>` [25], `<tool>` [65], `<trans-unit>` [24], `<x/>` [29] and `<xliff>` [17].

For instance, the following XLIFF code illustrates how to use attributes from the XHTML vocabulary (in bold) in the `<group>` [23] and `<trans-unit>` [24] XLIFF elements. The example shows how to carry formatting information about an extracted table:

```
<xliff version='1.2'
 xmlns='urn:oasis:names:tc:xliff:document:1.2'
 xmlns:htm='http://www.w3.org/1999/xhtml'>

<file original='table.htm' source-language='en' datatype='html'>
<group restype='table' htm:border='1'
```

```
 htm:cellpadding='5'
 htm:cellspacing='0'
 htm:width='100%'>
<group restype='row'>
<trans-unit id='1' htm:valign='top'
 htm:width='30%'>

<source>Text of row 1 column 1</source>
</trans-unit>
<trans-unit id='2' htm:valign='top'
 htm:width='30%'>
<source>Text of row 1 column 2</source>
</trans-unit>

 </group>

<group restype='row'>
<trans-unit id='3' htm:valign='top'
 htm:width='30%'>
<source>Text of row 2 column 1</source>
</trans-unit>
<trans-unit id='4' htm:valign='top'
 htm:width='30%'>

<source>Text of row 2 column 2</source>
</trans-unit>
</group>
</group>
</file>
</xliff>
```

In each of the XLIFF elements allowing non-XLIFF attributes: there is no specific location where to insert the non-XLIFF attributes, and there is no limit to the number of non-XLIFF attributes that can be used.

Adding Attribute Values

Many attributes in XLIFF offer a list of enumerated values. Some applications may find it necessary to add user-defined values to these lists. XLIFF allows for such extension.

The attributes where the list of values can be extended are the following: `alttranstype` [34], `context-type` [37], `count-type [38]`, `ctype [40]`, `datatype [40]`, `mtype [50]`, `purpose [55]`, `reformat [56]`, `restype [58]`, `size-unit [61]`, `state [63]`, `state-qualifier [63]`, and `unit [67]`.

User-defined values must start with an "`x-`" prefix. There is no specified mechanism to validate individual user-defined values. The XLIFF schema will allow any value starting with "`x-`" in addition to the pre-defined values.

For example, the following excerpt shows how the user-defined value `x-for-engineer` can be utilized in a document:

```
...
<group>
<context-group name='EngineersData'>
<context context-type='x-for-engineers'>Data...</context>

...
```

Validating Documents with Extensions

In order to validate an XLIFF document that contains non-XLIFF parts, you can use the schema validation mechanism: In addition to the namespace declarations, add the `schemaLocation` attribute of the XML Schema-instance namespace to define what schemas to use to validate the document (XLIFF and the non-XLIFF namespaces).

Note: XLIFF 1.2 XML Schemas set the attribute `processContents` to value `"skip"`, so the only validation requirement for non-XLIFF content is to ensure it is well-formed.

```
<xliff version='1.2'
 xmlns='urn:oasis:names:tc:xliff:document:1.2'
xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
 xsi:schemaLocation='

 urn:oasis:names:tc:xliff:document:1.2 xliff-core-1.2.xsd
 http://www.ChaucerState.ac.pg/Frm/XLFSup-v1 XLFSup-v1.xsd'
>
...
</xliff>
```

See http://www.w3.org/XML/Schema for more information on XML Schema and validation.

Embedding XLIFF

XML Namespace provides a convenient mechanism to use XLIFF constructs within another XML vocabulary.

If necessary an XLIFF document, or parts of a document, can be embedded within another XML document. The only requirement for this is on the side of the XML format that includes the XLIFF data. For the document to be valid, the schema of the given document type must include a definition for external elements.

If the including XML format uses XML Schema, it should include an `<any>` element in the definition of the element where the XLIFF data can be inserted. For example, the following XSD excerpt illustrates the case of an element type `dataBlockType` that can contain zero, one or more XLIFF constructs after a mandatory `<type>` element:

```
...
<xsd:complexType name="dataBlockType">

<xsd:sequence>
<xsd:element name="type" type="string" minOccurs="0"/>
<xsd:any namespace="##other" processContents="strict" minOccurs="0" maxOccurs="
</xsd:sequence>
</xsd:complexType>
...
```

The ways of inserting different vocabulary in an XML document using XSD are described in section "Any Element, Any Attribute" in the document "XML Schema Part 0: Primer" available here: http://www.w3.org/TR/xmlschema-0/#any.

Non equivalent translations

Linguistically complete text may have to be broken into a number of `<trans-unit>` [24] elements due to message size constraints or other reasons. In these instances the translator is not providing an equivalent translation for each `<source>` [25] , but rather fitting in the target

language text over a number of `<trans-unit> [24]` `<source>` [25] / `<target>` [25] pair elements to meet the requirements of the target application.

Example:

```
<trans-unit id="t1">

<source>Constrained text for limited</source>
<target>Tekst angielski dla</target>
</trans-unit>
<trans-unit id="t2">
<source>display for English</source>
<target>ograniczonego pola</target>

</trans-unit>
```

In this circumstance the `equiv-trans [43]` attribute for the `<target> [25]` element is used to denote that the translation should not be regarded as a direct translation of the `<source> [25]` element. The attribute is optional, and default value is `"yes"`. The other possible value will be `"no"` to indicate that the translation in <target> [25] for the given `<source> [25]` is not a direct equivalent linguistically of the source language text. The following example demonstrates the use of the `equiv-trans [43]` attribute:

```
<trans-unit id="t1">
<source>Constrained text for limited</source>

<target equiv-trans="no">Tekst angielski dla</target>
</trans-unit>
<trans-unit id="t2">
<source>display for English</source>
<target equiv-trans="no">ograniczonego pola</target>
</trans-unit>
```

Grouping translations across `<trans-unit>` elements

It is inevitable that individual XLIFF `<trans-unit> [24]` elements may not represent a piece of text that can be translated without reference to one or more following `<trans-unit> [24]` elements. The causes for this may be incorrect segmentation or bad document design.

Example:

```
<trans-unit id="t1">
 <source>The German acronym v.</source>
 <target>Niemiecki skrÃ³t v. OT oznacza gÃ³rna; pozycje; silnika.</target>

</trans-unit>
<trans-unit id="t2">
 <source> OT signifies the top dead center position for an engine.</source>
 <target/>
</trans-unit>
```

In these cases the `merged-trans [49]` attribute for the `<group> [23]` element can be used to denote that the individual `<trans-unit> [24]` elements cannot be regarded as a direct translation, but rather need to be treated linguistically as a merged group. This attribute has two possible values: `"yes"` or `"no"`. The default value is `"no"`. A value of `"yes"` indicates that the `<trans-unit> [24]` elements contained within this `<group> [23]` element

are to be treated together for linguistic purposes. All <trans-unit> [24] elements that are encompassed by a <group> [23] element that has its merged-trans [49] attribute set to "yes" normally have their related <target> [25] equiv-trans [43] attribute set to the value of "no". The text of all of the <source> [25] and <target> [25] elements taken together form one linguistic whole. No requirements are made regarding the distribution of the translation in the <target> [25] elements. This will be governed by the requirements of the individual applications. The translated text may be placed within the first <target> [25] element leaving the following <target> [25] elements blank, or distributed among the <target> [25] elements contained within the merged-trans [49] attribute of the <group> [23] element. The following example demonstrates the use of the merged-trans [49] attribute for the <group> [23] element:

```
<group merged-trans="yes">
<trans-unit id="t1">
<source>The German acronym v.</source>

<target equiv-trans="no">Niemiecki skrÃ³t v. OT oznacza gÃ³rna; pozycje; silnika
</trans-unit>
<trans-unit id="t2">
<source> OT signifies the top dead center position for an engine.</source>
<target equiv-trans="no"/>
</trans-unit>

</group>
```

Segmentation

During some operations, such as translation and leveraging, it may be important for the user agent to break down the content of the <source> [25] into smaller runs of text (for example, sentences). These smaller parts of text are called *segments*. The process of breaking down a text into segments is known as *segmentation*. It is important to note that the manipulation / segmentation of trans-unit elements is owned by the "translator" domain, not at the extraction filter domain. This means that segmentation will be performed by the editing tool or possibly an automated segmentation process.

In order to avoid modifying the content of the original <source> [25] element, during segmentation a new element <seg-source> [28] is introduced. The content of the <seg-source> [28] element is the same as the content of the <source> [25] element, but with segmentation markup. The segmentation markup is also transferred to the <target> [25] element as applicable during translation.

Each segment inside the <seg-source> [28] and <target> [25] content is represented using the <mrk> [32] element with attribute mtype [50] set to the value "seg". For example the following <source> [25] element contains three segments. After segmentation the content may look as follows:

```
<source>Richard stepped out of the kitchen hut. He noticed a movement from the c
eye. A monkey had climbed on top of one of the workshop sheds, trying to get in
ventilation shaft.</source>
<seg-source><mrk mtype="seg">Richard stepped out of the kitchen hut.</mrk>
<mrk mtype="seg">He noticed a movement from the corner of his eye.</mrk>

<mrk mtype="seg">A monkey had climbed on top of one of the workshop sheds, tryi
 by the ventilation shaft.</mrk>
</seg-source>
```

Note that it may be advisable for XLIFF processing tools to add any missing opening or closing tags when exporting standalone segments outside the original XLIFF document.

Non-clonable <g> [28] elements introduce a problem for localisation in general and segmentation in particular when the non-clonable <g> [28] elements content spans more than single words or isolated expressions. In this form they represent localisation-unfriendly content and are very likely to cause difficulties during translation. Being able to break a segment inside such an element may be the smallest of the problems that tools would be faced with. A non-clonable <g> [28] element clearly represents a piece of content that must be translated as one piece, and cannot be segmented.

Example: This example shows how content with clonable <g> [28] may be localised:

```
<source>This is a <g> sentence. It has
</g> markup.</source>
```

The translation into "Yoda-English" would be:

```
<target>A <g>sentence </g> this is. Markup <g> it has </g>.</target>
```

In this example the <g> [28] element is clonable, and can be localised correctly. However, in the case where cloning is not possible, the resulting content cannot be correctly localised, and is in fact irrespective of whether segments are introduced here or not.

If matching segments need to be identified between `<seg-source> [28]` and `<target> [25]` , and/or between `<seg-source> [28]` content and corresponding `<alt-trans> [26]` units, the `mid [49]` attribute should be used for this purpose.

Example: This example shows how corresponding segments are referenced between `<seg-source> [28]` and `<target>` elements in a `<trans-unit> [24]` .

```
<trans-unit id= "1">
<source>First sentence.Second sentence.</source>
<seg-source>
<mrk mtype="seg" mid="1">First sentence.</mrk>
<mrk mtype="seg" mid="2">Second sentence.</mrk>

</seg-source>
<target>
<mrk mtype="seg" mid="1">Translated first sentence.</mrk>
<mrk mtype="seg" mid="2">Translated second sentence.</mrk>
</target>
</trans-unit>
```

Example: In the following `<trans-unit> [24]` the `<alt-trans> [26]` represents a 75% fuzzy match for the second segment in the `<seg-source> [28]` . This is indicated by introducing the `mid [49]="2"` attribute on the `<alt-trans> [26]` .

```
<trans-unit id= "2">
<source>First sentence.Second sentence.</source>
<seg-source>
<mrk mtype="seg" mid="1">First sentence.</mrk>
<mrk mtype="seg" mid="2">Second sentence.</mrk>

</seg-source>
<alt-trans mid="2" match-quality="75%">
<source>The second sentence.</source>
<target>The translated second sentence.</target>
</alt-trans>
</trans-unit>
```

Example: An `<alt-trans> [26]` element may also have segmented content:

```
<trans-unit id="3">
<source>The second sentence.</source>

<alt-trans match-quality="50%">
<source>First sentence. Second sentence.</source>
<seg-source>
<mrk mtype="seg" mid="1">First sentence.</mrk>
<mrk mtype="seg" mid="2">Second sentence.</mrk>
</seg-source>

<target>
<mrk mtype="seg" mid="1">Translated first sentence.</mrk>
<mrk mtype="seg" mid="2">Translated second sentence.</mrk>
</target>
</alt-trans>
</trans-unit>
```

Detailed Specifications

XML Declaration

The XML declaration is strongly recommended. It indicates the XML version and sets the defaults for the encoding of the file. For example, the following declaration specifies the document is in ISO 8859-1, the Latin-1 encoding.

```
<?xml version="1.0" encoding="iso-8859-1"?>
```

As in all XML files, the default encoding for an XLIFF file is assumed to be either UTF-8, which is a superset of the 7-bit ASCII character set, or UTF-16, which is UCS-2 with surrogate pairs for code points above U+FFFF. Thus, for these character sets, the encoding declaration is not necessary. Further, all XML parsers support these encodings. If the encoding is in UTF-16 the first character of the file must be the Unicode Byte-Order-Mark, U+FEFF, which indicates the endianness of the file. Other encodings may be desirable and may be generally supported by XML parsers. These must be declared using the encoding declaration. The values to use for the encoding declaration are defined in the [IANA Charsets [76]] listing.

If necessary, you can also specify a namespace for XLIFF. The namespace identifier for this standard is "`urn:oasis:names:tc:xliff:document:1.2`".

A minimal XLIFF document with one entry looks something like this:

```
<?xml version="1.0"?>

<xliff version="1.2">
<file source-language="EN" datatype="plaintext" original="file.ext">
<body>
<trans-unit id="1">
<source>Hello World!</source>
</trans-unit>
</body>
</file>

</xliff>
```

If you need to validate the document, use the schema validation mechanism: In addition to the namespace declarations, add the `schemaLocation` attribute of the XML Schema-instance namespace to define what schema files to use. The same example as above would then look like this:

```
<?xml version="1.0"?>
<xliff version='1.2'
 xmlns='urn:oasis:names:tc:xliff:document:1.2'
 xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
 xsi:schemaLocation='
 urn:oasis:names:tc:xliff:document:1.2 xliff-core-1.2.xsd'>
<file source-language="EN" datatype="plaintext" original="file.ext">

<body>
<trans-unit id="1">
<source>Hello World!</source>
</trans-unit>
</body>
</file>
</xliff>
```

If a document of a previous compatible version of XLIFF is to be validated with the schema of a newer version, the document should use the same mechanism.

For validating documents that include non-XLIFF namespaces see the section Validating Documents with Extensions.

Elements

XLIFF elements can be divided into five main categories: the top-level and header elements, the named group elements, the structural elements, the inline elements, and the delimiter elements. Attributes are shared among them.

Table 1.

Top Level and Header elements	`<xliff>` [17], `<file>` [17], `<header>` [17], `<skl>` [18], `<external-file>` [18], `<internal-file>` [18], `<glossary>` [19], `<reference>` [19], `<phase-group>` [20], `<phase>` [20], `<tool>` [65], `<note>` [19].
Named Group Elements	`<context-group>` [21], `<context>` [22], `<count-group>` [21], `<count>` [21], `<prop-group>` [22], `<prop>` [22].
Structural elements	`<body>` [23], `<group>` [23], `<trans-unit>` [24], `<source>` [25], `<target>` [25], `<bin-unit>` [27], `<bin-source>` [27], `<bin-target>` [28], `<alt-trans>` [26].
Inline elements	`<g>` [28], `<x/>` [29], `<bx/>` [29], `<ex/>` [30], `<bpt>` [31], `<ept>` [31], `<sub>` [32], `<it>` [31], `<ph>` [30].
Delimiter element	`<mrk>` [32].

Top-level and Header Elements

The top-level and header elements are the following:

<xliff>

XLIFF document - The `<xliff>` element encloses all the other elements of the document. The required `version` [68] attribute specifies the version of XLIFF. The optional `xml:lang` [69] attribute is used to specify the language of the content of the document.

Required attributes:

`version` [68].

Optional attributes:

`xml:lang` [69], non-XLIFF attributes

Contents:

One or more `<file>` [17] elements, followed by Zero, one or more non-XLIFF elements.

<file>

File - The `<file>` element corresponds to a single extracted original document. The required `original` [53] attribute specifies the name of the file from which this file content is derived. The required `datatype` [40] attribute specifies the format of the original file; e.g. "html". The required `source-language` [62] attribute specifies the language of the `<source>` [25] content. The optional `target-language` [65] attribute is used to specify the language of the `<target>` [25] content. The optional `tool-id` [66] attribute accepts the id of the `<tool>` [65] used to generate this XLIFF document. The optional `date` [42] attribute is used to specify the creation date of this XLIFF file. The optional `xml:space` [69] attribute is used to specify how white-spaces are to be treated. The optional `category` [35] attribute is used to specify a general category of the content of the file; e.g. "medical". The optional `product-name` [55] attribute is used to specify the name of the product which uses this file. The optional `product-version` [55] and `build-num` [35] attributes are used to specify the revision of the product from which this file comes. The `tool` [20] and `ts` [67] attributes have been deprecated in XLIFF 1.1.

Required attributes:

`original` [53], `source-language [62]` , `datatype` [40].

Optional attributes:

`tool` [20], `tool-id` [66], `date` [42], `xml:space` [69], `ts` [67], `category` [35], `target-language` [65], `product-name` [55], `product-version` [55], `build-num` [35], non-XLIFF attributes

Contents:

Zero or one `<header>` [17] element, followed by One `<body>` [23] element.

<header>

File header - The `<header>` element contains metadata relating to the `<file>` [17] element.

Required attributes:

None.

Optional attributes:

None.

Contents:

zero or one `<skl>` [18] element, followed by zero or one `<phase-group>` [20] element, followed by zero, one or more `<glossary>` [19], `<reference>` [19], `<count-group>` [21], `<prop-group>` [22], `<note>` [19], `<tool>` [65] elements, in any order, followed by Zero, one or more non-XLIFF elements.

While for backward compatibility reasons no order is enforced for the elements before the non-XLIFF elements, the recommended order is the one in which they are listed here.

<skl>

Skeleton file - The `<skl>` element contains the skeleton file or the location of the skeleton file. The skeleton file is a template that can be used in recreating the original file, from the `<source>` [25] content, or the translated file, from the `<target>` [25] content.

Required attributes:

None.

Optional attributes:

None

Contents:

Either exactly one `<internal-file>` [18] or one `<external-file>` [18] element.

<internal-file>

Internal file - The `<internal-file>` element contains the actual file being referenced. It is a child of the `<skl>` [18], `<glossary>` [19], and `<reference>` [19] elements. The format of the file is specified by the optional `form` [45] attribute, which accepts mime-type values. The `crc` [39] attribute accepts a value that can be used to precisely identify and assure the authenticity of the file.

Required attributes:

None.

Optional attributes:

`form` [45], `crc` [39].

Contents:

An embedded file.

<external-file>

External file - The `<external-file>` element specifies the location of the actual file being referenced. The required `href` [45] attribute provides a URI to the file. The `crc` [39] attribute accepts a value that can be used to precisely identify and assure the authenticity of the file. The `uid` [67] attribute allows a unique ID to be assigned to the file.

Required attributes:

`href` [45].

Optional attributes:

`uid` [67], `crc` [39].

Contents:

The `<external-file>` is an empty element, including attributes only.

<glossary>

Glossary - The `<glossary>` element points to or contains a glossary, which can be used in the localization of the file.

Required attributes:

None.

Optional attributes:

None.

Contents:

The glossary description and either exactly one `<internal-file>` [18] or one `<external-file>` [18] element.

<reference>

Reference - The `<reference>` element points to or contains reference material, which can aid in the localization of the file.

Required attributes:

None.

Optional attributes:

None.

Contents:

A description of the reference material and either exactly one `<internal-file>` [18] or one `<external-file>` [18] element.

<note>

Note - The `<note>` element is used to add localization-related comments to the XLIFF document. The content of `<note>` may be instructions from developers about how to handle the `<source>` [25] , comments from the translator about the translation, or any comment from anyone involved in processing the XLIFF file. The optional `xml:lang` [69] attribute specifies the language of the note content. The optional `from` [45] attribute indicates who entered the note. The optional `priority` [54] attribute allows a priority from 1 (high) to 10 (low) to be assigned to the note. The optional `annotates` [34] attribute indicates if the note is a general note or, in the case of a `<trans-unit>` [24] , pertains specifically to the `<source>` [25] or the `<target>` [25] element.

Required attributes:

None.

Optional attributes:

`xml:lang` [69], `from` [45], `priority` [54], `annotates` [34] .

Contents:

Text, no standard elements.

<phase-group>

Phase group - The `<phase-group>` element contains information about the task that has been performed on the file. This phase information is specific to the tools and workflow used in processing the file.

Required attributes:

None.

Optional attributes:

None.

Contents:

One or more `<phase>` [20] elements.

<phase>

Phase information - The `<phase>` element contains metadata about the tasks performed in a particular process. The required `phase-name` [54] attribute uniquely identifies the phase for reference within the `<file>` [17] element. The required `process-name` [54] attribute identifies the kind of process the phase corresponds to; e.g. "proofreading". The optional `company-name` [36] attribute identifies the company performing the task. The optional `tool-id` [66] attribute references the `<tool>` [65] used in performing the task. The optional `date` [42] attribute provides a timestamp indicating when the task was performed. The optional `job-id` [46] attribute allows an ID to be assigned to the job. The optional `contact-name` [37] `contact-email` [37] and `contact-phone` [37] attributes all refer to the person performing the task.

Required attributes:

`phase-name` [54], `process-name` [54].

Optional attributes:

`company-name` [36] `tool` [20] `tool-id` [66] `date` [42] `job-id` [46], `contact-name` [37], `contact-email` [37], `contact-phone` [37].

Contents:

Zero, one or more `<note>` [19] elements.

<tool>

Tool - The `<tool>` element describes the tool that has been used to execute a given task in the document. The required `tool-id` [66] attribute uniquely identifies the tool for reference within the `<file>` [17] element. The required `tool-name` [66] attribute specifies the actual tool name. The optional `tool-version` [66] attribute provides the version of the tool. The optional `tool-company` [65] attribute provides the name of the company that produced the tool.

Required attributes:

`tool-id` [66] , `tool-name` [66] .

Optional attributes:

`tool-version` [66] , `tool-company` [65] , non-XLIFF attributes

Contents:

Zero, one or more non-XLIFF elements.

Named Group Elements

The named group elements are the following:

<count-group>

Count group - The `<count-group>` element holds count elements relating to the level in the tree in which it occurs. Each group for `<count> [21]` elements must be named, allowing different uses for each group. The required `name` [53] attribute uniquely identifies the `<count-group>` within the `<file> [17]` element.

Required attributes:

`name` [53].

Optional attributes:

None.

Contents:

One or more `<count>` [21] elements.

<count>

Count - The `<count>` element contains information about counts. For each `<count>` element the required `count-type` [38] attribute indicates what kind of count the element represents, and the optional `unit [67]` attribute indicates the unit of the count (by default: `word`). A list of values for `count-type` [38] and `unit` [67] is provided. The optional phase-name [54] attribute references the `<phase>` [20] in which the count was produced.

Required attributes:

`count-type` [38].

Optional attributes:

`phase-name [54], unit` [67].

Contents:

Number (the count value).

<context-group>

Context group - The `<context-group>` element holds context elements relating to the level in the tree in which it occurs. Thus context can be set at a `<group> [23]` level, a `<trans-unit> [24]` level, or a `<alt-trans> [26]` level. Each `<context-group>` element may be named, allowing different uses for each group. When the <context-group> is named, these uses can be controlled through the use of XML processing instructions. Because the `<context-group>` element may occur at a very high level, a default context can be established for all `<trans-unit> [24]` elements within a file. This default can be overridden at many subsequent levels. The optional `name [53]` attribute may uniquely identify the `<context-group>` within the `<file> [17]` element. The optional `crc [39]` attribute allows a verification of the data. The optional `purpose [55]` attribute indicates to what use this context information is used; e.g. "match" indicates the context information is for memory lookups.

Required attributes:

None.

Optional attributes:

`crc` [39], `name` [53], `purpose` [55]

Contents:

One or more `<context>` [22] elements.

<context>

Context - The `<context>` element describes the context of a `<source>` [25] within a `<trans-unit>` [24] or a `<alt-trans>` [26]. The purpose of this context information is to allow certain pieces of text to have different translations depending on where they came from. The translation of a piece of text may differ if it is a web form or a dialog or an Oracle form or a Lotus form for example. This information is thus required by a translator when working on the file. Likewise, the information may be used by any tool proposing to automatically leverage the text successfully.

The required `context-type` [37] attribute indicates what the context information is; e.g. "recordtitle" indicates the name of a record in a database. The optional `match-mandatory` [47] attribute indicates that translations of the `<source>` [25] elements within the scope of this context must have the same context. The optional `crc` [39] attribute allows a verification of the data.

Required attributes:

`context-type` [37].

Optional attributes:

`match-mandatory` [47], `crc` [39].

Contents:

Text, no standard elements.

<prop-group>

Property group - The `<prop-group>` element contains `<prop>` [22] elements. Each `<prop-group>` element may be named, allowing different uses for each group. These uses can be controlled through the use of XML processing instructions.

Important: The `<prop-group>` element was DEPRECATED in version 1.1. Instead, use attributes defined in a namespace different from XLIFF. See the Extensibility section for more information.

Required attributes:

None.

Optional attributes:

`name` [53].

Contents:

One or more `<prop>` [22] elements.

<prop>

Property - The `<prop>` element allows the tools to specify non-standard information in the XLIFF document. This information can be used by the tools that have produced the file or that translate the file or that do any other amount of processing specific to the producer.

Important: The <prop> element was DEPRECATED in version 1.1. Instead, use attributes defined in a namespace different from XLIFF. See the Extensibility section for more information.

Required attributes:

prop-type [55].

Optional attributes:

xml:lang [69].

Contents:

Tool-specific data or text, no standard elements.

Structural Elements

The structural elements specify the frame of a XLIFF document as well as contextual and processing information. The <source> [25] element contains the extracted data and, possibly, inline elements.

<body>

File body - The <body> element contains the content from the file.

Required attributes:

None.

Optional attributes:

None.

Contents:

Zero, one or more <group> [23], <trans-unit> [24], <bin-unit> [27] elements, in any order.

<group>

Group - The <group> element specifies a set of elements that should be processed together. For example: all the items of a menu, etc. Note that a <group> element can contain other <group> elements. The <group> element can be used to describe the hierarchy of the file.

The optional id [46] attribute is used to uniquely identify the <group> within the same <file> [17]. The optional datatype [40] attribute specifies the data type of the content of the <group>; e.g. "winres" for Windows resources. The optional xml:space [69] attribute is used to specify how white-spaces are to be treated within the <group>. The optional restype [58], resname [57], extradata [44], help-id [45], menu [48], menu-option [48], menu-name [48], coord [38], font [44], css-style [39], style [64], exstyle [43], and extype [44] attributes describe the resources contained within the <group>. The optional translate attribute provides a default value for all <trans-unit> elements contained within the <group>. The optional reformat [56] attribute specifies whether and which attributes can be modified for the <target> [25] elements of the <group>. The optional maxbytes [47] and minbytes [50] attributes specify the required maximum and minimum number of bytes for the translation units within the <group>. The optional size-unit [61] attribute determines the unit for the optional maxheight [47], minheight [50], maxwidth [48], and minwidth [50] attributes, which limit the size of the resource described by the <group>. The optional charclass [36] attribute restricts all translation units in the scope of the <group> to a subset of characters. The optional merged-trans [49] attribute indicates if the group element contains merged <trans-unit> [24] elements. The optional ts [67] attribute

was DEPRECATED in XLIFF 1.1. Lists of values for the `datatype [40]`, `restype [58]`, and `size-unit [61]` attributes are provided by this specification.

Required attributes:

None.

Optional attributes:

`id` [46], `datatype` [40], `xml:space` [69], `ts` [67], `restype` [58], `resname` [57], `extradata` [44], `help-id` [45], `menu` [48], `menu-option` [48], `menu-name` [48], `coord` [38], `font` [44], `css-style` [39], `style` [64], `exstyle` [43], `extype` [44], `translate` [66], `reformat` [56], `maxbytes` [47], `minbytes` [50], `size-unit` [61], `maxheight` [47], `minheight` [50], `maxwidth` [48], `minwidth` [50], `charclass` [36], `merged-trans` [49], non-XLIFF attributes

Contents:

Zero, one or more `<context-group>` [21] elements, followed by Zero, one or more `<count-group>` [21] elements, followed by Zero, one or more `<prop-group>` [22] elements, followed by Zero, one or more `<note>` [19] elements, followed by Zero, one or more non-XLIFF elements, followed by Zero, one or more `<group>` [23], `<trans-unit>` [24], `<bin-unit>` [27] elements, in any order.

All `<context-group>` [21], `<count-group>` [21], `<prop-group>` [22], `<note>` [19] and non-XLIFF elements pertain to the subsequent elements in the tree but can be overridden within a child element.

<trans-unit>

Translation unit - The `<trans-unit>` elements contains a `<source>` [25], `<target>` [25] and associated elements.

The required `id [46]` attribute is used to uniquely identify the `<trans-unit>` within all <trans-unit> and <bin-unit> [27] elements within the same <file> [17]. The optional `approved [35]` attribute indicates whether the translation has been approved by a reviewer. The optional `translate [66]` attribute indicates whether the `<trans-unit>` is to be translated. The optional `reformat [56]` attribute specifies whether and which attributes can be modified for the `<target> [25]` element of the `<trans-unit>`. The optional `xml:space [69]` attribute is used to specify how white-spaces are to be treated within the `<trans-unit>`. The optional `datatype [40]` attribute specifies the data type of the content of the `<trans-unit>`; e.g. "winres" for Windows resources. The optional `ts [67]` attribute was DEPRECATED in XLIFF 1.1. The optional `phase-name [54]` attribute references the phase that the `<trans-unit>` is in. The optional `restype [58]`, `resname [57]`, `extradata [44]`, `help-id [45]`, `menu [48]`, `menu-option [48]`, `menu-name [48]`, `coord [38]`, `font [44]`, `css-style [39]`, `style [64]`, `exstyle [43]`, and `extype [44]` attributes describe the resource contained within the `<trans-unit>`. The optional `maxbytes [47]` and `minbytes [50]` attributes specify the required maximum and minimum number of bytes for the text inside the `<source> [25]` and `<target> [25]` elements of the `<trans-unit>`. The optional `size-unit [61]` attribute determines the unit for the optional `maxheight [47]`, `minheight [50]`, `maxwidth [48]`, and `minwidth [50]` attributes, which limit the size of the resource described by the `<trans-unit>`. The optional `charclass [36]` attribute restricts all `<source> [25]` and `<target> [25]` text in the scope of the `<trans-unit>` to a subset of characters. Lists of values for the `datatype [40]`, `restype [58]`, and `size-unit [61]` attributes are provided by this specification. During translation the content of the `<source>` [25] element may be duplicated into a `<seg-source>` [28] element, in which additional segmentation related markup is introduced. See the Segmentation section for more information.

À [XLIFF

Required attributes:

`id` [46].

Optional attributes:

`approved` [35] `translate`] [

]

in XLIFF 1.1. The `restype` [58] attribute is DEPRECATED in XLIFF 1.2, since `<target>` will always be of the same `restype` [58] as its parent `<trans-unit>` [24] or `<alt-trans>` [26]. A list of preferred values for the `restype` [58], `state` [63], and `state-qualifier` [63] attributes are provided by this specification.

Required attributes:

None.

Optional attributes:

`state` [63], `state-qualifier` [63], `phase-name` [54], `xml:lang` [69], `ts` [67], `restype` [58], `resname` [57], `coord` [38], `font` [44], `css-style` [39], `style` [64], `exstyle` [43], `equiv-trans` [43], non-XLIFF attributes

Contents:

Text, Zero, one or more of the following elements: `<g>` [28], `<x/>` [29], `<bx/>` [29], `<ex/>` [30], `<bpt>` [31], `<ept>` [31], `<ph>` [30], `<it>` [31], `<mrk>` [32], in any order.

<alt-trans>

Translation match - The `<alt-trans>` element contains possible translations in `<target>` [25] elements along with optional context, notes, etc. The optional `mid` [49] attribute indicates that the `<alt-trans>` applies only to a specific `<mrk` [32] `mtype` [50]`="seg">` segment in the `<seg-source>` [28] element of the `<trans-unit>` [24]. (See the Segmentation section for further details.) The optional `match-quality` [47] attribute provides a value indicating the exactness of the match between the `<source>` [25] of the `<alt-trans>` [26] and that of the `<source>` [25] element of the parent `<trans-unit>` [24]; e.g. "90%". The optional `tool-id` [66] attribute accepts the id of the `<tool>` [20] used to generate this `<alt-trans>`. The optional `crc` [39] attribute allows a verification of the data. The optional `xml:lang` [69] attribute is used to specify the content language of the `<alt-trans>`. The optional `xml:space` [69] attribute is used to specify how white-spaces are to be treated within the `<alt-trans>`. The optional `datatype` [40] attribute specifies the data type of the content of the `<alt-trans>`; e.g. "`winres`" for Windows resources. The optional `restype` [58], `resname` [57], `extradata` [44], `help-id` [45], `menu` [48], `menu-option` [48], `menu-name` [48], `coord` [38], `font` [44], `css-style` [39], `style` [64], `exstyle` [43], and `extype` [44] attributes describe the resource contained within the `<alt-trans>`. The optional `origin` [53] attribute specifies where the alternate translation comes from; e.g. a previous version of the product. The `tool` [20] and `ts` [67] attributes were DEPRECATED in XLIFF 1.1. Multiple `<target>` [25] elements within a single `<alt-trans>` are DEPRECATED in XLIFF 1.2. A list of values for the `datatype` [40] and `restype` [58] attributes are provided by this specification.

Required attributes:

None.

Optional attributes:

`mid` [49], `match-quality` [47], `tool` [20], `tool-id` [66], `crc` [39], `xml:lang` [69], `datatype` [40], `xml:space` [69], `ts` [67], `restype` [58], `resname` [57], `extradata` [44], `help-id` [45], `menu` [48], `menu-option` [48], `menu-name` [48], `coord` [38], `font` [44], `css-style` [39], `style` [64], `exstyle` [43], `extype` [44], `origin` [53], `phase-name` [54], `alttranstype` [34], non-XLIFF attributes

Contents:

Zero or one <source> [25] element, followed by Zero or one <seg-source> [28] element, followed by One <target> [25] element, followed by Zero, one or more <context-group> [21], <prop-group> [22], <note> [19] elements, in any order, followed by Zero, one or more non-XLIFF elements.

All child elements of <alt-trans> pertain to their sibling <target> [25] element. While for backward compatibility reasons no order is enforced for the elements before the non-XLIFF elements, the recommended order is the one in which they are listed here. Although not enforced, it is recommended to adopt the convention that more recent <alt-trans> elements appear before older ones in order to define the order that changes are introduced.

<bin-unit>

Binary unit - The <bin-unit> element contains a binary object that may or may not be translatable. The required id [46] attribute is used to uniquely identify the <bin-unit> within all <trans-unit> [24] and <bin-unit> elements within the same <file> [17] The required mime-type [49] attribute specifies the data type of the binary object based on RFC 1341 [77]. The optional approved [35] attribute indicates whether the translation has been approved by a reviewer. The optional translate [66] attribute indicates whether the <bin-unit> [27] is to be translated. The optional reformat [56] attribute specifies whether and which attributes can be modified for the <bin-target> [28] element of the <bin-unit>. The optional ts [67] attribute was DEPRECATED in XLIFF 1.1. The optional phase-name [54] attribute references the phase that the <bin-unit> is in. The optional restype [58] and resname [57] attributes describe the resource contained within the <bin-unit>. A list of values for the restype [58] attribute is provided by this specification.

Required attributes:

id [46], mime-type [49].

Optional attributes:

approved [35], translate [66], reformat [56], ts [67], phase-name [54], restype [58], resname [57], non-XLIFF attributes

Contents:

One <bin-source> [27] element, followed by Zero or one <bin-target> [28] element, followed by Zero, one or more <context-group> [21], <count-group> [21], <prop-group> [22], <note> [19], <trans-unit> [24] elements, in any order, followed by Zero, one or more non-XLIFF elements.

All child elements of

<bin-target>

Binary target -The <bin-target> element is the container for the translated version of the binary data. The optional mime-type [49] attribute specifies the data type of the binary object based on RFC 1341 [77]. The optional ts [67] attribute was DEPRECATED in XLIFF 1.1. The optional state [63] and state-qualifier [63] attributes indicate in which state the <bin-target> is. The optional phase-name [54] attribute references the phase that the <bin-target> is in. The optional restype [58] and resname [57] attributes describe the resource contained within the <bin-target>. A list of values for the restype [58], state [63], and state-qualifier [63] attributes are provided by this specification.

Required attributes:

None.

Optional attributes:

mime-type [49], ts [67], state [63], phase-name [54], restype [58], resname [57], state-qualifier [63], non-XLIFF attributes

Contents:

One of <internal-file> [18] or <external-file> [18].

<seg-source>

Source text - The <seg-source> element is used to maintain a working copy of the <source> [25] element, where markup such as segmentation can be introduced without affecting the actual <source> [25] element content. The content of the <seg-source> is generally the translatable text, typically divided into segments through the use of <mrk [32] mtype [50]="seg"> elements. See the Segmentation section for more information. As with the <source> [25] element, the optional xml:lang [69] attribute is used to specify the content language of the <seg-source>; this should always match source-language [62] as a child of <trans-unit> [24] but can vary as a child of <alt-trans> [26]. The optional ts [67] attribute was DEPRECATED in XLIFF 1.1.

Required attributes:

None.

Optional attributes:

xml:lang [69], ts [67], non-XLIFF attributes

Contents:

Text, Zero, one or more of the following elements: <g> [28] <x/> [29] <bx/> [29] <ex/> [30], <bpt> [31], <ept> [31], <ph> [30], <it> [31], <mrk> [32], in any order.

Inline Elements

The inline elements are the elements that can appear inside the <source> [25] and <target> [25] elements. They enclose or replace any formatting or control code that is not text, but resides within the text unit.

<g>

Generic group placeholder - The <g> element is used to replace any inline code of the original document that has a beginning and an end, does not overlap other paired inline codes, and can be moved within its parent structural element. The required id [46] attribute is used to reference

the replaced code in the skeleton file. The optional ctype [40] attribute allows you to specify what kind of attribute the placeholder represents; e.g. "bold". The optional ts [67] attribute was DEPRECATED in XLIFF 1.1. The optional clone [36] attribute indicates whether this <g> element may be duplicated. The optional xid [68] attribute references a <trans-unit> [24] or <bin-unit> [27], through its id [46] attribute value, which can contain any translatable text from the replaced code. A list of values for the ctype [40] attribute is available. The optional equiv-text [43] attribute specifies text to substitute in place of the inline tag. A <g> element can contain another <g> element.

Required attributes:

id [46].

Optional attributes:

ctype [40], ts [67], clone [36], xid [68], equiv-text [43], non-XLIFF attributes

Contents:

Text, Zero, one or more of the following elements: <g> [28], <x/> [29], <bx/> [29], <ex/> [30], <bpt> [31], <ept> [31], <ph> [30], <it> [31], <mrk> [32], in any order.

<x/>

Generic placeholder - The <x/> element is used to replace any code of the original document. The required id [46] attribute is used to reference the replaced code in the skeleton file. The optional ctype [40] attribute allows you to specify what kind of attribute the placeholder represents; e.g. "bold". The optional ts [67] attribute was DEPRECATED in XLIFF 1.1. The optional clone [36] attribute indicates whether this <x/> element may be duplicated. The optional xid [68] attribute references a <trans-unit> [24] or <bin-unit> [27], through its id [46] attribute value, which can contain any translatable text from the replaced code. A list of values for the ctype [40] attribute is provided by this specification. The optional equiv-text [43] attribute specifies text to substitute in place of the inline tag.

Required attributes:

id [46].

Optional attributes:

ctype [40], ts [67], clone [36], xid [68], equiv-text [43], non-XLIFF attributes

Contents:

Empty.

<bx/>

Begin paired placeholder - The <bx/> element is used to replace a beginning paired code of the original document. It should be used for paired codes that do not follow XML well-formedness rules (i.e. no overlapping elements). If the paired codes follow that rule, it is strongly recommended that the <g> [28] element is used because it simplifies processing. The <bx/> element should be followed by a matching <ex/> [30] element. These paired elements are related via their rid [61] attributes. If the rid [61] attribute is not present (in a 1.0 document for example), the attribute id [46] is used to match both tags. The required id [46] attribute is used to reference the replaced code in the skeleton file. The optional ctype [40] attribute allows you to specify what kind of attribute the placeholder represents; e.g. "bold". The optional ts [67] attribute was DEPRECATED in XLIFF 1.1. The optional clone [36] attribute indicates whether this <bx/> element may be duplicated. The optional xid [68] attribute references a

<trans-unit> [24] or <bin-unit> [27], through its id [46] attribute value, which can contain any translatable text from the replaced code. A list of values for the ctype [40] attribute is provided by this specification. The optional equiv-text [43] attribute specifies text to substitute in place of the inline tag.

Required attributes:

id [46].

Optional attributes:

rid [61], ctype [40], ts [67], clone [36], xid [68], equiv-text [43], non-XLIFF attributes

Contents:

Empty.

<ex/>

End paired placeholder - The <ex/> element is used to replace an ending paired code of the original document. It should be used for paired codes that do not follow XML well-formedness rules (i.e. no overlapping elements). If the paired codes follow that rule, it is strongly recommended that the <g> [28] element is used because it simplifies processing. The <ex/> element should be preceded by a matching <bx/> [29] element. These paired elements are related via their rid [61] attributes. If the rid [61] attribute is not present (in a 1.0 document for example), the attribute id [46] is used to match both tags. The required id [46] attribute is used to reference the replaced code in the skeleton file. The optional ts [67] attribute was DEPRECATED in XLIFF 1.1. The optional xid [68] attribute references a <trans-unit> [24] or <bin-unit> [27], through its id [46] attribute value, which can contain any translatable text from the replaced code. The optional equiv-text [43] attribute specifies text to substitute in place of the inline tag.

Required attributes:

id [46].

Optional attributes:

rid [61], ts [67], xid [68]., equiv-text [43], non-XLIFF attributes

Contents:

Empty.

<ph>

Placeholder - The <ph> element is used to delimit a sequence of native stand-alone codes in the translation unit. The required id [46] attribute is used to identify the <ph> inline code. The optional ctype [40] attribute allows you to specify what kind of attribute the placeholder represents; e.g. "bold". The optional ts [67] attribute was DEPRECATED in XLIFF 1.1. The optional crc [39] attribute allows a verification of the data. The optional assoc [35] attribute specifies whether this placeholder code is associated with the text prior or after. The optional xid [68] attribute references a <trans-unit> [24] or <bin-unit> [27], through its id [46] attribute value, which can contain any translatable text from the replaced code. A list of values for the ctype [40] attribute is provided by this specification. The optional equiv-text [43] attribute specifies text to substitute in place of the inline tag.

Required attributes:

id [46].

Optional attributes:

`ctype` [40], `ts` [67], `crc` [39], `assoc` [35], `xid` [68]., `equiv-text` [43], non-XLIFF attributes

Contents:

Code data, Zero, one or more `<sub>` [32] elements.

<bpt>

Begin paired tag - The `<bpt>` element is used to delimit the beginning of a paired sequence of native codes. Each `<bpt>` has a corresponding `<ept>` [31] element within the translation unit. These paired elements are related via their `rid` [61] attributes. If the `rid` [61] attribute is not present (in a 1.0 document for example), the attribute `id` [46] is used to match both tags. The required `id` [46] attribute is used to identify the `<bpt>` inline code. The optional `ctype` [40] attribute allows you to specify what kind of attribute the code represents; e.g. "bold". The optional `ts` [67] attribute was DEPRECATED in XLIFF 1.1. The optional `crc` [39] attribute allows a verification of the data. The optional `xid` [68] attribute references a `<trans-unit>` [24] or `<bin-unit>` [27], through its `id` [46] attribute value, which can contain any translatable text from the inline code. A list of values for the `ctype` [40] attribute is provided by this specification. The optional equiv-text [43] attribute specifies text to substitute in place of the inline tag.

Required attributes:

`id` [46].

Optional attributes:

`rid` [61], `ctype` [40], `ts` [67], `crc` [39], `xid` [68], `equiv-text` [43], non-XLIFF attributes

Contents:

Code data, Zero, one or more `<sub>` [32] elements.

<ept>

End paired tag - The `<ept>` element is used to delimit the end of a paired sequence of native codes. Each `<ept>` has a corresponding `<bpt>` [31] element within the translation unit. These paired elements are related via their `rid` [61] attributes. If the `rid` [61] attribute is not present (in a 1.0 document for example), the attribute `id` [46] is used to match both tags. The required `id` [46] attribute is used to identify the `<ept>` inline code. The optional `ts` [67] attribute was DEPRECATED in XLIFF 1.1. The optional `crc` [39] attribute allows a verification of the data. The optional `xid` [68] attribute references a `<trans-unit>` [24] or `<bin-unit>` [27], through its `id` [46] attribute value, which can contain any translatable text from the inline code. The optional equiv-text [43] attribute specifies text to substitute in place of the inline tag.

Required attributes:

`id` [46].

Optional attributes:

`rid` [61], `ts` [67], `crc` [39], `xid` [68], `equiv-text` [43], non-XLIFF attributes

Contents:

Code data, Zero, one or more `<sub>` [32] elements.

<it>

Isolated tag - The <it> element is used to delimit a beginning/ending sequence of native codes that does not have its corresponding ending/beginning within the translation unit. The required id [46] attribute is used to identify the <it> inline code. The required pos [54] attribute specifies whether this is the begin or end code. The optional ctype [40] attribute allows you to specify what kind of attribute the code represents; e.g. "bold". The optional ts [67] attribute was DEPRECATED in XLIFF 1.1. The optional crc [39] attribute allows a verification of the data. The optional xid [68] attribute references a <trans-unit> [24] or <bin-unit> [27] , through its id [46] attribute value, which can contain any translatable text from the inline code. A list of values for the ctype [40] attribute is provided by this specification. The optional equiv-text [43] attribute specifies text to substitute in place of the inline tag.

Required attributes:

id [46], pos [54] .

Optional attributes:

rid [61], ctype [40], ts [67], crc [39], xid [68], equiv-text [43], non-XLIFF attributes

Contents:

Code data, Zero, one or more <sub> [32] elements.

<sub>

Sub-flow - The <sub> element is used to delimit sub-flow text inside a sequence of native code, for example: the definition of a footnote or the text of a title attribute in a HTML <a> element. The optional datatype [40] attribute specifies the data type of the content of the <sub>; e.g. "html". The optional ctype [40] attribute allows you to specify what kind of attribute the code represents. The optional xid [68] attribute references a <trans-unit> [24] or <bin-unit> [27] , through its id [46] attribute value, which can contain any translatable text from the inline code. Lists of values for the ctype [40] and datatype [40] attributes are provided by this specification.

Required attributes:

None.

Optional attributes:

datatype [40], ctype [40], xid [68] .

Contents:

Text, Zero, one or more of the following elements: <g> [28], <x/> [29], <bx/> [29], <ex/> [30], <bpt> [31], <ept> [31], <ph> [30], <it> [31], <mrk> [32] , in any order.

Delimiter Element

XLIFF defines an additional element to support various types of text processing. This element is usually not generated by the extraction module and is ignored most of the time during merging, but it can be very powerful with tools such as Machine Translation, glossary handling, quality assurance, etc.

<mrk>

Marker - The <mrk> element delimits a section of text that has special meaning, such as a terminological unit, a proper name, an item that should not be modified, etc. It can be used for various processing tasks. For example: to indicate to a Machine Translation tool proper names that should not be translated; for terminology verification; to mark suspect expressions after a grammar checking.

The <mrk> element is usually not generated by the extraction tool and it is not part of the tags used to merge the XLIFF file back into its original format. The required mtype [50] attribute specifies what is being delimited; e.g. "abbrev" for an abbreviation. The optional ts [67] attribute was DEPRECATED in XLIFF 1.1. The optional comment [36] attribute allow a free-form comment to be entered. A list of values for the mtype [50] attribute is provided by this specification. The <mrk> element can be used to delimit segments as described in the Segmentation section.

Required attributes:s

mtype [50].

Optional attributes:

mid [49], ts [67], comment [36], non-XLIFF attributes

Contents:

Text, Zero, one or more of the following elements: <g> [28] <x/> [29] <bx/> [29] , <ex/> [30] <bpt> [31] <ept> [31] , <ph> [30] <it> [31] <mrk> [32] , in any order.

Attributes

This section lists the various attributes used in the XLIFF elements. An attribute is never specified more than once for each element. Along with some of the attributes is the list of their possible values.

Table 2.

XLIFF attributes	alttranstype [34] annotates [34], approved [35] assoc [35] build-num [35] ctype [40] category [35] charclass [36] comment [36] company-name [36] contact-email [37] , contact-name [37] , contact-phone [37] , context-type [37] , coord [38] count-type [38] crc [39] css-style [39] datatype [40] date [42] exstyle [43] equiv-text [43] equiv-trans [43] extradata [44] extype [44] font [44] form [45] from [45] , help-id [45] href [45] id [46] job-id [46] , match-mandatory [47] match-quality [47] , maxheight [47] maxbytes [47] maxwidth [48] menu [48] menu-name [48] menu-option [48] mid [49] merged-trans [49] mime-type [49] minheight [50] minbytes [50] minwidth [50] mtype [50] name [53] original [53] phase-name [54] pos [54] priority [54] process-name [54] product-name [55] product-version [55] prop-type [55] purpose [55] , reformat [56] resname [57]

	restype [58], rid [61], size-unit [61], source-language [62], state [63], state-qualifier [63], style [64], tool [20], tool-company [65], tool-id [66], tool-name [66], tool-version [66], target-language [65], translate [66], ts [67], uid [67], unit [67], version [68], xid [68].
XML namespace attributes	xml:lang [69], xml:space [69].

XLIFF Attributes

alttranstype

Resource type - Indicates the type of translation within the containing alt-trans element.

Value description:

The pre-defined values are defined in the table below.

Table 3.

Value	Description
proposal	Represents a translation proposal from a translation memory or other resource.
previous-version	Represents a previous version of the target element.
rejected	Represents a rejected version of the target element.
reference	Represents a translation to be used for reference purposes only, for example from a related product or a different language.
accepted	Represents a proposed translation that was used for the translation of the trans-unit, possibly modified.

In addition, user-defined values can be used with this attribute. A user-defined value must start with an "x-" prefix.

Default value:

proposal.

Used in:

<alt-trans> [26]

annotates

Annotates - Indicates if a <note> [19] element pertains to the <source> [25] or the <target> [25], or neither in particular.

Value description:

source, target, or general.

Default value:

`general`.

Used in:

`<note>` [19] .

approved

Approved - Indicates whether a translation is final or has passed its final review.

Value description:

Boolean: `yes` or `no`.

Default value:

`no`.

Used in:

`<trans-unit>` [24], `<bin-unit>` [27].

assoc

Association - Indicates the association of a `<ph>` [30] with the text prior or after the inline element.

Value description:

`preceding` (the element is associated with the text preceding the element), `following` (the element is associated with the text following the element), and `both` (the element is associated with the text on both sides).

Default value:

Undefined.

Used in:

`<ph>` [30].

build-num

Build number - The build number of the version of the product or application the localizable material is for. For example: `build-num="12"` for the 12th build of the new version of a product.

Value description:

Alpha-numeric.

Default value:

Undefined.

Used in:

`<file>` [17].

category

Category - This provides information on the subject of what is being translated. For example: `category="medical"` for files from a medical related product.

Value description:

Text.

Default value:

Undefined.

Used in:

`<file>` [17].

charclass

Character class - This indicates that a translation is restricted to a subset of characters (i.e. ASCII only, Katakana only, uppercase only, etc.). A blank value indicates there is no limitation.

Value description:

Text.

Default value:

Undefined.

Used in:

`<group>` [23], `<trans-unit>` [24].

clone

Clone - This indicates that a copy of the given inline element can be made and placed multiple times in the `<target>` [25] . This is useful for codes such as bold which may require duplication after localization of a segment.

Value description:

Boolean: `yes` or `no`.

Default value:

`yes`.

Used in:

`<g>` [28], `<x/>` [29], `<bx/>` [29] .

comment

Comment - A comment in a tag.

Value description:

Alpha-numeric.

Default value:

Undefined.

Used in:

`<mrk>` [32].

company-name

Company name - The name of the company that has performed a task.

Value description:

Text.

Default value:

Undefined.

Used in:

`<phase>` [20].

contact-email

Contact email - The contact email of the `contact-name [37]` person.

Value description:

Text.

Default value:

Undefined.

Used in:

`<phase>` [20].

contact-name

Contact name - The name of the person that has performed a task in a phase.

Value description:

Text.

Default value:

Undefined.

Used in:

`<phase>` [20].

contact-phone

Contact phone - The phone number of the `contact-name [37]` person.

Value description:

Text.

Default value:

Undefined.

Used in:

`<phase>` [20].

context-type

Context type - The `context-type` attribute specifies the context and the type of resource or style of the data of a given element. For example, to define if it is a label, or a menu item in the case of resource-type data, or the style in the case of document-related data.

Value description:

The pre-defined values are defined in the table below.

Table 4.

Value	Description
`database`	Indicates a database content.
`element`	Indicates the content of an element within an XML document.
`elementtitle`	Indicates the name of an element within an XML document.
`linenumber`	Indicates the line number from the sourcefile (see context-type="sourcefile") where the <source> is found.
`numparams`	Indicates a the number of parameters contained within the <source>.
`paramnotes`	Indicates notes pertaining to the parameters in the <source>.
`record`	Indicates the content of a record within a database.
`recordtitle`	Indicates the name of a record within a database.
`sourcefile`	Indicates the original source file in the case that multiple files are merged to form the original file from which the XLIFF file is created. This differs from the original <file> attribute in that this sourcefile is one of many that make up that file.

In addition, user-defined values can be used with this attribute. A user-defined value must start with an "`x-`" prefix.

Default value:

Undefined.

Used in:

`<context>` [22].

coord

Coordinates - The `coord` attribute specifies the x, y, cx and cy coordinates of the text for a given element. The cx and cy values must represent the width and the height (as in Windows resources). The extraction and merging tools must make the right conversion if the original format uses a top-left/bottom-right coordinate system.

Value description:

Four decimal (possibly negative) values, in the order: x, y, cx and cy, separated by semi-colons. Null values may be entered as "#"; (e.g. `coord="#;#;183;272"`).

Default value:

Undefined.

Used in:

`<group>` [23], `<trans-unit>` [24], `<target>` [25], `<alt-trans>` [26].

count-type

Count type - The `count-type` attribute specifies the purpose of the `<count> [21]` element. For example: `count-type="total"` for the total count of words in the current scope.

Value description:

The pre-defined values are defined in the table below.

Table 5.

Value	Description
`num-usages`	Indicates the count units are items that are used X times in a certain context; example: this is a reusable text unit which is used 42 times in other texts.
`repetition`	Indicates the count units are translation units existing already in the same document.
`total`	Indicates a total count.

In addition, user-defined values can be used with this attribute. A user-defined value must start with an "`x-`" prefix.

In addition, the `count-type` attribute can take any value defined for `datatype [40]`, `restype [58]`, or `state [63]`.

Default value:

None.

Used in:

`<count>` [21].

crc

Cyclic redundancy checking - A private value used to verify data as it is returned to the producer. The generation and verification of this number is tool-specific.

Value description:

Number (possibly not decimal).

Default value:

None.

Used in:

`<internal-file>` [18], `<external-file>` [18], `<context-group>` [21], `<context>` [22], `<alt-trans>` [26], `<bpt>` [31], `<ept>` [31], `<it>` [31], `<ph>` [30].

css-style

Cascading style-sheet style- The `css-style` attribute allows any valid CSS statement [76] to be specified.

Value description:

Text, the value is subject to CSS syntax rules.

Default value:

Undefined.

Used in:

<group> [23], <trans-unit> [24], <target> [25], <alt-trans> [26].

ctype

Content type - The ctype attribute specifies the type of code that is represented by the inline element; e.g. ctype="bold" means that the code represents a bolding code.

Value description for the ctype attribute of the <x/> [29] and <ph> [30] elements:

The pre-defined values are defined in the table below.

Table 6.

Value	Description
image	Indicates a inline image.
pb	Indicates a page break.
lb	Indicates a line break.

In addition, user-defined values can be used with this attribute. A user-defined value must start with an "x-" prefix.

Value description for the ctype attribute of other elements:

The pre-defined values are defined in the table below.

Table 7.

Value	Description
bold	Indicates a run of bolded text.
italic	Indicates a run of text in italics.
underlined	Indicates a run of underlined text.
link	Indicates a run of hyper-text.

In addition, user-defined values can be used with this attribute. A user-defined value must start with an "x-" prefix.

Default value:

Undefined.

Used in:

<g> [28], <x/> [29], <bx/> [29], <bpt> [31]

Table 8.

Value	Description
asp	Indicates Active Server Page data.
c	Indicates C source file data.
cdf	Indicates Channel Definition Format (CDF) data.
cfm	Indicates ColdFusion data.
cpp	Indicates C++ source file data.
csharp	Indicates C-Sharp data.
cstring	Indicates strings from C, ASM, and driver files data.
csv	Indicates comma-separated values data.
database	Indicates database data.
documentfooter	Indicates portions of document that follows data and contains metadata.
documentheader	Indicates portions of document that precedes data and contains metadata.
filedialog	Indicates data from standard UI file operations dialogs (e.g., Open, Save, Save As, Export, Import).
form	Indicates standard user input screen data.
html	Indicates HyperText Markup Language (HTML) data - document instance.
htmlbody	Indicates content within an HTML document's <body> element.
ini	Indicates Windows INI file data.
interleaf	Indicates Interleaf data.
javaclass	Indicates Java source file data (extension '.java').
javapropertyresourcebundle	Indicates Java property resource bundle data.
javalistresourcebundle	Indicates Java list resource bundle data.
javascript	Indicates JavaScript source file data.
jscript	Indicates JScript source file data.
layout	Indicates information relating to formatting.
lisp	Indicates LISP source file data.
margin	Indicates information relating to margin formats.
menufile	Indicates a file containing menu.
messagefile	Indicates numerically identified string table.
mif	Indicates Maker Interchange Format (MIF) data.
mimetype	Indicates that the datatype attribute value is a MIME Type value and is defined in the mime-type attribute.
mo	Indicates GNU Machine Object data.
msglib	Indicates Message Librarian strings created by Novell's Message Librarian Tool.
pagefooter	Indicates information to be displayed at the bottom of each page of a document.

pageheader	Indicates information to be displayed at the top of each page of a document.
parameters	Indicates a list of property values (e.g., settings within INI files or preferences dialog).
pascal	Indicates Pascal source file data.
php	Indicates Hypertext Preprocessor data.
plaintext	Indicates plain text file (no formatting other than, possibly, wrapping).
po	Indicates GNU Portable Object file.
report	Indicates dynamically generated user defined document. e.g. Oracle Report, Crystal Report, etc.
resources	Indicates Windows .NET binary resources.
resx	Indicates Windows .NET Resources.
rtf	Indicates Rich Text Format (RTF) data.
sgml	Indicates Standard Generalized Markup Language (SGML) data - document instance.
sgmldtd	Indicates Standard Generalized Markup Language (SGML) data - Document Type Definition (DTD).
svg	Indicates Scalable Vector Graphic (SVG) data.
vbscript	Indicates VisualBasic Script source file.
warning	Indicates warning message.
winres	Indicates Windows (Win32) resources (i.e. resources extracted from an RC script, a message file, or a compiled file).
xhtml	Indicates Extensible HyperText Markup Language (XHTML) data - document instance.
xml	Indicates Extensible Markup Language (XML) data - document instance.
xmldtd	Indicates Extensible Markup Language (XML) data - Document Type Definition (DTD).
xsl	Indicates Extensible Stylesheet Language (XSL) data.
xul	Indicates XUL elements.

In addition, user-defined values can be used with this attribute. A user-defined value must start with an "x-" prefix.

Default value:

Empty string.

Used in:

<file> [17], <group> [23], <trans-unit> [24], <alt-trans> [26], <sub> [32].

date

Date - The date attribute indicates when a given element was created or modified.

Value description:

Date in [ISO 8601 [77]] Format. The recommended pattern to use is: `CCYY-MM-DDThh:mm:ssZ` Where: `CCYY` is the year (4 digits), `MM` is the month (2 digits), `DD` is the day (2 digits), `hh` is the hours (2 digits), `mm` is the minutes (2 digits), `ss` is the second (2 digits), and `Z` indicates the time is UTC time. For example:

```
 date="2002-01-25T21:06:00Z"
is January 25, 2002 at 9:06pm GMT
is January 25, 2002 at 2:06pm US Mountain Time
is January 26, 2002 at 6:06am Japan time
```

Default value:

Undefined.

Used in:

`<file>` [17], `<phase>` [20].

equiv-text

equiv-text - Indicates the equivalent text to substitute in place of an inline tag. It is useful for inserting whitespace or other content in place of markup to facilitate consistent word counting. The equiv-text attribute is also useful for ensuring consistent round trip conversion between native resource formats and XLIFF content, for example the resource string "`F&ile`" converts to the following XLIFF: "`F<x id='1' ctype='x-akey' equiv-text=''/>ile`" to preserve the underlying translatable content.

Value description:

Text

Default value:

Undefined.

Used in:

`<g>` [28], `<x/>` [29], `<bx/>` [29], `<ex/>` [30], `<bpt>` [31], `<ept>` [31], `<ph>` [30], `<it>` [31].

equiv-trans

equiv-trans - Indicates if the target language translation is a direct equivalent of the source text.

Value description:

`yes`, or `no`.

Default value:

`yes`.

Used in:

`<target>` [25]

exstyle

Extended style - The `exstyle` attribute stores the extended style of a control. For example, in Windows resources it corresponds to the EXSTYLE statement.

Value description:

Text.

Default value:

Undefined.

Used in:

`<group>` [23], `<trans-unit>` [24], `<target>` [25], `<alt-trans>` [26].

extradata

Extra data - The `extradata` attribute stores the extra data properties of an item.

Value description:

Text.

Default value:

Undefined.

Used in:

`<group>` [23], `<trans-unit>` [24], `<alt-trans>` [26].

extype

Extended type -The `extype` attribute stores the extra type properties of an item.

Value description:

Text.

Default value:

Undefined.

Used in:

`<group>` [23], `<trans-unit>` [24], `<alt-trans>` [26].

font

Font - The `font` attribute specifies the font name, size, and weight of the text for a given element. The font attribute would generally be used for resource-type data: change of font in document-type data can be marked with the `<g>` [28] element.

Value description:

Name of the font, its size, its weight, its style and its encoding separated by semi-colons. Only the name of the font is required.

Default value:

Undefined.

Used in:

`<group>` [23], `<trans-unit>` [24], `<target>` [25], `<alt-trans>` [26].

form

Format - Describes the type of format used in an `<internal-file>` [18] element. For example: `form="text"` indicates a plain text format internal file.

Value description:

The value can be either `text` (for plain text data), `base64` (for data coded in base64 format), or one of values available from the [RFC 1341 [77]] document: the MIME specification.

Default value:

`text`.

Used in:

`<internal-file>` [18] .

from

From - Indicates the author of a `<note> [19]` element. For example: `from="reviewer"` indicates a note from a reviewer.

Value description:

Text.

Default value:

Undefined.

Used in:

`<note>` [19].

help-id

Help ID -The `help-id` attribute stores the help identifier of an item. For example, in Windows resources it corresponds to the Help ID parameter of a control.

Value description:

Number.

Default value:

Undefined.

Used in:

`<group>` [23], `<trans-unit>` [24], `<alt-trans>` [26].

href

Hypertext reference - The location of the file or the URL for an `<external-file>` [18] element. For example: `href="file:///C:/MyFolder/MyProject/MyFile.htm"` indicates a file on a local drive.

Value description:

Text.

Default value:

Undefined.

Used in:

`<external-file>` [18] .

id

Identifier - The `id` attribute is used in many elements as a reference to the original corresponding code data or format for the given element. The value of the `id` element is determined by the tool creating the XLIFF document. It may or may not be a resource identifier. The identifier of a resource should, at least, be stored in the `resname [57]` attribute.

For example:

```
<trans-unit id="34" resname="IDD_ABOUT_DLG" restype="dialog"
coord="0;0;235;100" font="MS Sans Serif;8" style="0x0932239">

<source>About Dialog</source>
</trans-unit>


<trans-unit id="IDD_ABOUT_DLG" resname="IDD_ABOUT_DLG"
 restype="dialog" coord="0;0;235;100" font="MS Sans Serif;8"
 style="0x0932239">
<source>About Dialog</source>

</trans-unit>
```

Value description:

Text. Note that, while allowed, spaces are usually not used in identifiers.

Default value:

Undefined.

Used in:

`<group>` [23], `<trans-unit>` [24], `<bin-unit>` [27], `<g>` [28], `<x/>` [29], `<bx/>` [29], `<ex/>` [30], `<bpt>` [31], `<ept>` [31], `<it>` [31], `<ph>` [30] .

job-id

Job ID - The identifier given to the localization job. This is determined by the entity creating the phase element at the time of processing the file.

Value description:

Text.

Default value:

Undefined.

Used in:

`<phase>` [20].

match-mandatory

Match mandatory -Indicates that any `<alt-trans> [26]` element of the parent `<trans-unit> [24]` must have the same `<context>` [22] as the `<trans-unit> [24]` .

Value description:

Boolean: `yes` or `no`.

Default value:

`no`.

Used in:

`<context>` [22].

match-quality

Match quality - The match quality of the `<alt-trans> [26]` element is tool specific and can be a score expressed in percentage or an arbitrary value (e.g. `match-quality="high"`).

Value description:

Text.

Default value:

Undefined.

Used in:

`<alt-trans>` [26] .

maxheight

Maximum height - The maximum height for the `<target> [25]` of a `<trans-unit> [24]` . This could be interpreted as lines, pixels, or any other relevant unit. The unit is determined by the `size-unit` [61] attribute, which defaults to `pixel`.

Value description:

Number.

Default value:

Undefined.

Used in:

`<group> [23]` , `<trans-unit>` [24].

maxbytes

Maximum bytes - The maximum number of bytes for the `<target> [25]` of a `<trans-unit> [24]` . The verification of whether the relevant text respects this requirement must be done using the encoding and line-break type of the final target environment.

Value description:

Number.

Default value:

Undefined.

Used in:

`<group> [23]` , `<trans-unit>` [24].

maxwidth

Maximum width - The maximum width for the `<target> [25]` of a `<trans-unit>` [24]. This could be interpreted as lines, pixels, or any other relevant unit. The unit is determined by the `size-unit` [61] attribute, which defaults to `pixel`.

Value description:

Number.

Default value:

Undefined.

Used in:

`<group> [23]` , `<trans-unit>` [24].

menu

Menu - The `menu` attribute stores the menu property of an item.

Value description:

Text.

Default value:

Undefined.

Used in:

`<group>` [23], `<trans-unit>` [24], `<alt-trans>` [26].

menu-name

Menu name - The `menu-name` attribute stores the menu name of a control.

Value description:

Text.

Default value:

Undefined.

Used in:

`<group>` [23], `<trans-unit>` [24], `<alt-trans>` [26].

menu-option

Menu option - The `menu-option` attribute stores the option data of a control.

Value description:

Text.

Default value:

Undefined.

Used in:

<group> [23], <trans-unit> [24], <alt-trans> [26].

mid

Marker ID - Identifier for an <mrk> [32] element. When used with in combination with mtype [50]="seg" the value of this attribute is used to reference segments between the <seg-source> [28] and <target> [25] of a <trans-unit> [24]. When used in <alt-trans> [26] this attribute indicates that the entire <alt-trans> [26] element references a particular <mrk mtype [50] ="seg">segment in the <seg-source> [28] (and <target> [25]) element. See the Segmentation section for further details.

Value description:

Text.

Default value:

Undefined.

Used in:

<mrk> [32], <alt-trans> [26]

merged-trans

merged-trans - Indicates if the group element contains merged trans-unit elements.

Value description:

yes, or no.

Default value:

no.

Used in:

<group> [23]

mime-type

Mime type -Indicates the type of a binary object. These roughly correspond to the content-type of RFC 1341 [77] , the MIME specification; e.g. mime-type="image/jpeg" indicates the binary object is an image file of JPEG format. This is important in determining how to edit the binary object.

Value description:

Text. A list of preferred values is available from the [RFC 1341 [77]] document: the MIME specification.

Default value:

Undefined.

Used in:

<bin-unit> [27], <bin-target> [28].

minheight

Minimum height - The minimum height for the <target> [25] of a <trans-unit> [24]. This could be interpreted as lines, pixels, or any other relevant unit. The unit is determined by the size-unit [61] attribute, which defaults to pixel.

Value description:

Number.

Default value:

Undefined.

Used in:

<group> [23] , <trans-unit> [24].

minbytes

Minimum bytes - The minimum number of bytes for the <target> [25] of a <trans-unit> [24] . The verification of whether the relevant text respects this requirement must be done using the encoding and line-break type of the final target environment.

Value description:

Number.

Default value:

Undefined.

Used in:

<group> [23] , <trans-unit> [24].

minwidth

Minimum width - The minimum width for the <target> [25] of a <trans-unit> [24]. This could be interpreted as lines, pixels, or any other relevant unit. The unit is determined by the size-unit [61] attribute, which defaults to pixel.

Value description:

Number.

Default value:

Undefined.

Used in:

<group> [23] , <trans-unit> [24].

mtype

Marker type -The mtype attribute specifies what a <mrk> [32] element is defining within the content of a <source> [25] or <target> [25] element.

Value description:

The pre-defined values are defined in the table below.

Table 9.

Value	Description
`abbrev`	Indicates the marked text is an abbreviation.
`abbreviated-form`	ISO-12620 2.1.8: A term resulting from the omission of any part of the full term while designating the same concept.
`abbreviation`	ISO-12620 2.1.8.1: An abbreviated form of a simple term resulting from the omission of some of its letters (e.g. 'adj.' for 'adjective').
`acronym`	ISO-12620 2.1.8.4: An abbreviated form of a term made up of letters from the full form of a multiword term strung together into a sequence pronounced only syllabically (e.g. 'radar' for 'radio detecting and ranging').
`appellation`	ISO-12620: A proper-name term, such as the name of an agency or other proper entity.
`collocation`	ISO-12620 2.1.18.1: A recurrent word combination characterized by cohesion in that the components of the collocation must co-occur within an utterance or series of utterances, even though they do not necessarily have to maintain immediate proximity to one another.
`common-name`	ISO-12620 2.1.5: A synonym for an international scientific term that is used in general discourse in a given language.
`datetime`	Indicates the marked text is a date and/or time.
`equation`	ISO-12620 2.1.15: An expression used to represent a concept based on a statement that two mathematical expressions are, for instance, equal as identified by the equal sign (=), or assigned to one another by a similar sign.
`expanded-form`	ISO-12620 2.1.7: The complete representation of a term for which there is an abbreviated form.
`formula`	ISO-12620 2.1.14: Figures, symbols or the like used to express a concept briefly, such as a mathematical or chemical formula.
`head-term`	ISO-12620 2.1.1: The concept designation that has been chosen to head a terminological record.
`initialism`	ISO-12620 2.1.8.3: An abbreviated form of a term consisting of some of the initial letters of the words making up a multiword term or the term elements making up a compound term when these letters are pronounced individually (e.g. 'BSE' for 'bovine spongiform encephalopathy').
`international-scientific-term`	ISO-12620 2.1.4: A term that is part of an international scientific nomenclature as adopted by an appropriate scientific body.

internationalism	ISO-12620 2.1.6: A term that has the same or nearly identical orthographic or phonemic form in many languages.
logical-expression	ISO-12620 2.1.16: An expression used to represent a concept based on mathematical or logical relations, such as statements of inequality, set relationships, Boolean operations, and the like.
materials-management-unit	ISO-12620 2.1.17: A unit to track object.
name	Indicates the marked text is a name.
near-synonym	ISO-12620 2.1.3: A term that represents the same or a very similar concept as another term in the same language, but for which interchangeability is limited to some contexts and inapplicable in others.
part-number	ISO-12620 2.1.17.2: A unique alphanumeric designation assigned to an object in a manufacturing system.
phrase	Indicates the marked text is a phrase.
phraseological-unit	ISO-12620 2.1.18: Any group of two or more words that form a unit, the meaning of which frequently cannot be deduced based on the combined sense of the words making up the phrase.
protected	Indicates the marked text should not be translated.
romanized-form	ISO-12620 2.1.12: A form of a term resulting from an operation whereby non-Latin writing systems are converted to the Latin alphabet.
seg	Indicates that the marked text represents a segment.
set-phrase	ISO-12620 2.1.18.2: A fixed, lexicalized phrase.
short-form	ISO-12620 2.1.8.2: A variant of a multiword term that includes fewer words than the full form of the term (e.g. 'Group of Twenty-four' for 'Intergovernmental Group of Twenty-four on International Monetary Affairs').
sku	ISO-12620 2.1.17.1: Stock keeping unit, an inventory item identified by a unique alphanumeric designation assigned to an object in an inventory control system.
standard-text	ISO-12620 2.1.19: A fixed chunk of recurring text.
symbol	ISO-12620 2.1.13: A designation of a concept by letters, numerals, pictograms or any combination thereof.
synonym	ISO-12620 2.1.2: Any term that represents the same or a very similar concept as the main entry term in a term entry.
synonymous-phrase	ISO-12620 2.1.18.3: Phraseological unit in a language that expresses the same semantic content as another phrase in that same language.
term	Indicates the marked text is a term.

`transcribed-form`	ISO-12620 2.1.11: A form of a term resulting from an operation whereby the characters of one writing system are represented by characters from another writing system, taking into account the pronunciation of the characters converted.
`transliterated-form`	ISO-12620 2.1.10: A form of a term resulting from an operation whereby the characters of an alphabetic writing system are represented by characters from another alphabetic writing system.
`truncated-term`	ISO-12620 2.1.8.5: An abbreviated form of a term resulting from the omission of one or more term elements or syllables (e.g. 'flu' for 'influenza').
`variant`	ISO-12620 2.1.9: One of the alternate forms of a term.

In addition, user-defined values can be used with this attribute. A user-defined value must start with an "`x-`" prefix.

Default value:

Undefined.

Used in:

`<mrk>` [32].

name

Name - The `name` attribute specifies the user-defined name of a named group element. This is used for identification purposes only and is not referenced with the file, unless by a processing instruction.

Value description:

Text.

Default value:

Undefined.

Used in:

`<prop-group>` [22], `<context-group>` [21], `<count-group>` [21].

origin

Translation Match Origin - The `origin` attribute specifies where a translation match came from; for example, from a previous version of the same product, a different product, a shared translation memory, etc.

Value description:

Text.

Default value:

Undefined.

Used in:

`<alt-trans>` [26] .

original

Original file - The original attribute specifies the name of the original file from which the contents of a <file> [17] element has been extracted.

Value description:

Text.

Default value:

Undefined.

Used in:

`<file>` [17].

phase-name

Phase Name - The `phase-name` attribute provides a unique name for a `<phase>` [20] element. It is used in other elements in the file to refer to the given `<phase>` [20] element.

Value description:

Text.

Default value:

Undefined.

Used in:

`<count>` [21], `<phase>` [20], `<trans-unit>` [24], `<target>` [25], `<bin-unit>` [27], `<bin-target>` [28], `<alt-trans>` [26] .

pos

Position - Indicates whether an isolated tag `<it> [31]` is a beginning or an ending tag.

Value description:

`open` or `close`.

Default value:

Undefined.

Used in:

`<it>` [31].

priority

Priority - The priority of a <note> [19] element.

Value description:

A number between 1 and 10, 1 being the highest priority.

Default value:

1

Used in:

`<note>` [19].

process-name

Process name - The name specifying the type of process a given `<phase> [20]` corresponds to (e.g. Translation, Proofreading, Sizing, etc.).

Value description:

Text.

Default value:

Undefined.

Used in:

`<phase>` [20].

product-name

Product name - The name of the product which uses this file.

Value description:

Text.

Default value:

Undefined.

Used in:

`<file>` [17].

product-version

Product version - The version of the product which uses this file.

Value description:

Alpha-numeric.

Default value:

Undefined.

Used in:

`<file>` [17].

prop-type

Property type - The `prop-type` attribute specifies the type of a `<prop> [22]` element.

Important: Because the `<prop> [22]` element was DEPRECATED in version 1.1 and this attribute is only a member of that element, this attribute is also deprecated. Instead, use attributes defined in a namespace different from XLIFF. See the Extensibility section for more information.

Value description:

Text. No value defined by the standard.

Default value:

Undefined.

Used in:

`<prop>` [22].

purpose

Purpose - The `purpose` attribute specifies the purpose of a `<context-group> [21]` element. For example: `purpose="information"` indicates the content is informational only and not used for specific processing.

Value description:

The pre-defined values are defined in the table below.

Table 10.

Value	Description
`information`	Indicates that the context is informational in nature, specifying for example, how a term should be translated. Thus, should be displayed to anyone editing the XLIFF document.
`location`	Indicates that the context-group is used to specify where the term was found in the translatable source. Thus, it is not displayed.
`match`	Indicates that the context information should be used during translation memory lookups. Thus, it is not displayed.

In addition, user-defined values can be used with this attribute. A user-defined value must start with an "`x-`" prefix.

Combinations of these values can be used. For example, `purpose="location match x-validate"` provides both location (`location`) and TM matching (`match`) contextual information, as well as some user-defined data (`x-validate`).

Default value:

Undefined.

Used in:

`<context-group> [21]` .

reformat

Reformat - Indicates whether some properties (size, font, etc.) of the target can be formatted differently from the source.

Value description (stand-alone):

The pre-defined values are defined in the table below.

Table 11.

Value	Description
`yes`	This value indicates that all properties can be reformatted. This value must be used alone.
`no`	This value indicates that no properties should be reformatted. This value must be used alone.

Value description (enumerated):

The pre-defined values are defined in the table below.

Table 12.

Value	Description

coord	This value indicates that all information in the coord attribute can be modified.
coord-x	This value indicates that the x information in the coord attribute can be modified.
coord-y	This value indicates that the y information in the coord attribute can be modified.
coord-cx	This value indicates that the cx information in the coord attribute can be modified.
coord-cy	This value indicates that the cy information in the coord attribute can be modified.
font	This value indicates that all the information in the font attribute can be modified.
font-name	This value indicates that the name information in the font attribute can be modified.
font-size	This value indicates that the size information in the font attribute can be modified.
font-weight	This value indicates that the weight information in the font attribute can be modified.
css-style	This value indicates that the information in the css-style attribute can be modified.
style	This value indicates that the information in the style attribute can be modified.
ex-style	This value indicates that the information in the exstyle attribute can be modified.

In addition, user-defined values can be used with this attribute. A user-defined value must start with an "x-" prefix.

Except for the values yes and no, the other values can be used in combination, separated by a space. For example:

```
reformat="yes"
```

All properties can be reformatted.

```
reformat="no"
```

No properties should be reformatted.

```
reformat="font-name coord-x coord-y"
```

Only the name part of the font attribute, the x part of the coord attribute and the y part of the coord attribute can be modified.

Default value:

yes.

Used in:

<group> [23] , <trans-unit> [24], <bin-unit> [27].

resname

Resource name - Resource name or identifier of a item. For example: the key in the key/value pair in a Java properties file, the ID of a string in a Windows string table, the index value of an entry in a database table, etc.

Value description:

Text.

Default value:

Undefined.

Used in:

`<group>`. [23] `<trans-unit>` [24], `<alt-trans>` [26], `<target>` [25], `<bin-unit>` [27], `<bin-target>` [28].

restype

Resource type - Indicates the resource type of the container element.

Value description:

The pre-defined values are defined in the table below.

Table 13.

Value	Description
`auto3state`	Indicates a Windows RC AUTO3STATE control.
`autocheckbox`	Indicates a Windows RC AUTOCHECKBOX control.
`autoradiobutton`	Indicates a Windows RC AUTORADIOBUTTON control.
`bedit`	Indicates a Windows RC BEDIT control.
`bitmap`	Indicates a bitmap, for example a BITMAP resource in Windows.
`button`	Indicates a button object, for example a BUTTON control Windows.
`caption`	Indicates a caption, such as the caption of a dialog box.
`cell`	Indicates the cell in a table, for example the content of the <td> element in HTML.
`checkbox`	Indicates check box object, for example a CHECKBOX control in Windows.
`checkboxmenuitem`	Indicates a menu item with an associated checkbox.
`checkedlistbox`	Indicates a list box, but with a check-box for each item.
`colorchooser`	Indicates a color selection dialog.
`combobox`	Indicates a combination of edit box and listbox object, for example a COMBOBOX control in Windows.
`comboboxexitem`	Indicates an initialization entry of an extended combobox DLGINIT resource block. (code 0x1234).

comboboxitem	Indicates an initialization entry of a combobox DLGINIT resource block (code 0x0403).
component	Indicates a UI base class element that cannot be represented by any other element.
contextmenu	Indicates a context menu.
ctext	Indicates a Windows RC CTEXT control.
cursor	Indicates a cursor, for example a CURSOR resource in Windows.
datetimepicker	Indicates a date/time picker.
defpushbutton	Indicates a Windows RC DEFPUSHBUTTON control.
dialog	Indicates a dialog box.
dlginit	Indicates a Windows RC DLGINIT resource block.
edit	Indicates an edit box object, for example an EDIT control in Windows.
file	Indicates a filename.
filechooser	Indicates a file dialog.
fn	Indicates a footnote.
font	Indicates a font name.
footer	Indicates a footer.
frame	Indicates a frame object.
grid	Indicates a XUL grid element.
groupbox	Indicates a groupbox object, for example a GROUPBOX control in Windows.
header	Indicates a header item.
heading	Indicates a heading, such has the content of <h1>, <h2>, etc. in HTML.
hedit	Indicates a Windows RC HEDIT control.
hscrollbar	Indicates a horizontal scrollbar.
icon	Indicates an icon, for example an ICON resource in Windows.
iedit	Indicates a Windows RC IEDIT control.
keywords	Indicates keyword list, such as the content of the Keywords meta-data in HTML, or a K footnote in WinHelp RTF.
label	Indicates a label object.
linklabel	Indicates a label that is also a HTML link (not necessarily a URL).
list	Indicates a list (a group of list-items, for example an or element in HTML).
listbox	Indicates a listbox object, for example an LISTBOX control in Windows.
listitem	Indicates an list item (an entry in a list).
ltext	Indicates a Windows RC LTEXT control.
menu	Indicates a menu (a group of menu-items).

menubar	Indicates a toolbar containing one or more tope level menus.
menuitem	Indicates a menu item (an entry in a menu).
menuseparator	Indicates a XUL menuseparator element.
message	Indicates a message, for example an entry in a MESSAGETABLE resource in Windows.
monthcalendar	Indicates a calendar control.
numericupdown	Indicates an edit box beside a spin control.
panel	Indicates a catch all for rectangular areas.
popupmenu	Indicates a standalone menu not necessarily associated with a menubar.
pushbox	Indicates a pushbox object, for example a PUSHBOX control in Windows.
pushbutton	Indicates a Windows RC PUSHBUTTON control.
radio	Indicates a radio button object.
radiobuttonmenuitem	Indicates a menuitem with associated radio button.
rcdata	Indicates raw data resources for an application.
row	Indicates a row in a table.
rtext	Indicates a Windows RC RTEXT control.
scrollpane	Indicates a user navigable container used to show a portion of a document.
separator	Indicates a generic divider object (e.g. menu group separator).
shortcut	Windows accelerators, shortcuts in resource or property files.
spinner	Indicates a UI control to indicate process activity but not progress.
splitter	Indicates a splitter bar.
state3	Indicates a Windows RC STATE3 control.
statusbar	Indicates a window for providing feedback to the users, like 'read-only', etc.
string	Indicates a string, for example an entry in a STRINGTABLE resource in Windows.
tabcontrol	Indicates a layers of controls with a tab to select layers.
table	Indicates a display and edits regular two-dimensional tables of cells.
textbox	Indicates a XUL textbox element.
togglebutton	Indicates a UI button that can be toggled to on or off state.
toolbar	Indicates an array of controls, usually buttons.
tooltip	Indicates a pop up tool tip text.
trackbar	Indicates a bar with a pointer indicating a position within a certain range.

tree	Indicates a control that displays a set of hierarchical data.
uri	Indicates a URI (URN or URL).
userbutton	Indicates a Windows RC USERBUTTON control.
usercontrol	Indicates a user-defined control like CONTROL control in Windows.
var	Indicates the text of a variable.
versioninfo	Indicates version information about a resource like VERSIONINFO in Windows.
vscrollbar	Indicates a vertical scrollbar.
window	Indicates a graphical window.

In addition, user-defined values can be used with this attribute. A user-defined value must start with an "x-" prefix.

Default value:

Undefined.

Used in:

<group> [23], <trans-unit> [24], <target> [25], <alt-trans> [26], <bin-unit> [27], <bin-target> [28].

rid

Reference identifier - The rid attribute is used to link paired inline elements. The rid attribute of a begin-paired-code element should have the same value as the close-paired-code element. For example: <bx id="1" rid="1"/> ... <ex id="3" rid="1"/> indicates these elements are paired. If the rid attribute is not present (in a 1.0 document for example), the attribute id [46] is used to match both tags. For example: <bpt id='5'>xx</bpt> ... <ept id='5'>xx</ept>.

Value description:

Alpha-numeric without spaces.

Default value:

Undefined.

Used in:

<bpt> [31], <ept> [31], <it> [31], <bx/> [29], <ex/> [30].

size-unit

Unit of size attributes - The size-unit attribute specifies the units of measure used in the maxheight [47], minheight [50], maxwidth [48], and minwidth [50] attributes. The size-unit attribute is not related to the coord [38] attribute.

Value description:

The pre-defined values are defined in the table below.

Table 14.

Value	Description

byte	Indicates a size in 8-bit bytes.
char	Indicates a size in Unicode characters.
col	Indicates a size in columns. Used for HTML text area.
cm	Indicates a size in centimeters.
dlgunit	Indicates a size in dialog units, as defined in Windows resources.
em	Indicates a size in 'font-size' units (as defined in CSS).
ex	Indicates a size in 'x-height' units (as defined in CSS).
glyph	Indicates a size in glyphs. A glyph is considered to be one or more combined Unicode characters that represent a single displayable text character. Sometimes referred to as a 'grapheme cluster'
in	Indicates a size in inches.
mm	Indicates a size in millimeters.
percent	Indicates a size in percentage.
pixel	Indicates a size in pixels.
point	Indicates a size in point.
row	Indicates a size in rows. Used for HTML text area.

In addition, user-defined values can be used with this attribute. A user-defined value must start with an "x-" prefix.

Default value:

pixel.

Used in:

<group> [23], <trans-unit> [24].

source-language

Source language - The language for the <source> [25] elements in the given <file> [17] element.

Value description:

A language code as described in the [RFC 4646 [77]], the successor to [RFC 3066]. The values for this attribute follow the same rules as the values for xml:lang [69] . Unlike the other XLIFF attributes, the values for xml:lang [69] are not case-sensitive. For more information see the section on xml:lang in the XML specification [http://www.w3.org/TR/REC-xml#sec-lang-tag], and the erratum E11 [http://www.w3.org/XML/xml-V10-2e-errata#E11] (which replaces RFC 1766 by RFC 3066).

The source language can be also specified by xml:lang [69] in each <source> [25] element. The values of source-language and xml:lang [69] in <source> [25] can be different only in an <alt-trans> [26] element.

Default value:

Undefined.

Used in:

<file> [17].

state

State - The status of a particular translation in a <target> [25] or <bin-target> [28] element.

Value description:

The pre-defined values are defined in the table below.

Table 15.

Value	Description
final	Indicates the terminating state.
needs-adaptation	Indicates only non-textual information needs adaptation.
needs-l10n	Indicates both text and non-textual information needs adaptation.
needs-review-adaptation	Indicates only non-textual information needs review.
needs-review-l10n	Indicates both text and non-textual information needs review.
needs-review-translation	Indicates that only the text of the item needs to be reviewed.
needs-translation	Indicates that the item needs to be translated.
new	Indicates that the item is new. For example, translation units that were not in a previous version of the document.
signed-off	Indicates that changes are reviewed and approved.
translated	Indicates that the item has been translated.

In addition, user-defined values can be used with this attribute. A user-defined value must start with an "x-" prefix.

Default value:

Undefined.

Used in:

<target> [25], <bin-target> [28].

state-qualifier

State-qualifier - Describes the state of a particular translation in a <target> [25] or <bin-target> [28] element.

Value description:

The pre-defined values are defined in the table below.

Table 16.

Value	Description

`exact-match`	Indicates an exact match. An exact match occurs when a source text of a segment is exactly the same as the source text of a segment that was translated previously.
`fuzzy-match`	Indicates a fuzzy match. A fuzzy match occurs when a source text of a segment is very similar to the source text of a segment that was translated previously (e.g. when the difference is casing, a few changed words, white-space discripancy, etc.).
`id-match`	Indicates a match based on matching IDs (in addition to matching text).
`leveraged-glossary`	Indicates a translation derived from a glossary.
`leveraged-inherited`	Indicates a translation derived from existing translation.
`leveraged-mt`	Indicates a translation derived from machine translation.
`leveraged-repository`	Indicates a translation derived from a translation repository.
`leveraged-tm`	Indicates a translation derived from a translation memory.
`mt-suggestion`	Indicates the translation is suggested by machine translation.
`rejected-grammar`	Indicates that the item has been rejected because of incorrect grammar.
`rejected-inaccurate`	Indicates that the item has been rejected because it is incorrect.
`rejected-length`	Indicates that the item has been rejected because it is too long or too short.
`rejected-spelling`	Indicates that the item has been rejected because of incorrect spelling.
`tm-suggestion`	Indicates the translation is suggested by translation memory.

In addition, user-defined values can be used with this attribute. A user-defined value must start with an "`x-`" prefix.

Default value:

Undefined.

Used in:

<target> [25], <bin-target> [28].

style

Style - The resource style of a control. For example, in Windows resources it corresponds to the STYLE statement.

Value description:

Text.

Default value:

Undefined.

Used in:

<group> [23], <trans-unit> [24], <target> [25], <alt-trans> [26].

target-language

Target language - The language for the `<target> [25]` elements in the given `<file> [17]` element.

Value description:

A language code as described in the [RFC 4646 [77]], the successor to [RFC 3066]. The values for this attribute follow the same rules as the values for `xml:lang [69]` . Unlike the other XLIFF attributes, the values for `xml:lang [69]` are not case-sensitive. For more information see the section on `xml:lang` in the XML specification [http://www.w3.org/TR/REC-xml#sec-lang-tag], and the erratum E11 [http://www.w3.org/XML/xml-V10-2e-errata#E11] (which replaces RFC 1766 by RFC 3066).

The target language can be also specified by `xml:lang [69]` in each `<target> [25]` element. The values of `target-language` and `xml:lang [69]` in `<target> [25]` can be different only when in an `<alt-trans>` [26] element.

Default value:

Undefined.

Used in:

`<file>` [17].

tool

Creation tool - The `tool` attribute is used to specify the signature and version of the tool that created or modified the document.

Important: The `tool` attribute was DEPRECATED in version 1.1. Instead, use the `<tool> [65]` element and a tool-id [66] attribute.

Value description:

Text

Default value:

manual.

Used in:

`<file>` [17], `<phase>` [20], `<alt-trans>` [26].

tool-company

Tool company - The `tool-company` attribute specifies the company from which a tool originates.

Value description:

Text.

Default value:

Undefined.

Used in:

`<tool>` [65] .

tool-id

Tool identifier - The `tool-id` attribute allows unique identification of a `<tool>` [65] element. It is also used in other elements in the file to refer to the given `<tool>` [65] element.

Value description:

Text.

Default value:

Undefined.

Used in:

`<file>` [17], `<phase>` [20], `<alt-trans>` [26], `<tool> [65]` .

tool-name

Tool name - The `tool-name` attribute specifies the name of a given tool.

Value description:

Text.

Default value:

Undefined.

Used in:

`<tool> [65]` .

tool-version

Tool version - The `tool-version` attribute specifies the version of a given tool.

Value description:

Text.

Default value:

Undefined.

Used in:

<tool> [65].

translate

Translate - The `translate` attribute indicates whether or not the text referred to should be translated.

Value description:

Boolean: `yes` or `no`.

Default value:

yes.

Used in:

`<group>` [23] , `<trans-unit>` [24], `<bin-unit>` [27].

ts

Tool-specific data - The `ts` attribute allows you to include short data understood by a specific toolset. You can also use the `<prop>` [22] element to define large properties at the element level.

Important: The `ts` attribute was DEPRECATED in version 1.1. Instead, use attributes defined in a namespace different from XLIFF. See the Extensibility section for more information.

Value description:

Text. No value defined by the standard.

Default value:

Undefined.

Used in:

`<file>` [17], `<group>` [23], `<trans-unit>` [24], `<source>` [25], `<target>` [25], `<bin-unit>` [27], `<bin-source>` [27], `<bin-target>` [28], `<alt-trans>` [26], `<mrk>` [32], `<g>` [28], `<x/>` [29], `<bx/>` [29], `<ex/>` [30] , `<bpt>` [31], `<ept>` [31], `<ph>` [30] , `<it>` [31].

uid

Unique ID - The `uid` attribute is used to provide a unique ID to identify the skeleton file.

Value description:

Text.

Default value:

Undefined.

Used in:

`<external-file>` [18] .

unit

Unit - The `unit` attribute specifies the units counted in a `<count>` [21] element.

Value description:

The pre-defined values are defined in the table below.

Table 17.

Value	Description

word	Refers to words.
page	Refers to pages.
trans-unit	Refers to <trans-unit> elements.
bin-unit	Refers to <bin-unit> elements.
glyph	Refers to glyphs.
item	Refers to <trans-unit> and/or <bin-unit> elements.
instance	Refers to the occurrences of instances defined by the count-type value.
character	Refers to characters.
line	Refers to lines.
sentence	Refers to sentences.
paragraph	Refers to paragraphs.
segment	Refers to segments.
placeable	Refers to placeables (inline elements).

In addition, user-defined values can be used with this attribute. A user-defined value must start with an "x-" prefix.

Default value:

Undefined.

Used in:

<count> [21].

version

XLIFF version - The version attribute is used to specify the format version of the XLIFF document. This corresponds to the version number of the XLIFF specification that is being adhered to.

Value description:

Text.

Default value:

1.2

Used in:

<xliff> [17].

xid

Extern Reference identifier - The xid attribute is used to link an inline element to a different <trans-unit> [24] or <bin-unit> [27] element. For example, to link the text within a code to a corresponding translation unit.

Value description:

The value of the referenced id [46] .

Default value:

Undefined.

Used in:

<bpt> [31], <ept> [31], <it> [31], <ph> [30], <g> [28], <x/> [29], <bx/> [29], <ex/> [30], <sub> [32].

XML Namespace Attributes

xml:lang

Language - The xml:lang attribute specifies the language variant of the text of a given element. For example: xml:lang="fr-FR" indicates the French language as spoken in France.

Value description:

A language code as described in the [RFC 4646 [77]], the successor to [RFC 3066]. This declared value is considered to apply to all elements within the content of the element where it is specified, unless overridden with another instance of the xml:lang attribute. Unlike the other XLIFF attributes, the values for xml:lang are not case-sensitive. For more information see the section on xml:lang in the XML specification [http://www.w3.org/TR/REC-xml#sec-lang-tag], and the erratum E11 [http://www.w3.org/XML/xml-V10-2e-errata#E11] (which replaces RFC 1766 by RFC 3066).

Default value:

Undefined.

Used in:

<xliff> [17], <note> [19], <prop> [22], <source> [25], <target> [25], <alt-trans> [26].

xml:space

White spaces - The xml:space attribute specifies how white spaces (ASCII spaces, tabs and line-breaks) should be treated.

Value description:

default or preserve. The value default signals that an application's default white-space processing modes are acceptable for this element; the value preserve indicates the intent that applications preserve all the white space. This declared intent is considered to apply to all elements within the content of the element where it is specified, unless overridden with another instance of the xml:space attribute.

For more information see the section on xml:space in the XML specification [http://www.w3.org/TR/REC-xml#sec-white-space].

Default value:

default.

Used in:

<file> [17], <group> [23], <trans-unit> [24], <alt-trans> [26].

Conformance

The last section contains the conformance clauses/statements.

A. XLIFF Tree Structure

The following figure shows the possible structure as a tree. Each element is followed by notation indicating its possible occurrence according to the corresponding legend.

```
(legend: 1 = one
 + = one or more
 ? = zero or one
 * = zero, one or more)

<xliff> [17]1
| |
| +--- [Extension Point]
| |
+--- <file> [17]+
 |
 +--- <header> [17]?
 | |
 | +--- <skl> [18]?
 | | |
 | | +--- (<internal-file> [18] | <external-file> [18])1
 | |
 | +--- <phase-group> [20]?
 | | |
 | | +--- <phase> [20]+
 | | |
 | | +--- <note> [19]*
 | |
 | +--- <glossary> [19]*
 | | |
 | | +--- (<internal-file> [18] | <external-file> [18])1
 | |
 | +--- <reference> [19]*
 | | |
 | | +--- (<internal-file> [18] | <external-file> [18])1
 | |
 | +--- <count-group> [21]*
 | | |
 | | +--- <count> [21]*
 | |
 | +--- <tool> [65]*
 | | |
 | | +--- [Extension Point]
 | |
 | +--- <prop-group> [22]*
 | | |
 | | +--- <prop> [22]*
 | |
 | +--- [Extension Point]
 | |
 | +--- <note> [19]*
 |
 +--- <body> [23]1
 |
 +--- <group> [23]*
 | |
```

```
| +--- <context-group> [21]*
| | |
| | +--- <context> [22]+
| |
| +--- <count-group> [21]*
| | |
| | +--- <count> [21]*
| |
| +--- <prop-group> [22]*
| | |
| | +--- <prop> [22]*
| |
| +--- [Extension Point]
| |
| +--- <note> [19]*
| |
| +--- At least one of: (<group> [23]* <trans-unit> [24]* <bin-unit> [27]*)
|
+--- <trans-unit> [24]*
| |
| +--- <source> [25]1
| | |
| | +--- [Inline Elements]
| |
| +--- <target> [25]?
| | |
| | +--- [Inline Elements]
| |
| +--- <context-group> [21]*
| | |
| | +--- <context> [22]+
| |
| +--- <count-group> [21]*
| | |
| | +--- <count> [21]*
| |
| +--- <prop-group> [22]*
| | |
| | +--- <prop> [22]*
| |
| +--- <seg-source> [28]?
| | |
| | +--- [Inline Elements]
| |
| +--- [Extension Point]
| |
| +--- <note> [19]*
| |
| +--- <alt-trans> [26]*
| |
| +--- <context-group> [21]*
| | |
| | +--- <context> [22]+
| |
| +--- <source> [25]?
| | |
| | +--- [Inline Elements]
| | | +--- <target> [25]+
```

```
 | | |
 | | +--- [Inline Elements]
 | |
 | +--- <prop-group> [22]*
 | | |
 | | +--- <prop> [22]*
 | |
 | +--- <seg-source> [28]?
 | | |
 | | +--- [Inline Elements]
 | |
 | +--- [Extension Point]
 | |
 | +---- <note> [19]*
 |
 +--- <bin-unit> [27]*
 |
 +--- <bin-source> [27]1 & <bin-target> [28]?
 | |
 | +--- (<internal-file> [18] | <external-file> [18])1
 |
 +--- <context-group> [21]*
 | |
 | +--- <context> [22]+
 |
 +--- <count-group> [21]*
 | |
 | +--- <count> [21]*
 |
 +--- <prop-group> [22]*
 | |
 | +--- <prop> [22]*
 |
 +--- [Extension Point]
 |
 +--- <note> [19]*
 |
 +--- <trans-unit>*

Struct_Extension_Elements


Inline Elements:

---+--- <ph> [30]*
 | |
 | +--- <sub> [32]*
 | |
 | +--- [Inline Elements]
 |
 +--- <it> [31]*
 | |
 | +--- <sub> [32]*
 | |
 | +--- [Inline Elements]
 |
 +--- <bpt> [31]*
```

```
| |
| +--- <sub> [32]*
| |
| +--- [Inline Elements]
|
+--- <ept> [31]*
| |
| +--- <sub> [32]*
| |
| +--- [Inline Elements]
|
+--- <g> [29]*??? [29] | |
| +--- [Inline Elements]
|
+--- <x/> [29]*??? [29] | |
| +--- [Inline Elements]
|
+--- <bx/> [29]*
| |
| +--- [Inline Elements]
|
+--- <ex/> [30]*
| |
| +--- [Inline Elements]
|
+--- <mrk> [32]*
|
+--- [Inline Elements]
```

B. Schema

- The XML schema for XLIFF is available as strict or transitional:
  - Strict: xliff-core-1.2-strict.xsd [http://docs.oasis-open.org/xliff/v1.2/cs02/xliff-core-1.2-strict.xsd]
  - Transitional: xliff-core-1.2-transitional.xsd [http://docs.oasis-open.org/xliff/v1.2/cs02/xliff-core-1.2-transitional.xsd]

C. Changes Since Previous Version (Non-Normative)

The changes in this version relative to the previous version are as follows:

- Revised version number from 1.1 to 1.2.
- A new section (2.7) describing the Non Equivalent Translation concept including relevant examples.
- A new section (2.8) describing the Merged-translations concept including relevant examples.
- A new section (2.9) describing the Segmentation concept including relevant examples.
- Add the documentation concerning the `equiv-trans` [43] attribute regarding `<trans-unit>` [24] elements

- Add the documentation concerning the `merged-trans` [49] attribute for `<group>` [23] elements to section 3.2 (Elements).
- Add the `<seg-source>` [28] element as optional in the `<trans-unit>` [24] and `<alt-trans>` [26] content models, at the same level as `<source>` [25] .
- Create a new value "seg" for the `mtype` [50] attribute of the `<mrk>` [32] element.
- Add `mid` [49] as an optional attribute for the `<alt-trans>` [26] element.
- Updated Technical Committee section to reflect membership status as of 10 Oct 2005.
- Updated document link and name of XSD, and removed all references to DTD.
- Updated tree structure with new elements.
- Fixed typo in `<phase>` [20] section, changing "optional phase-name" to "required phase-name", as reported in this comment email -> http://lists.oasis-open.org/archives/xliff-comment/200509/msg00001.html [http://lists.oasis-open.org/archives/xliff-comment/200509/msg00001.html]
- Fixed additional typos as submitted via comment email from Yves Savourel on 2 November 2005 (email not visible in archive).
- Changed name attribute for `<context-group>` [21] from required to optional, and modified description.
- Added extension point at `<xliff>` [17]
- Added `alttranstype` [34] attribute to `<alt-trans>` [26].
- Deprecated the use of multiple `<target>` [25] elements in a single `<alt-trans>` [26].
- Deprecated the `restype` [58] attribute in `<target>` [25] element.
- Introduced `phase-name` [54] attribute in `<alt-trans>` [26].
- Introduced convention for more recent `<alt-trans>` [26] elements to appear before older ones.
- Fixed typo in section 2.4 Inline Elements, "act has placeholders" -> "act as placeholders"
- Fixed a number of problems with anchors and hyperlinks.
- Added explanation for deprecating the `restype` [58] attribute in `<target>` [25] element.
- Corrected missing `state-qualifier` `[63]` attribute in `<bin-target>` [28] element.
- In <group> [23] element rewrote: "One or more `<group>` [23], `<trans-unit>` [24], `<bin-unit>` [27] elements, in any order." to align with the schema definition: "Zero, one or more `<group>` [23], `<trans-unit>` [24], `<bin-unit>` [27] elements, in any order."
- Corrected broken link to <alt-trans> [26] from phase-name [54].
- Added `<xliff>` [17] and `<seg-source>` [28] to Adding Elements section.
- Clarified `<group>` [23], `<trans-unit>` [24] and `<bin-unit>` [27] to explicitly specify `id` [46] attribute is unique.

- Added guidelines for TMX interchange to inline tag discussion in Inline Elements section.
- Small. change to inline tags section to - "use `<bpt>` [31] and `<ept>` [31] instead of `<g>` [28] and `<ph>` [30] tags instead of `<x/>` [29]" , and added `equiv-text` [43] attribute description and appended to all inline tags.
- Numerous minor corrections: including: `<xliff> [17]` and `<seg-source> [28]` to section 2.5.4., added `alttranstype` [34] and `reformat` [56] added `equiv-trans` [43] to `<target>` [25], added `merged-trans` [49] to `<group> [23]` , and added anchor for `<seg-source> [28]` in `<alt-trans> [26].`
- Revised Segmentation section to describe representation of spaces in `<seg-source>` [28].
- Added text "non-XLIFF attributes" to all element definitions that support optional non-XLIFF attributes.
- Added text to `<xliff> [17]` indicating it supports non-XLIFF elements.
- Modified "uniquely identifies ... within file" to "uniquely identifies ... within <file> element" in `<phase> [20], <context-group> [21] and <tool> [65] .`
- Updated date of [XML Names] reference to latest revision.
- Removed SRX specification version numbers, and edited text in Segmentation section.
- Updated RFC3066 to [RFC4066] [77].
- Edited contents of `<seg-source> [28]` to be same as `<source> [25]` as per schema.
- Corrected typo "merged-trans element" to "merged-trans attribute".
- Corrected typos as pointed out by Asgeir in an email [http://lists.oasis-open.org/archives/xliff-comment/200610/msg00001.html] to the comments list , including removing from section 2.5.3 [http://lists.oasis-open.org/archives/xliff-comment/200610/msg00000.html] the priority [54] attribute erroneously included in the list of extensible attributes.
- Added note to section 2.5.4 indicating that non-XLIFF content will not be validated beyond ensuring it is well-formed.
- Fixed typo in section 3.1 where sample XLIFF XML declaration was erroneously located in the preceding paragraph.
- Corrected <alt-trans> [26] contents section so number of `<target>` [25] elements is one (multiple `<target>` [25]s in <alt-trans> [26] deprecated in 1.2)
- Corrected section 2.9 segmentation sample for an alt-trans.
- Clarified validation rules for strict and transitional variants in section 1.1.

D. Naming Guidelines (Non-Normative)

The following naming guidelines were used in writing this specification.

Elements and Attributes

The following guidelines were used for element and attribute naming.

1. Standard English letters.

2. Lower case only.

3. Hyphen is the preferred mean for creating composite names.

4. Industry standard terminology should be followed where possible.

Attribute Values

Attribute values are case sensitive. It is recommended that lower-case values are used. The specification recommends a number of values for some attributes, these are all lower-case.

Where multiple attribute values are to be used in an XLIFF document, two approaches are used: For enumerated attributes (such as the `purpose [55]` attribute of `<context-group> [21]`) the separator must be a space. For other textual attributes based on string, the specification recommends the use of the semi-colon as a separator for values. For example, multiple contacts may be listed for a `<file> [17]` with the attribute written thusly: `contact-name="Frank Sinatra;Sammy Davis Jnr;Dean Martin"`.

Processing Instructions

XLIFF reserves processing instructions that begin with "`xliff-`".

XLIFF File Extension

XLIFF documents use the `.xlf` extension. No other extension is recommended by the specification.

E. Acknowledgements

The XLIFF Technical Committee at OASIS is composed of the following members:

- Anastasiou, Dr. Dimitra, Localisation Research Centre
- Filip, Dr. David, Localisation Research Centre
- Lommel, Mr. Arle, Localization Industry Standards Assoc.
- Morado Vazquez, Lucia, Localisation Research Centre
- Raya, Mr. Rodolfo (Secretary), Maxprograms
- Reynolds, Peter, Polish Association of Translation Agencies
- Savourel, Mr. Yves, ENLASO Corporation
- Schnabel, Bryan, Tektronix
- Swift, Mr. Andrew, SDL International

F. References

Normative

[CSS]

CSS Specifications [http://www.w3.org/Style/CSS/#specs]. W3C (World Wide Web Consortium).

[IANA Charsets]

IANA Names for Character Sets [http://www.iana.org/assignments/character-sets]. IANA (Internet Assigned Numbers Authority), Aug 2001

[ISO 639]

Codes for the Representation of Names of Languages [http://lcweb.loc.gov/standards/iso639-2/langhome.html]. ISO (International Standards Organization), Nov 2001.

[ISO 3166]

Codes for the representation of names of countries and their subdivisions [http://www.iso.org/iso/en/prods-services/iso3166ma/index.html]. ISO (International Organization for Standardization), Jun 2000.

[ISO 8601]

Representation of dates and times [http://www.w3.org/TR/1998/NOTE-datetime-19980827]. ISO (International Organization for Standardization), Dec 2000.

[RFC 1341]

Multipurpose Internet Mail Extensions [http://www.ietf.org/rfc/rfc1341.txt]. IETF (Internet Engineering Task Force), Jun 1992.

[RFC 4646]

RFC 4646 Tags for the Identification of Languages [http://www.ietf.org/rfc/rfc4646.txt]. IETF (Internet Engineering Task Force), Sep 2006.

[XML 1.0]

Extensible Markup Language (XML) 1.0 [http://www.w3.org/TR/REC-xml] . W3C (World Wide Web Consortium).

[XML Names]

Namespaces in XML [http://www.w3.org/TR/REC-xml-names/] . W3C (World Wide Web Consortium), August 16, 2006.

Non-Normative

[ISO]

International Organization for Standardization [http://www.iso.org/] Web site.

[LISA]

Localisation Industry Standards Association [http://www.lisa.org/] Web site.

[OASIS]

Organization for the Advancement of Structured Information Standards [http://www.oasis-open.org/] Web site.

[OpenTag 1.2]

OpenTag Format Specifications [http://www.opentag.com/otspecs.htm]. ILE (International Language Engineering), Nov 1998.

[TMX]

Translation Memory eXchange (TMX) [http://www.lisa.org/standards/tmx/]. LISA (Localisation Industry Standards association).

[SRX]

Segmentation Rules eXchange (SRX) [http://www.lisa.org/standards/srx/]. LISA (Localisation Industry Standards association).

[Unicode]

Unicode Consortium [http://www.unicode.org/] Web site.

[W3C]

World Wide Web Consortium [http://www.w3c.org/] Web site.