XLIFF 1.1 Specification

Working Draft 2a, 15 Oct 2002

This version:
<not applicable yet>

Latest version:
<not applicable yet>

Previous version:
http://www.oasis-open.org/committees/xliff/documents/xliff-20020415.htm

Editor:
Yves Savourel <ysavourel@translate.com>

This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

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.

Status of this Document

This document is the latest Working Draft of the committee for the XLIFF 1.1 Specification. It is an OASIS draft document for review by OASIS members and other interested parties. Comments may be sent to xliff-comment@lists.oasis-open.org.

This document may be updated, replaced, or rendered obsolete by other documents at any time. It is inappropriate to use this document as reference material other than "work in progress".

Text to be added in version 1.1 is marked in green highlighting. Text that is 1.0 but is to be removed in 1.1 is marked in red highlighting. Draft notes (that will be removed in the final version) are in [brackets and yellow highlighting].

Table of Contents

1. Introduction
    1.1. Naming Convention
        1.1.1. Elements and Attributes
        1.1.2. Attribute Values
        1.1.3. Processing Instructions
        1.1.4. XLIFF File Extension
2. General Structure
    2.1. Header
    2.2. Body
    2.3. Named Groups
    2.4. Extensibility
        2.4.1. Adding Elements
        2.4.2. Adding Attributes
        2.4.3. Adding Attribute Values
        2.4.4. Validating Documents with Extensions
    2.5. Embedding XLIFF
3. Detailed Specifications
    3.1. XML Declaration
    3.2. Elements
        3.2.1. Top-level and Header Elements
        3.2.2. Named Group Elements
        3.2.3. Structural Elements
        3.2.4. Inline Elements
        3.2.5. Delimiter Element
    3.3. Attributes

3.3.1. XLIFF Attributes
3.3.2. XML Namespace Attributes

Appendices

A. XLIFF Tree Structure
B. Document Type Definition for XLIFF and Schemas
C. Changes Since Previous Version (Non-Normative)
D. Naming Guidelines (Non-Normative)
DE. XLIFF Technical Committee (Non-Normative)
EF. References

1. 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.

[The Naming Conventions section has been renamed and moved as an appendix]

2. General Structure

XLIFF is XML, as such it begins with an XML declaration. After the XML declaration comes the XLIFF document itself, enclosed within the `<xliff>` element. A XLIFF document is composed of zero, one or more sections, each enclosed within a `<file>` element. The `<file>` element consists of a `<header>` element, which contains meta-data about the `<file>`, and a `<body>` element, which contains the extracted translatable data from the `<file>`. The translatable data is contained within `<trans-unit>` elements in `<source>` and `<target>` paired elements. These `<trans-unit>` elements can be grouped recursively in `<group>` elements.

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

The complete tree structure is available in Appendix A.

2.1. Header

The XLIFF <header> contains meta-data about the file and the localization process. It contains the <skl>, <phase-group>, <glossary>, <reference>, <count-group>, <tool>, <prop-group>, and <note> elements. The <skl> element contains either a skeleton file of the file submitted for localization or a hypertext link to that file.

The <phase-group> element contains information about each processing phase used in localizing the file; references to these phases are stored along with the translations. The <glossary> and <reference> 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> element is a grouping element of count information of the entire file. The <prop-group> element contains tool-specific information used in combining the data with the skeleton file or storing the data in a repository. The <note> element contains instructions for the localization process. The <count-group>, <prop-group>, and <note> elements can also appear in the body of the file.

2.2. Body

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

The <trans-unit> and <bin-unit> elements are the leaf nodes of the tree structure. The <trans-unit> element contains the text to be translated, the translations, and other related information. The <bin-unit> 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> element the text to be translated is contained in a <source> element. This element may contain inline elements that either remove the codes from the source (<g>, <x/>, <bx/>, <ex/>) or that mask off codes left inline (<bpt>, <ept>, <sub>, <it>, <ph>). The translated text is contained in a <target> element that has the same inline codes available to it as does the <source> element. Translation matches generated by a TM or MT or entered by a translator may be provided in a <alt-trans> element, which also contains the <source> and <target> elements.

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

2.3. 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>, <count-group> and <prop-group>.

The <context-group> element contains context information for the source strings to be used in the translation process. There could be context information that is useful to the translator that should be shown and context information that is designed specifically for a translation tool. Each of these groups of context information can be grouped in a named `<context-group>` and processing instructions can tell the systems which `<context-group>` to use for each purpose.

The `<count-group>` 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>` to update during the localization process.

The `<prop-group>` 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>` 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>` has been deprecated in version 1.1.

--- START NEW SECTION (2.4.*) --- [not highlighted for legibility]

2.4. 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. There are several ways to do this in XLIFF. All of them use the namespace mechanism [XML Names]. You can add non-XLIFF elements, as well as attributes and attribute values.

Although XLIFF offer this extensibility mechanism, in order 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.

2.4.1. Adding Elements

XLIFF provides several extension points in the following elements: `<header>`, `<group>`, `<trans-unit>`, `<alt-trans>`, and `<bin-unit>`. Within each of these elements, the extension point is located just before the optional <note> elements; that is also just after where the optional `<prop-group>` elements (now deprecated) would be.

Several non-XLIFF element 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 element (in bold) within an XLIFF document:

```
<xliff version='1.1'
 xmlns='urn:oasis:names:tc:xliff:document:1.1'
 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">
     <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 neither the <source> and <target> elements. However, the <mrk> element can be used to markup sections of the text with user-defined values assigned to the mtype attribute. You can also add non-XLIFF attributes to most of the inline elements used in <source> and <target>.

2.4.2. Adding Attributes

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

The following elements allow non-XLIFF attributes: <file>, <group>, <trans-unit>, <source>, <target>, <bin-unit>, <bin-source>, <bin-target>, <alt-trans>, <mrk>, <g>, <x/>, <bx/>, <ex/>, <bpt>, <ept>, <ph>, and <it>.

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

```
<xliff version='1.1'
 xmlns='urn:oasis:names:tc:xliff:document:1.1'
 xmlns:htm='http://www.w3.org/TR/REC-html40'>
 <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='1' htm:valign='top' htm:width='30%'>
     <source>Text of row 1 column 2</source>
    </trans-unit>
   </group>
   <group restype='row'>
    <trans-unit id='1' htm:valign='top' htm:width='30%'>
     <source>Text of row 2 column 1</source>
    </trans-unit>
    <trans-unit id='1' 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.

2.4.3. Adding Attribute Values

[NOTE: This is the same method as initially proposed. We still need to make a decision on this section.]

Many attributes in XLIFF offer a list of enumerated values. Some applications may found 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: context-type, count-type, ctype, datatype, mtype, restype, size-unit, state, unit, and priority.

The lists of values for XLIFF attributes are defined in a specific schema. Each list has a set of pre-defined standard values declared as a type in that schema. A second schema is reserved for declaring extended values. This file contains a type for each set of extended values. The type used in the definition of the value for a given element is the union of both the standard type and the extended type.

For example, the standard values for the attribute context-type that is used in the <context> element are defined as follow:

```
<xsd:simpleType name="context-typeValueListStd">
 <xsd:restriction base="xsd:NMTOKEN">
  <xsd:enumeration value="database"/>
  <xsd:enumeration value="element"/>
  <xsd:enumeration value="elementtitle">
  <xsd:enumeration value="recordtitle">
 </xsd:restriction>
</xsd:simpleType>
```

Extended values can be defined in the type `context-typeValueListExt`, reserved for this:

```
<xsd:simpleType name="context-typeValueListExt">
 <xsd:restriction base="xsd:NMTOKEN">
  <!-- Add your values here -->
  <xsd:enumeration value="user-defined-value1"/>
 </xsd:restriction>
</xsd:simpleType>
```

The type assigned to the context-type attribute is the union of both standard and extended types:

```
<xsd:simpleType name="context-typeValueList">
 <xsd:union memberTypes="context-typeValueListStd context-typeValueListExt"/>
</xsd:simpleType>
```

```
<xsd:complexType name="ElemType_context">
 <xsd:simpleContent>
  <xsd:extension base="xsd:string">
   <xsd:attribute name="context-type" type="xlfv:context-typeValueList" use="required
   ...
  </xsd:simpleContent>
</xsd:complexType>
```

To define your own values, and be able to validate the documents that use them, you need to enumerate the values in the type declaration of the relevant extended value list. For instance, if you defined the following values for the extended value list for `context-type`, you can use them in an XLIFF document.

```
<xsd:simpleType name="context-typeValueListExt">
 <xsd:restriction base="xsd:NMTOKEN">
  <!-- Add your values here -->
  <xsd:enumeration value="for-translators"/>
  <xsd:enumeration value="for-engineers"/>
  <xsd:enumeration value="for-authors"/>
  <xsd:enumeration value="for-editors"/>
 </xsd:restriction>
</xsd:simpleType>
```

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

2.4.4. 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 XMLSchema-instance namespace to define what schemas to use to validate the document (XLIFF and the non-XLIFF namespaces).

```
<xliff version='1.1'
 xmlns='urn:oasis:names:tc:xliff:document:1.1'
 xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
 xsi:schemaLocation='
  urn:oasis:names:tc:xliff:document:1.1 xliff-1-1.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.

--- END NEW SECTION ---

--- START NEW SECTION (2.5.*) --- [not highlighted for legibility]

2.5. 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, it 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="unbo
 </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.

--- END NEW SECTION ---

3. Detailed Specifications

3.1. 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] listing.

In addition to the XML declaration, it is recommended to use the following DOCTYPE declaration:

```
<!DOCTYPE xliff PUBLIC "-//XLIFF//DTD XLIFF//EN"
 "http://www.oasis-open.org/committees/xliff/documents/xliff.dtd" >
```

If necessary, you can also specify a namespace for XLIFF. The namespace identifier for this standard is "`urn:oasis:names:tc:xliff:document:1.01`".

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

```
<?xml version="1.0"?>
<!DOCTYPE xliff PUBLIC "-//XLIFF//DTD XLIFF//EN"
 "http://www.oasis-open.org/committees/xliff/documents/xliff.dtd" >
<xliff version="1.1">
 <file source-language="EN" datatype="plaintext" original="file.ext">
  <header></header>
  <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 XMLSchema-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='1.1'
 xmlns='urn:oasis:names:tc:xliff:document:1.1'
 xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
 xsi:schemaLocation='
  urn:oasis:names:tc:xliff:document:1.1 xliff-1-1.xsd
  urn:oasis:names:tc:xliff-values:document:1.1 xliff-values-1-1.xsd'>
 <file source-language="EN" datatype="plaintext" original="file.ext">
  <body>
   <trans-unit id="1">
    <source>Hello World!</source>
   </trans-unit>
  </body>
 </file>
</xliff>
```

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

3.2. 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.

Top Level and Header elements	<xliff>, <file>, <header>, <skl>, <external-file>, <internal-file>, <glossary>, <reference>, <phase-group>, <phase>, <tool>, <note>.
Named Group Elements	<context-group>, <context>, <count-group>, <count>, <prop-group>, <prop>.
Structural elements	<body>, <group>, <reformat>, <trans-unit>, <source>, <target>, <bin-unit>, <bin-source>, <bin-target>, <alt-trans>.
Inline elements	<g>, <x/>, <bx/>, <ex/>, <bpt>, <ept>, <sub>, <it>, <ph>.
Delimiter element	<mrk>.

3.2.1. 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.

Required attributes:

`version`.

Optional attributes:

`xml:lang`.

Contents:

One or more `<file>` elements.

<file>

File - The `<file>` element corresponds to a single extracted original document.

Required attributes:

`original`, `source-language`, `datatype`.

Optional attributes:

`tool`, `tool-id`, `date`, `xml:space`, `ts`, `category`, `target-language`, `product-name`, `product-version`, `build-num`.

Contents:

Zero or oOne <header> element, followed by
One <body> element.

<header>

File header - The <header> element contains data relating to the <file> element.

Required attributes:

None.

Optional attributes:

None.

Contents:

zero or one <skl> element, followed by
zero or one <phase-group> element, followed by
zero, one or more <glossary> elements, followed by
zero, one or more <reference> elements, followed by
zero, one or more <count-group> elements, followed by
zero, one or more <tool> elements, followed by
zero, one or more <note> elements, followed byzero, one or more <prop-group>, <glossary>, <reference>, <note>, <count-group> elements in any order.
zero, one or more <prop-group> elements, followed by
Zero, one or more non-XLIFF elements, followed by

<skl>

Sekeleton file - The <skl> element contains the skeleton file or the location of the skeleton file.

Required attributes:

None.

Optional attributes:

None

Contents:

Either exactly one <internal-file> or one <external-file> element.

<internal-file>

Internal file - The <internal-file> element will contain the data for the skeleton file.

Required attributes:

None.

Optional attributes:

`form`, `crc`.

Contents:

An embedded file.

<external-file>

External file - An empty element that specifies the file location.

Required attributes:

`href`.

Optional attributes:

`uid`, `crc`.

Contents:

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

<glossary>

Glossary - Element points to the glossary source.

Required attributes:

None.

Optional attributes:

None.

Contents:

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

<reference>

Reference - Contains information about the reference material.

Required attributes:

None.

Optional attributes:

None.

Contents:

A description of the reference material and either exactly one `<internal-file>` or one `<external-file>` 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>`, comments from the translator about the translation, or any comment from anyone involved in processing the XLIFF file.

Required attributes:

None.

Optional attributes:

`xml:lang`, `from`, `priority`.

Contents:

Text, no standard elements.

<phase-group>

Phase group - The `<phase-group>` element contains phase information. This phase information is specific to the users.

Required attributes:

None.

Optional attributes:

None.

Contents:

One or more `<phase>` elements.

<phase>

Phase information - The `<phase>` contains metadata about a particular version of XLIFF data.

Required attributes:

`phase-name`, `process-name`.

Optional attributes:

`company-name`, `tool`, `tool-id`, `date`, `job-id`, `contact-name`, `contact-email`, `contact-phone`.

Contents:

Zero, one or more `<note>` elements.

<tool>

Tool - The `<tool>` element allows to identify the tool that has been used to execute a given task in the document.

Required attributes:

`tool-id`, `tool-name`.

Optional attributes:

`tool-version`, `tool-company`.

Contents:

Empty.

3.2.2. 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>` elements must be named, allowing different uses for each group. These uses can be controlled through the use of XML processing instructions. For example, the following instruction could indicate that any `<count-group>` element with a name set to `'value'` should be updated during translation:

```
<?xliff-update-count-group name='value' ?>
```

Required attributes:

`name`.

Optional attributes:

None.

Contents:

One or more `<count>` elements.

<count>

Count - For each `<count>` element the the required count-type attribute would indicate what kind of count the element represents, and the optional `unit` attribute would indicate the unit of the count (by default: `word`). A list of recommended values for `count-type` and `unit` is provided in the XLIFF Values schemaby the specification.

Required attributes:

`count-type`.

Optional attributes:

`unit`.

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>` level, a `<trans-unit>` level, or a `<alt-trans>` level.

Each `<context-group>` element must be named, allowing different uses for each group. These uses can be controlled through the use of XML processing instructions. For example, the following instruction could indicate that any `<context-group>` element with a name set to `'value'` should be displayed to the end-user:

```
<?xliff-show-context-group name='value' ?>
```

Because the `<context-group>` element may occur at a very high level, a default context can be established for all `<trans-unit>` elements within a file. This default can be overridden at many subsequent levels.

Required attributes:

`name`.

Optional attributes:

crc.

Contents:

One or more <context> elements.

<context>

Context - The <context> element describes the context of a <source> within a <trans-unit> or a <alt-trans>. 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 visibility of <context> information can be controlled by XML processing instructions. For example, the following instruction may indicate that any <context> element with a context-type set to 'value' should be displayed to the end-user.

```
<?xliff-show-context context-type='value'?>
```

Required attributes:

context-type

Optional attributes:

match-mandatory, crc

Contents:

Text, no standard elements.

<prop-group>

Property group - The <prop-group> element contains <prop> 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 is 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.

Contents:

One or more <prop> 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 is 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.

Optional attributes:

xml:lang.

Contents:

Tool-specific data or text, no standard elements.

3.2.3. Structural Elements

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

<body>

File body - The <body> element contains the structural elements.

Required attributes:

None.

Optional attributes:

None.

Contents:

Zero, one or more <group>, <trans-unit>, <bin-unit> 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.

Required attributes:

None.

Optional attributes:

`id`, `datatype`, `xml:space`, `ts`, `restype`, `resname`, `extradata`, `help-id`, `menu`, `menu-option`, `menu-name`, `coord`, `font`, `css-style`, `style`, `exstyle`, `extype`.

Contents:

Zero, one or more `<context-group>` elements, followed by
Zero, one or more `<count-group>` elements, followed by
Zero, one or more `<note>` elements, followed by
Zero, one or more `<prop-group>` elements, followed by
Zero, one or more non-XLIFF elements, followed by
Zero, one or moreAt least one of `<group>`, `<trans-unit>`, `<bin-unit>` elements in any order.
All `<context-group>`, `<count-group>`, `<prop-group>`, and `<note>` 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>`, `<target>` and associated elements. Lists of recommended values for the `datatype`, `restype`, and `size-unit` attributes are available in the XLIFF Values schema.

Required attributes:

`id`.

Optional attributes:

`approved`, `translate`, `reformat`, `xml:space`, `datatype`, `ts`, `phase-name`, `restype`, `resname`, `extradata`, `help-id`, `menu`, `menu-option`, `menu-name`, `coord`, `font`, `css-style`, `style`, `exstyle`, `extype`, `maxbytes`, `minbytes`, `size-unit`, `maxheight`, `minheight`, `maxwidth`, `minwidth`, `charclass`.

Contents:

One `<source>` element, followed by
Zero or one `<target>` element, followed by
Zero, one or more `<alt-trans>` elements, followed by
Zero, one or more `<context-group>` elements, followed by
Zero, one or more `<count-group>` elements, followed by
Zero, one or more `<reformat>` elements, followed by
Zero, one or more `<note>` elements, followed by
Zero, one or more `<prop-group>` elements, followed by
Zero, one or more non-XLIFF elements.

Zero, one or more <note>, <context-group>, <prop-group>, <alt-trans>, <count-group> elements in any order.
All child elements of <trans-unit> pertain to their sibling <source> element.

<reformat>

Reformat - The <reformat> element is to indicate whether the target properties can be formatted (size, font, etc.) differently from the source. Use the element attribute to specify the name of the element content to reformat or not, or use the attribute attribute to specify the name of the attribute value to reformat or not. The attribute edit allows you to specify whether or not the given item is to be modified.

Example:

```
<trans-unit id='1' restype='button' coord='8;8;54;14'>
 <reformat attribute='coord' edit='yes'/>
 <source>Accept</source>
</trans-unit>
```

Required attributes:

element or attribute, edit.

Optional attributes:

None.

Contents:

None.

<source>

Source text - The <source> element is used to delimit a unit of text that could be a paragraph, a title, a menu item, a caption, etc.

Required attributes:

None.

Optional attributes:

xml:lang, ts.

Contents:

Text,
Zero, one or more of the following elements: <g>, <x/>, <bx/>, <ex/>, <bpt>, <ept>, <ph>, <it>, <mrk>, in any order.

<target>

Target - The `<target>` element is used to delimit a unit of text. A paragraph in XLIFF does not necessarily correspond to a "paragraph" in a word-processor. It's simply a unit of text that could be a paragraph, a title, a menu item, a caption, etc. A list of preferred values for the `restype` attribute is available.

Required attributes:

None.

Optional attributes:

`state`, `phase-name`, `xml:lang`, `ts`, `restype`, `resname`, `coord`, `font`, `css-style`, `style`, `exstyle`.

Contents:

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

<alt-trans>

Translation match - The `<alt-trans>` element contains a possible translation in a `<target>` along with optional context, notes, etc.

Required attributes:

None.

Optional attributes:

`match-quality`, `tool`, `tool-id`, `crc`, `xml:lang`, `datatype`, `xml:space`, `ts`, `restype`, `resname`, `extradata`, `help-id`, `menu`, `menu-option`, `menu-name`, `coord`, `font`, `css-style`, `style`, `exstyle`, `extype`, `origin`.

Contents:

Zero or one `<source>` element, followed by
One or more `<target>` elements, followed by
Zero, one or more `<context-group>` elements, followed by
Zero, one or more `<note>` elements.
Zero, one or more `<prop-group>` elements, followed by
Zero, one or more non-XLIFF elements.
Zero, one or more `<note>`, `<context-group>`, `<prop-group>` elements in any order.
All child elements of `<alt-trans>` pertain to their sibling `<target>` element.

<bin-unit>

Binary unit - The `<bin-unit>` element contains a binary object that may or may not be translatable.

Required attributes:

`id`, `mime-type`.

Optional attributes:

`approved`, `translate`, `reformat`, `ts`, `phase-name`, `restype`, `resname`.

Contents:

One `<bin-source>` element, followed by
Zero or one `<bin-target>` elements, followed by
Zero, one or more `<context-group>` elements, followed by
Zero, one or more `<count-group>` elements, followed by
Zero, one or more `<reformat>` elements, followed by
Zero, one or more `<note>` elements, followed by
Zero, one or more `<prop-group>` elements, followed by
Zero, one or more non-XLIFF elements, followed by
Zero or more `<trans-unit>` elements.
Zero, one or more `<note>`, `<context-group>`, `<prop-group>`, `<trans-unit>`, `<count-group>` elements in any order.
All child elements of `<bin-unit>` pertain to their sibling `<bin-source>` element.

<bin-source>

Binary source - The `<bin-source>` element is the container for the binary source data.

Required attributes:

None.

Optional attributes:

`ts`.

Contents:

One of `<internal-file>` or `<external-file>`.

<bin-target>

Binary target - The `<bin-target>` element is the container for the translated version of the binary data.

Required attributes:

None.

Optional attributes:

mime-type, ts, state, phase-name, restype, resname.

Contents:

One of <internal-file> or <external-file>.

3.2.4. Inline Elements

The inline elements are the elements that can appear inside the <source> and <target> elements. They enclose or replace any formatting or control codes 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 and can be moved within its parent structural element. When possible, the ctype allows you to specify what kind of attribute the placeholder represents. A list of preferred values for the ctype attribute is available.

A <g> element can contain another <g> element. In this case, if the embedded group has an id attribute, it should never be moved outside of its parent group.

Required attributes:

id.

Optional attributes:

ctype, ts, clone.

Contents:

Text,
Zero, one or more of the following elements: <g>, <x/>, <bx/>, <ex/>, <bpt>, <ept>, <ph>, <it>, <mrk>, in any order.

<x/>

Generic placeholder - The <x/> element is used to replace any code of the original document. When possible, the ctype allows you to specify what kind of attribute the placeholder represents. A list of preferred values for the ctype attribute is available.

Required attributes:

id.

Optional attributes:

ctype, ts, clone.

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>` element is used because it simplifies processing. The `<bx/>` element should be followed by a matching `<ex/>` element. These paired elements are related via their `rid` attributes. When possible, the `ctype` allows you to specify what kind of attribute the placeholder represents. A list of preferred values for the `ctype` attribute is available.

Required attributes:

`id`.

Optional attributes:

`rid`, `ctype`, `ts`, `clone`.

Contents:

Empty.

<ex/>

End paired placeholder - The `<ex/>` 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>` element is used because it simplifies processing. The `<ex/>` element should be preceded by a matching `<bx/>` element. These paired elements are related via their `rid` attributes.

Required attributes:

`id`.

Optional attributes:

`rid`, `ts`.

Contents:

Empty.

<ph>

Placeholder - The `<ph>` element is used to delimit a sequence of native stand-alone codes in the segment. When possible, the ctype allows you to specify what kind of attribute the placeholder represents. A list of preferred values for the ctype attribute is available.

Required attributes:

id.

Optional attributes:

ctype, ts, crc, assoc.

Contents:

Code data,
Zero, one or more <sub> 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> element within the segment. When possible, the ctype allows you to specify what kind of attribute the placeholder represents. A list of preferred values for the ctype attribute is available.

Required attributes:

id.

Optional attributes:

rid, ctype, ts, crc.

Contents:

Code data,
Zero, one or more <sub> 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> element within the segment.

Required attributes:

id.

Optional attributes:

rid, ts, crc.

Contents:

Code data,

Zero, one or more <sub> 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 segment. When possible, the ctype allows you to specify what kind of attribute the placeholder represents. A list of preferred values for the ctype attribute is available.

Required attributes:

id, pos.

Optional attributes:

rid, ctype, ts, crc.

Contents:

Code data,
Zero, one or more <sub> 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. When possible, the ctype allows you to specify what kind of attribute the placeholder represents. Lists of preferred values for the ctype and datatype attributes are available.

Required attributes:

None.

Optional attributes:

datatype, ctype.

Contents:

Text,
Zero, one or more of the following elements: <g>, <x/>, <bx/>, <ex/>, <bpt>, <ept>, <ph>, <it>, <mrk>, in any order.

3.2.5. 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 are 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. A list of preferred values for the `mtype` attribute is available.

Required attributes:

`mtype`.

Optional attributes:

`mid`, `ts`, `comment`.

Contents:

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

3.3. 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 are the "Recommended Attribute Values". Values for these attributes are case sensitive. These lists are purely informative, the goal is to specify a preferred syntax so tools can have some level of compatibility.

XLIFF attributes	`approved`, `assoc`, `attribute`, `build-num`, `ctype`, `category`, `charclass`, `comment`, `company-name`, `contact-email`, `contact-name`, `contact-phone`, `coord`, `count-type`, `crc`, `css-style`, `datatype`, `date`, `edit`, `element`, `exstyle`, `extradata`, `extype`, `font`, `form`, `from`, `help-id`, `href`, `id`, `job-id`, `match-mandatory`, `match-quality`, `maxheight`, `maxbytes`, `maxwidth`, `menu`, `menu-name`, `menu-option`, `mid`, `mime-type`, `minheight`, `minbytes`, `minwidth`, `mtype`, `name`, `original`, `phase-name`, `pos`, `priority`, `process-name`, `product-name`, `product-version`, `prop-type`, `reformat`, `resname`, `restype`, `rid`, `source-language`, `state`, `style`, `tool`, `tool-company`, `tool-id`, `tool-name`, `tool-version`, `target-language`, `translate`, `ts`, `uid`, `unit`, `version`.

XML namespace attributes	xml:lang, xml:space.

3.3.1. XLIFF Attributes

---

approved

Approved - Indicates whether a translation is final.

Value description:
Boolean: `yes` or `no`.

Default value:
`no`.

Used in:
`<trans-unit>`, `<bin-unit>`.

---

assoc

Association - Indicates the association of a `<ph>` with the text prior or after.

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>`.

---

attribute

Attribute - Indicates the attribute name which should or should not be reformatted.

Value description:
Name of the attribute to be or not to be reformatted.

Default value:
Empty string.

Used in:

`<reformat>`.

---

build-num

Build number - The build number of version of the product or application the localizable material is for.

Value description:

Alpha-numeric.

Default value:

Undefined.

Used in:

`<file>`.

---

category

Category - This would give information on the subject of what is being translated.

Value description:

Text.

Default value:

Undefined.

Used in:

`<file>`.

---

charclass

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

Value description:

Text.

Default value:

Undefined.

Used in:

<trans-unit>.

---

clone

Clone - This indicates that a copy of the given inline element can be made and placed in the <target>.

Value description:
Boolean: `yes` or `no`.

Default value:
`yes`.

Used in:
<g>, <x/>, <bx/>.

---

comment

Comment - A comment in a tag.

Value description:
Alpha-numeric.

Default value:
Undefined.

Used in:
<mrk>.

---

company-name

Company name - The client name.

Value description:
Text.

Default value:
Undefined.

Used in:
<phase>.

---

contact-email

Contact email - The contact email at the client company (email of the contact-name person).

Value description:
Text.

Default value:
Undefined.

Used in:
`<phase>`.

---

contact-name

Contact name - The contact at the client company.

Value description:
Text.

Default value:
Undefined.

Used in:
`<phase>`.

---

contact-phone

Contact phone - Phone number of the contact-name person.

Value description:
Text.

Default value:
Undefined.

Used in:
`<phase>`.

---

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:

Text. The recommended values for the context-type attribute of the <context> element are defined in the XLIFF Values schema as the context-typeValueList type.as follow (this list is not exhaustive):

- Database
- Element
- ElementTitle
- Record
- RecordTitle[Note: attribute values are supposed to be lowercase]

Default value:

Undefined.

Used in:

<context>.

---

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 (like 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>, <trans-unit>, <target>, <alt-trans>.

---

count-type

Count type - The count-type attribute specifies the purpose of the <count> element.

Value description:

Text. These values may correspond to phases.[What do we mean by this previous sentence?] The recommended values for the count-type attribute are defined in the XLIFF Values schema

as the `count-typeValueList` type.as follow (this list is not exhaustive):
- `new` = New items to translate.
- `exact-match` = Items that have at least one exact match (in a `<alt-trans>` element).
- `fuzzy-match` = Items that have at least one fuzzy match (in a `<alt-trans>` element).
- `total` = Total count for the given item.

Default value:

None.

Used in:

`<count>`.

---

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>`, `<external-file>`, `<context-group>`, `<context>`, `<alt-trans>`, `<bpt>`, `<ept>`, `<it>`, `<ph>`.

---

css-style

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

Value description:

Text, the value is subject to CSS syntax rules.

Default value:

Undefined.

Used in:

`<group>`, `<trans-unit>`, `<target>`, `<alt-trans>`.

---

ctype

Content type - The type attribute specifies the content 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 value will depend on each element. The recommended values for the `ctype` attribute of the `<x/>` and `<ph>` elements are defined in the XLIFF Values schema as the `InlinePlaceholdersValueList` type.as follow (this list is not exhaustive):
- `pb` = Paragraph break.
- `lb` = Line-break.
- `image` = Image or other graphic.

Text. The recommended values for the `ctype` attribute of the other elements are defined in the XLIFF Values schema as the `InlineDelimitersValueList` type.as follow (this list is not exhaustive):
- `bold` = Bold or strong text.
- `font` = Font size, font face, color change, etc.).
- `italic` = Italicized text.
- `underlined` = Underlined text.
- `link` = hypertext link.

Default value:

Undefined.

Used in:

`<g>`, `<x/>`, `<bx/>`, `<bpt>`, `<sub>`, `<it>`, `<ph>`.

---

datatype

Data type - The `datatype` attribute specifies the kind of text contained in the element. Depending on that type, you may apply different processes to the data.

Value description:

Text. The recommended values for the `datatype` attribute are defined in the XLIFF Values schema as the `datatypeValueList` type.as follow (this list is not exhaustive):
- `cdf` = Channel Definition Format.
- `cpp` = C and C++ style text.
- `html` = HTML, DHTML, etc.
- `interleaf` = Interleaf documents.
- `java` = Java, source and property files.
- `javascript` = JavaScript, ECMAScript scripts.
- `lisp` = Lisp.
- `mif` = Framemaker MIF, MML, etc.
- `pascal` = Pascal, Delphi style text.
- `plaintext` = Plain text.
- `rtf` = Rich Text Format.

- `sgml` = SGML.
- `vbscript` = Visual Basic scripts.
- `winres` = Windows resources from RC, DLL, EXE.
- `xml` = XML.

Default value:

Empty string.

Used in:

`<file>`, `<group>`, `<trans-unit>`, `<alt-trans>`, `<sub>`.

---

date

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

Value description:

Date in [ISO 8601] 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>`, `<phase>`.

---

edit

Edit - Indicates whether a given attribute or element should or should not be reformatted.

Value description:

Boolean: `yes` or `no`.

Default value:

`yes`.

Used in:

`<reformat>`.

---

element

Attribute - Indicates the element name which should or should not be reformatted.

Value description:
  Name of the element to be or not to be reformatted.

Default value:
  Empty string.

Used in:
  `<reformat>`.

---

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>`, `<trans-unit>`, `<target>`, `<alt-trans>`.

---

extradata

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

Value description:
  Text.

Default value:
  Undefined.

Used in:
  `<group>`, `<trans-unit>`, `<alt-trans>`.

---

extype

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

Value description:

Text.

Default value:

Undefined.

Used in:

`<group>`, `<trans-unit>`, `<alt-trans>`.

---

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>` element.

Value description:

Name of the font and its size, weight separated by a semi-colon.

Default value:

Undefined.

Used in:

`<group>`, `<trans-unit>`, `<target>`, `<alt-trans>`.

---

form

Format - Describes the type of format used in an `<internal-file>` element.

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] document: the MIME specification.

Default value:

`text`.

Used in:

`<internal-file>`.

from

From - Indicates the author of a `<note>` element.

Value description:

Text.

Default value:

Undefined.

Used in:

`<note>`.

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>`, `<trans-unit>`, `<alt-trans>`.

href

Hypertext reference - The location of the file or the URL for an `<external-file>` element.

Value description:

Text.

Default value:

Undefined.

Used in:

`<external-file>`.

---

id

Identifier - The `id` attribute is used in many elements, as a unique reference to the original corresponding code data or format for the given element. the `id` element is not the identifier of a resource: use `resname` for that purpose.

Value description:

Alpha-numeric without spaces.

Default value:

Undefined.

Used in:

`<group>`, `<trans-unit>`, `<bin-unit>`, `<g>`, `<x/>`, `<bx/>`, `<ex/>`, `<bpt>`, `<ept>`, `<it>`, `<ph>`.

---

job-id

Job ID - The identifier given to the localization job.

Value description:

Text.

Default value:

Undefined.

Used in:

`<phase>`.

---

match-mandatory

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

Value description:

Boolean: `yes` or `no`.

Default value:

`no`.

Used in:

`<context>`.

match-quality

Match quality - The match quality of the `<alt-trans>` element. This value 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>`.

maxheight

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

Value description:

Number.

Default value:

Undefined.

Used in:

`<trans-unit>`.

maxbytes

Maximum bytes - The maximum number of bytes for the `<target>` of a `<trans-unit>`. 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:

`<trans-unit>`.

---

maxwidth

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

Value description:

Number.

Default value:

Undefined.

Used in:

`<trans-unit>`.

---

menu

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

Text.

Default value:

Undefined.

Used in:

`<group>`, `<trans-unit>`, `<alt-trans>`.

---

menu-name

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

Value description:

Text.

Default value:

Undefined.

Used in:

`<group>`, `<trans-unit>`, `<alt-trans>`.

menu-option

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

Value description:

Text.

Default value:

Undefined.

Used in:

`<group>`, `<trans-unit>`, `<alt-trans>`.

mid

Marker ID - Identifier for an `<mrk>` element.

Value description:

Text.

Default value:

Undefined.

Used in:

`<mrk>`.

mime-type

Mime type - Indicates the type of a binary object. 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] document: the MIME specification.

Default value:

Undefined.

Used in:

`<bin-unit>`, `<bin-target>`.

minheight

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

Value description:

Number.

Default value:

Undefined.

Used in:

`<trans-unit>`.

minbytes

Minimum bytes - The minimum number of bytes for the `<target>` of a `<trans-unit>`. 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:

`<trans-unit>`.

minwidth

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

Value description:

Number.

Default value:

Undefined.

Used in:

<trans-unit>.

---

mtype

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

Value description:

Text. The recommended values for the mtype attribute are defined in the XLIFF Values schema as the mtypeValueList type.as follow (this list is not exhaustive):
- abbrev = abbreviation, acronym, etc.
- datetime = date or time information.
- name = proper or common name.
- phrase = sub-sentence level.
- protected = text that should remain untouched during the process.
- term = one or more words of a terminology entry.

Default value:

Undefined.

Used in:

<mrk>.

---

name

Name - The name attribute specifies the user-defined name of a named group element.

Value description:

Text.

Default value:

Undefined.

Used in:

<prop-group>, <context-group>, <count-group>.

---

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>.

---

original

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

Value description:

Text.

Default value:

Undefined.

Used in:

<file>.

---

phase-name

Phase Name - The `phase-name` attribute allows to name uniquely a <phase> element. It is also used in other elements in the file to refer to the given <phase> element.

Value description:

Text.

Default value:

Undefined.

Used in:

<phase>, <trans-unit>, <target>, <bin-unit>, <bin-target>.

---

pos

Position - Indicates whether an isolated tag <it> is a beginning or and ending tag.

Value description:

`open` or `close`.

Default value:

Undefined.

Used in:

`<it>`.

---

priority

Priority - The priority of a <note> element.

Value description:

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

Default value:

1

Used in:

`<note>`.

---

process-name

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

Value description:

Text.

Default value:

Undefined.

Used in:

`<phase>`.

---

product-name

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

Value description:

Text.

Default value:

Undefined.

Used in:

`<file>`.

---

product-version

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

Value description:

Alpha-numeric.

Default value:

Undefined.

Used in:

`<file>`.

---

prop-type

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

Value description:

Text. No value defined by the standard.

Default value:

Undefined.

Used in:

`<prop>`.

---

reformat

Reformat - Indicates whether the target can be formatted (size, font, etc.) different than the source.

[NOTE: We need to decide what to do with this attribute, if we use <reformat> instead.]

Value description:

Boolean: `yes` or `no`.

Default value:

yes.

Used in:

<trans-unit>, <bin-unit>.

---

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>. <trans-unit>, <alt-trans>, <target>, <bin-unit>, <bin-target>.

---

restype

Resource type - Indicates the type of the container element.

Value description:

Text. The recommended values for the `restype` attribute are defined in the XLIFF Values schema as the `restypeValueList` type.as follow (this list is not exhaustive):
- `button` = Button in UI.
- `caption` = Title in UI, caption in documentation, alternate text, etc.
- `checkbox` = Check box in UI.
- `cell` = Text in a table cell.
- `dialog` = Dialog box in UI.
- `file` = Filename, path.
- `fn` = Footnote.
- `footer` = Footer text.
- `font` = Font name.
- `frame` = Frame or window, or any generic group of components.
- `header` = Header text.
- `heading` = Title or header-type segment.
- `keywords` = List of keywords, enumeration within a paragraph, etc.
- `label` = Static text, label in UI, etc.
- `listitem` = Paragraph in a list, entry in a list box, etc.
- `menu` = Menu.
- `menuitem` = Entry in a UI menu.

- `message` = Prompt, error or warning message.
- `radio` = Radio button in UI.
- `shortcut` = Windows accelerators, shortcuts in resource or property files.
- `string` = Generic text from source code, string table, etc.
- `var` = Variable.

Default value:

Undefined.

Used in:

`<group>`, `<trans-unit>`, `<target>`, `<alt-trans>`, `<bin-unit>`, `<bin-target>`.

---

rid

Reference identifier - The `rid` attribute is used to link different elements that are related. For example, a reference to its definition, or paragraphs belonging to the same group, etc.

Value description:

Alpha-numeric without spaces.

Default value:

Undefined.

Used in:

`<bpt>`, `<ept>`, `<it>`, `<bx/>`, `<ex/>`.

---

size-unit

Unit of size attributes - The `size-unit` attribute specifies the units of measure used in the `maxheight`, `minheight`, `maxwidth`, and `minwidth` attributes.

Value description:

Text. The recommended values for the `size-unit` attribute are defined in the XLIFF Values schema as the `size-unitValueList` type.as follow (this list is not exhaustive):
- `pixel` = Pixel.
- `byte` = 8-bit byte.
- `char` = Unicode character.

Default value:

`pixel`.

Used in:

`<trans-unit>`.

---

source-language

Source language - The language for the `<source>` elements in the given `<file>` element.

Value description:

A language code as described in the [RFC 3066]. The values for this attribute follow the same rules as the values for `xml:lang`. 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, and the erratum E11 (which replaces RFC 1766 by RFC 3066).

The source language can be also specified by `xml:lang` in each `<source>` element. The value of `source-language` and `xml:lang` can be different to allow having different source languages if necessary (for example in an `<alt-trans>` element).

Default value:

Undefined.

Used in:

`<file>`.

---

state

State - The status of a particular translation in a `<target>` or `<bin-target>` element.

Value description:

Text. The recommended values for the `state` attribute are defined in the XLIFF Values schema as the `stateValueList` type.as follow (this list is not exhaustive):
- `needs-translation` = The item needs to be translated.
- `needs-review` = The item needs to be reviewed.
- `needs-resizing` = The item needs to be resized or reformatted.

Default value:

Undefined.

Used in:

`<target>`, `<bin-target>`.

---

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>`, `<trans-unit>`, `<target>`, `<alt-trans>`.

---

target-language

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

Value description:

A language code as described in the [RFC 3066]. The values for this attribute follow the same rules as the values for `xml:lang`. 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, and the erratum E11 (which replaces RFC 1766 by RFC 3066).

The target language can be also specified by `xml:lang` in each `<target>` element. The value of `target-language` and `xml:lang` can be different to allow having different target languages if necessary (for example in an `<alt-trans>` element).

Default value:

Undefined.

Used in:

`<file>`.

---

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 is DEPRECATED in version 1.1. Instead, use the `<tool>` element and a tool-id attribute.

Value description:

Text

Default value:

`manual`.

Used in:

`<file>`, `<phase>`, `<alt-trans>`.

---

tool-company

Tool company - The `tool-company` attribute allows to specify the company from which a tool originates.

Value description:

Text

Default value:

Undefined.

Used in:

`<tool>`.

---

tool-id

Tool identifier - The `tool-id` attribute allows to identify uniquely a `<tool>` element. It is also used in other elements in the file to refer to the given `<tool>` element.

Value description:

Text

Default value:

Undefined.

Used in:

`<file>`, `<phase>`, `<alt-trans>`, `<tool>`.

---

tool-name

Tool name - The `tool-name` attribute allows to specify the name of a given tool.

Value description:

Text

Default value:

Undefined.

Used in:

`<tool>`.

---

tool-version

Tool version - The `tool-version` attribute allows to specify the version of a given tool.

Value description:
Text

Default value:
Undefined.

Used in:
`<tool>`.

---

translate

Translate - Indicates whether or not the text referred to should be translated.

Value description:
Boolean: `yes` or `no`.

Default value:
`yes`.

Used in:
`<trans-unit>`, `<bin-unit>`.

---

ts

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

Important: The `ts` attribute is 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>`, `<group>`, `<trans-unit>`, `<source>`, `<target>`, `<bin-unit>`, `<bin-source>`, `<bin-`

target>, <alt-trans>, <mrk>, <g>, <x/>, <bx/>, <ex/>, <bpt>, <ept>, <ph>, <it>.

---

uid

Unique ID - The unique id used to identify the skeleton file.

Value description:
Text.

Default value:
Undefined.

Used in:
<external-file>.

---

unit

Unit - The units counted in a <count> element.

Value description:
Text. The recommended values for the `unit` attribute are defined in the XLIFF Values schema as the `unitValueList` type.as follow (this list is not exhaustive):
- `word` = standard words.
- `page` = pages.
- `trans-unit` = number of <trans-unit>.
- `bin-unit` = number of <bin-unit>.
- `item` = number of <trans-unit> and <bin-unit>.

Default value:
Undefined.

Used in:
<count>.

---

version

XLIFF version - The `version` attribute is used to specify the format version of the XLIFF document.

Value description:
Fixed text.

Default value:

1.01

Used in:

<xliff>.

3.3.2. XML Namespace Attributes

---

xml:lang

Language - The xml:lang attribute specifies the locale language variant of the text of a given element.

Value description:

A language code as described in the [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, and the erratum E11 (which replaces RFC 1766 by RFC 3066).

Default value:

Undefined.

Used in:

<xliff>, <note>, <prop>, <source>, <target>, <alt-trans>.

---

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 applications' 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.

Default value:

default.

Used in:

<file>, <group>, <trans-unit>, <alt-trans>.

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.

[NOTE: The tree has been completely redone. Please double check it.]

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

<xliff>1
|
+--- <file>+
     |
     +--- <header>?
     |    |
     |    +--- <skl>?
     |    |    |
     |    |    +--- (<internal-file> | <external-file>)1
     |    |
     |    +--- <phase-group>?
     |    |    |
     |    |    +--- <phase>+
     |    |         |
     |    |         +--- <note>*
     |    |
     |    +--- <glossary>*
     |    |    |
     |    |    +--- (<internal-file> | <external-file>)1
     |    |
     |    +--- <reference>*
     |    |    |
     |    |    +--- (<internal-file> | <external-file>)1
     |    |
     |    +--- <count-group>*
     |    |    |
     |    |    +--- <count>*
     |    |
     |    +--- <tool>*
     |    |    |
     |    |    +--- [Extension Point]
     |    |
     |    +--- <prop-group>*
     |    |    |
     |    |    +--- <prop>*
     |    |
     |    +--- [Extension Point]
     |    |
     |    +--- <note>*
     |
     +--- <body>1
          |
          +--- <group>*
          |    |
          |    +--- <context-group>*
          |    |    |
          |    |    +--- <context>+
          |    |
          |    +--- <count-group>*
          |    |    |
          |    |    +--- <count>*
          |    |
          |    +--- <prop-group>*
          |    |    |
          |    |    +--- <prop>*
          |    |
          |    +--- [Extension Point]
          |    |
          |    +--- <note>*
          |    |
          |    +--- At least one of: (<group>* <trans-unit>* <bin-unit>*)
          |
          +--- <trans-unit>*
          |    |
          |    +--- <source>1
          |    |    |
          |    |    +--- [Inline Elements]
          |    |
          |    +--- <target>?
          |    |    |
          |    |    +--- [Inline Elements]
          |    |
          |    +--- <context-group>*
          |    |    |
          |    |    +--- <context>+
          |    |
          |    +--- <count-group>*
          |    |    |
          |    |    +--- <count>*
          |    |
          |    +--- <prop-group>*
          |    |    |
          |    |    +--- <prop>*
          |    |
          |    +--- [Extension Point]
          |    |
          |    +--- <note>*
          |    |
          |    +--- <alt-trans>*
          |         |
          |         +--- <context-group>*
          |         |    |
          |         |    +--- <context>+
          |         |
          |         +--- <source>?
          |         |    |
          |         |    +--- [Inline Elements]
```

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

Inline Elements:

```
---+--- <ph>*
   |    |
   |    +--- <sub>*
   |         |
   |         +--- [Inline Elements]
   |
   +--- <it>*
   |    |
   |    +--- <sub>*
   |         |
   |         +--- [Inline Elements]
   |
   +--- <bpt>*
   |    |
   |    +--- <sub>*
   |         |
   |         +--- [Inline Elements]
   |
   +--- <ept>*
   |    |
   |    +--- <sub>*
   |         |
   |         +--- [Inline Elements]
   |
   +--- <x/>*
   |    |
   |    +--- [Inline Elements]
   |
   +--- <bx/>*
   |    |
   |    +--- [Inline Elements]
   |
   +--- <ex/>*
   |    |
   |    +--- [Inline Elements]
   |
   +--- <mrk>*
        |
        +--- [Inline Elements]
```

B. Document Type Definition and Schemasfor XLIFF

- The document type definition file for XLIFF is available at: http://www.oasis-open.org/committees/xliff/documents/xliff.dtd.
- The XML schema for the XLIFF Core is available at: http://www.oasis-open.org/committees/xliff/documents/xliff-1-1.xsd
- The XML schema for the XLIFF Values is available at: http://www.oasis-open.org/committees/xliff/documents/xliff-values-1-1.xsd

C. Changes Since Previous Version (Non-Normative)

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

[This is the list of the changes made in this document and that we will need to voted on]

- Updated all references to version from 1.0 to 1.1.
- Updated the description of naming convention to explain better the apparent discrepancies.
- Renamed the "Naming Convention" section to "Naming Guidelines" and moved it as a non-normative appendix.
- Changed description for `xml:lang`.
- Added to the description of `source-language`.
- Added to the description of `target-language`.
- Changed namespace identifier to "`urn:oasis:names:tc:xliff:document:1.1`".
- Deprecated the `<prop-group>` element.
- Deprecated the `<prop>` element.
- Deprecated the `ts` attribute.
- Added the Extensibility section.

- Updated the contribution list to reflect the current TC,
- Made the `<header>` element optional.
- Added the reference the XLIFF schemas.
- Updated the XLIFF Tree Structure appendix.
- Changed the definition of the content of `<group>` to include at least one of `<group>`, `<trans-unit>`, `<bin-unit>` elements.
- Added paragraph about validation with schemas in the XML Declaration section.
- Changed in various attributes definitions the description of the list of values so they refer to the schema rather than this document. Also changed the text about "recommended values" where appropriate elsewhere (the values are not just merely recommended any more).
- Changed the sequence of the elements in `<header>`, `<group>`, `<trans-unit>`, `<alt-trans>`, and `<bin-unit>`.
- Removed the references to `xliff-show-context` and `xliff-update-count` reserved processing instructions.
- Added the `<tool>` element and its corresponding attributes.
- Deprecated the `tool` attribute.
- Added the `tool-id` attribute in the `<file>`, `<phase>` and `<alt-trans>` elements.
- Added the `<reformat>` element and its corresponding attributes.
- Added the Embedding XLIFF section.

D. Naming Guidelines (Non-Normative)

The following naming conventionsguidelines were used in writing this specification.

D.1. Elements and Attributes

The following conventionsguidelines were used for element and attribute naming.

1. Standard English letters.
2. Lower case only.
3. Hyphen, '-', may or may not be used for concatenation.Hyphen is the preferred mean for creating composite names.
4. Elements and attributes should not have the same name, even attributes of different elements.
5. Attribute names must be consistently defined throughout.
6. Industry standard terminology should be followed where possible.

D.2. Attribute Values

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

The specification also recommends the use of the semi-colon as a concatenation separator for values. For example, multiple contacts may be listed for a `<file>` with the attribute-value written thusly: `contact-name="Peter;Mark"`.

D.3. Processing Instructions

XLIFF reserves processing instructions that begin with "`xliff-`" for definition at some future time.

This specification has recommended `xliff-show-context-group` and `xliff-show-context` as a means of displaying context information to anyone processing a XLIFF file. Also, the processing instruction `xliff-update-count-group` has been given as an example.

D.4. XLIFF File Extension

XLIFF documents use the `.xlf` extension. This conforms to a 8.3 standard name. No other extension is recommended by the specification.

E. XLIFF Technical Committee (Non-Normative)

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

[Need to get the latest updated list]

- Gérard Cattin des Bois, Microsoft
- Jonathan Clark, Lionbridge - (Vice-Chair)
- John Corrigan, Sun Microsystems
- Mirek Driml, Moravia-IT
- Ian Dunlop, Novell
- Fiona Ebbs, Novell
- Tony Jewtushenko, Oracle - (Chair)
- Milan Karásek, Moravia-IT
- Caroline Koff, HP
- David Leland [??]
- Mark Levins, IBM/Lotus
- Christian Lieske, SAP
- Mat Lovatt, Oracle
- Enda McDonnell
- Mike McKenna, CommerceOne [??]
- Ultan Ó Broin, Oracle [??]
- David Pooley, SDL
- David Ramsey, Xerox [??]
- John Reid, Novell
- Peter Reynolds - (Secretary)
- François Richard, Xerox [??]
- Yves Savourel, RWS Group - (Editor)
- Bryan Schnabel, Tektronix
- Reinhard Schäler

E. References

Normative

[IANA Charsets]
IANA Names for Character Sets. IANA (Internet Assigned Numbers Authority), Aug 2001

[ISO 639]
Codes for the Representation of Names of Languages. ISO (International Standards Organization), Nov 2001.

[ISO 3166]
Codes for the representation of names of countries and their subdivisions. ISO (International Organization for Standardization), Jun 2000.

[ISO 8601]
Representation of dates and times. ISO (International Organization for Standardization), Dec 2000.

[OpenTag 1.2]
OpenTag Format Specifications. ILE (International Language Engineering), Nov 1998.

[RFC 1341]
Multipurpose Internet Mail Extensions. IETF (Internet Engineering Task Force), Jun 1992.

[RFC 3066]
RFC 3066 Tags for the Identification of Languages. IETF (Internet Engineering Task Force), Jan 2001.

[TMX 1.3]
TMX Format Specifications. LISA (Localisation Industry Standards association), Aug 2001.

[XML 1.0]
Extensible Markup Language (XML) 1.0 Second Edition. W3C (World Wide Web Consortium), Oct 2000.

[XML Names]
Namespaces in XML. W3C (World Wide Web Consortium), Jan 1999.

Non-Normative

[ISO]
International Organization for Standardization Web site.

[LISA]
Localisation Industry Standards Association Web site.

[OASIS]
Organization for the Advancement of Structured Information Standards Web site.

[Unicode]
Unicode Consortium Web site.

[W3C]
World Wide Web Consortium Web site.