INTERNATIONAL STANDARD

ISO 24616

First edition
2012-09-01

Language resources management — Multilingual information framework

Gestion des ressources langagières — Plateforme d'informations multilingues

Reference number
ISO 24616:2012(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights.

ISO 24616 was prepared by Technical Committee ISO/TC 37, *Terminology and other language and content resources*, Subcommittee SC 4, *Language resource management*.

Language resources management — Multilingual information framework

1 Scope

This International Standard provides a generic platform for modelling and managing multilingual information in various domains: localization, translation, multimedia annotation, document management, digital library support, and information or business modelling applications. MLIF (multilingual information framework) provides a metamodel and a set of generic data categories [ISO 12620:2009] for various application domains. MLIF also provides strategies for the interoperability and/or linking of models including, but not limited to, XLIFF, TMX, smilText and ITS.

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO 12620:2009; *Terminology and other language and content resources — Specification of data categories and management of a Data Category Registry for language resources*

ISO 8879, *Information processing — Text and office systems —Generalized Markup Language (SGML)*

Extensible Markup Language. Fifth Edition, T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, F. Yergeau Editors, W3C Recommendation, 26 November 2008, http://www.w3.org/TR/xml

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply:

3.1
adornment
data category attached to a component of a metamodel

3.2
inline code
inline instructions inserted in a source document

Note to entry: Native code can, for instance, provide presentational instructions (e.g. HTML codes).

3.3
subtitle
textual versions of the dialog in films, television programs, video games, etc., usually displayed at the bottom of the screen

3.4
working language
language in which linguistic sequences are expressed

4 Specification principles

4.1 Key standard used in the specification: Unified Modeling Language (UML)

The MLIF specification complies with the modelling principles of UML as defined by the Object Management Group (OMG) [UML]. The specification uses the UML subset that is relevant for the purposes of MLIF.

4.2 Metamodel and adornment

In line with Terminological Markup Framework (TMF) as defined in ISO 16642, MLIF defines a metamodel that is adorned by data categories, as defined in ISO 12620.

4.3 XML serialization

Associated with the metamodel and its adornment, MLIF proposes a representation in XML called “XML serialization”, in line with Extensible Markup Language (XML) as defined in ISO 8879.

5 Metamodel specification

The MLIF metamodel is specified in the UML object diagram in Figure 1.

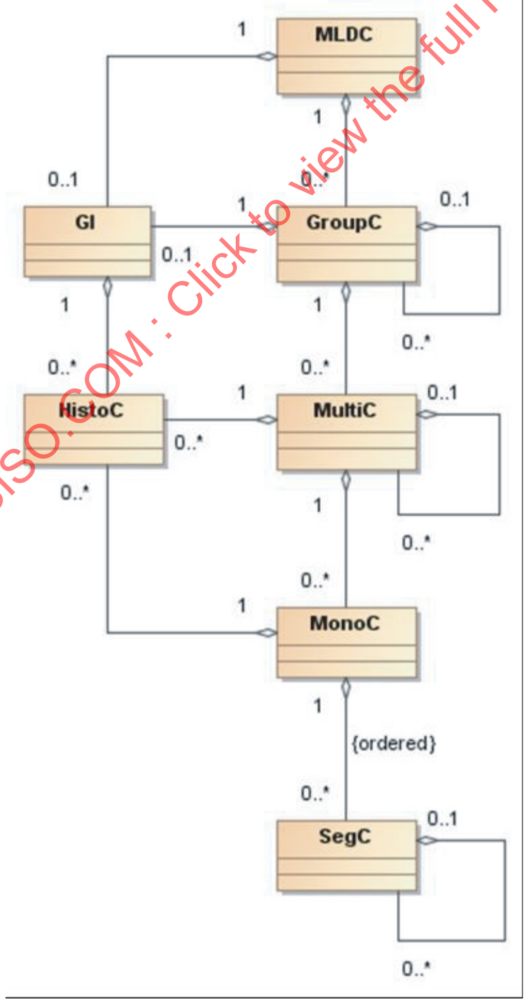

Figure 1 — MLIF metamodel

The MLIF metamodel is defined by the following seven "core components". These components are listed as follows, according to their XML serialization:

- <MLDC> (Multilingual Data Collection), which represents a collection of data containing global information and several multilingual units;
- <GI> (Global Information), which represents technical and administrative information applying to the entire multilingual data collection;
- <GroupC> (Grouping components), which represents a sub-collection of multilingual data that have a common origin or purpose within a given project;
- <MultiC> (Multilingual Component), which groups together all variants of a given textual content;
- <MonoC> (Monolingual Component), which groups together information related to one language and is part of a multilingual component (MultiC);
- <HistoC> (History Component), which traces modifications to the component to which it is anchored (i.e. versioning);
- <SegC> (Segmentation Component), which allows any level of segmentation for textual information, possibly in a recursive manner.

6 MLIF compliance

Any format compliant with this International Standard may use the MLIF metamodel in two possible ways:

- by fully implementing the MLIF metamodel starting at the level of <MLDC>;
- by specifically embedding MLIF-compliant information within another model, by implementing one of the lower level MLIF elements, namely <GroupC>, <MultiC> or <MonoC>.

7 Metamodel adornment

7.1 Introduction

The MLIF XML serialization proposes a set of XML elements and XML attributes, which are described in the following sections, where the characters “<” and “>” delimit the name of the element. Following the TEI guidelines (http://www.tei-c.org), some attributes are specified by means of a class attribute, with the convention that the name of the class attribute is prefixed by “att.” (e.g. “att.xlink”). The other XML attributes are listed with the convention that two quotes delimit the name of the attribute (e.g. “xml:lang”). The specifications in Annex G shall be applied.

7.2 General principles concerning the use of W3C generic attributes

The following W3C attributes are to be used by all MLIF-compliant applications:

- the attribute xml:lang shall be used in accordance with W3C recommendations to represent the working language of any relevant element and, in particular, shall be used systematically for any implementation of MonoC;
- the attribute xml:id shall be used in accordance with W3C recommendations to provide a unique identifier to an element of the MLIF metamodel.

7.3 Recommended adornment for GI

- <domain>
- <project>
- <source>
- <sourceType>
- <sourceLanguage>
- <sourceFormat>
- <targetLanguage>
- <formatVersion>
- <legalStatus>
- <creationTool>
- <creationToolVersion>
- <creationDate>
- <creationIdentifier>
- <changeDate>
- <changeIdentifier>

7.4 Recommended adornment for GroupC

- <groupType>

7.5 Recommended adornment for MultiC

- <class>
- <changeDate>
- <changeIdentifier>
- <creationTool>
- <creationToolVersion>
- <creationIdentifier>
- <creationDate>
- <translationStatus>
- <matchQuality>

7.6 Recommended and mandatory adornment for MonoC

- att.lang
- <translationRole>
- <segmentation>
- att.xlink

The language attribute is mandatory on MonoC. All other adornments are optional.

7.7 Recommended adornment for SegC

- <translationRole>
- <beginPairedTag>
- <endPairedTag>
- <genericGroupPlaceholder>
- <placeholder>
- <genericPlaceholder>
- <translate>
- att.linguistic
- att.xlink

7.8 Recommended adornment for HistoC

The HistoC component is a generic component that traces modifications made on the component to which it is anchored (e.g. creation, modification and validation). In the MLIF metamodel, the HistoC component may be anchored to the GI, MultiC or MonoC component. This makes it possible for all evolutions of, or enhancements to, the component to be recorded.

HistoC may be adorned by four elements:

- <author>
- <version>
- <transaction>
- <date>

7.9 Recommended online annotation adornment

Multilingual text documents are often only one stage in a complex workflow that involves external document sources in a wide variety of formats. From these, it is often necessary to keep inline markup indicating the presentational features that have to be retained in a translated target document. To this end, MLIF-compliant applications should use the following elements, in relation to the <SegC> element, that map onto similar subsets in TMX and XLIFF:

- <beginPairedTag>
- <endPairedTag>
- <genericGroupPlaceholder>
- <genericPlaceholder>
- <placeholder>

7.10 Recommended adornment for localization

All the following elements should be used to provide localization-related information:

- <translationRole>
- <translationStatus>

7.11 Recommended adornment for internationalization

- <translate>

7.12 Recommended adornment for temporal synchronization

The following elements should be used when textual content has to be conveyed (in written or spoken form) together with some constraints:

- <duration>
- <begin>
- <next>

8 Relation with other standards

As with the “Terminological Markup Framework” TMF [ISO 16642] in terminology, MLIF introduces a metamodel that combines with selected data categories as a way of ensuring interoperability between several multilingual applications and corpora. MLIF deals with multilingual corpora, multilingual fragments, and the translation relations between them. In each domain where MLIF is applicable, a specific granularity may be considered for segmentation and description. These two last processes may rely on MAF [ISO 24611], SynAF [ISO 24615] and TMF for morphological description, syntactical annotation and terminological description respectively.

MLIF supports the construction and the interoperability of localization and translation memories resources, and also deals with the description of a metamodel for multilingual content. MLIF does not propose a closed list of description features. Rather, it provides a list of data categories that is much easier to update and extend. This list represents a point of reference for multilingual information in the context of various application scenarios.

However, MLIF not only describes elementary linguistic segments (e.g. sentence, syntactical fragment, word and part of speech), but may also be used to represent document structure (e.g. title, abstract, paragraph and section). In addition, MLIF allows for external and internal links (annotations and references).

MLIF is designed to provide a common framework that facilitates the interoperability with formats such as TMX (LISA OSCAR) and XLIFF (OASIS). MLIF can be seen as a parent of these formats, since both of them deal with multilingual data expressed in the form of segments or text units. Both can be stored, manipulated and translated in a similar manner.

Examples of using MLIF are given in Annexes A to F.

Annex A
(informative)

Example using MLIF for Computer-Assisted Translation (CAT)

The main reason for lemma, part-of-speech and morphological features is to allow CAT tools based on translation memory to produce translations of new words and sentences that are not in the translation database.

For example, using a translation memory that contains the English sentence "The meal is nice." and its translation in French "Le repas est bon.", current CAT tools such as SDL TRADOS[1] Translator's Workbench are not able to provide the predicted translation for the sentence "The meals are nice." even though the word lemmas of "The meal is nice." and "The meals are nice." are matching. This weakness is due to the fact that these tools use limited linguistic criteria during the translation process.

The data produced by TRADOS Translator's Workbench is as follows:

```
<tmx version="1.4">
 <header
   creationtool="TRADOS Translator's Workbench for Windows"
   creationtoolversion="Edition 8 Build 863"
   segtype="sentence"
   o-tmf="TW4Win 2.0 Format"
   adminlang="EN-US"
   srclang="EN-GB"
   datatype="rtf"
   creationdate="20100528T144322Z"
   creationid="USER"/>
 <body>
  <tu creationdate="20100528T144322Z" creationid="USER">
   <tuv xml:lang="EN-GB">
    <seg>The meal is nice.</seg>
   </tuv>
   <tuv xml:lang="FR-FR">
    <seg>Le repas est bon.</seg>
   </tuv>
  </tu>
 </body>
</tmx>
```

To translate the sentence "The meals are nice.", an MLIF-compliant tool should implement the following procedure:

Step-1 Represent in MLIF and add linguistic properties to all the words within the translation memory.

Step-2 Run a part-of-speech tagger on the sentence in order to obtain the right morphosyntactic word categories.

Step-3 Translate the lemmas using an English-to-French bilingual lexicon.

[1] SDL TRADOS Translator's Workbench is an example of a suitable product available commercially. This information is given for the convenience of users of this International Standard and does not constitute an endorsement by ISO of this product.

Step-4 Consult a French lexicon of inflected forms in order to retrieve the correct inflected form using the lemma and morphological features.

Step-5 Generate the translation of "The meals are nice." by substituting each English word with its French inflected form as follows:

"The meals are nice." => "Les repas sont bons."

The XML data will include a feature structure declaration defining a tagset (e.g. for "nS"), with a word segmentation and tagset defined in MAF:

```
<MLDC xmlns="http://www.tei-c.org/ns/1.0">
 <tei:fLib>
  <tei:f xml:id="nS" name="grammaticalNumber" fVal="singular"/>
  <tei:f xml:id="gM" name="grammaticalGender" fVal="masculine"/>
  <tei:f xml:id="mP" name="verbFormMood" fVal="present"/>
  <tei:f xml:id="p1" name="person" fVal="thirdPerson"/>
  <tei:f xml:id="nS" name="grammaticalNumber" fVal="singular"/>
 </fLib>
 <GroupC>
  <MultiC>
   <creationIdentifier>SEMMAR</creationIdentifier>
   <creationDate>20090922T140653Z</creationDate>
   <MonoC xml:lang="en">
    <SegC>The meal is nice.</SegC>
   </MonoC>
   <MonoC xml:lang="fr">
    <SegC>Le repas est bon.</SegC>
   </MonoC>
  </MultiC>
  <MultiC class="translation">
   <MonoC xml:lang="en">
    <SegC class="word" lemma="the" pos="definiteArticle">The</SegC>
    <SegC
      class="word"
      lemma="meal"
      pos="commonNoun"
      tag="#nS">meal</SegC>
    <SegC
      class="word"
      lemma="be"
      pos="verb"
      tag="#mP #p1 #nS">is</SegC>
    <SegC class="word" lemma="nice" pos="qualifierAdjective">nice</SegC>
    <SegC class="word" lemma="." pos="mainPunctuation">.</SegC>
   </MonoC>
   <MonoC xml:lang="fr">
    <SegC
      class="word"
      lemma="le"
      pos="definiteArticle"
      tag="#gM #nS">Le</SegC>
    <SegC
      class="word"
      lemma="repas"
      pos="commonNoun"
      tag="#gM #nS">repas</SegC>
    <SegC
      class="word"
      lemma="être"
      pos="verb"
      tag="#mP #p1 #nS">est</SegC>
    <SegC
      class="word"
      lemma="bon"
      pos="qualifierAdjective"
      tag="#gM #nS">bon</SegC>
    <SegC class="word" lemma="." pos="mainPunctuation">.</SegC>
   </MonoC>
  </MultiC>
 </GroupC>
</MLDC>
```

Annex B
(informative)

Example: representing TMX data

B.1 Introduction

TMX (Translation Memory eXchange) is the vendor-neutral open XML standard for the exchange of Translation Memory (TM) data created by computer-aided translation (CAT) and localization tools. The purpose of TMX is to allow easier exchange of translation memory data between tools and/or translation vendors with little or no loss of critical data during the process. TMX, which has been on the market since 1998, is a certifiable standard format. It was developed, and is maintained by, OSCAR (Open Standards for Container/Content Allowing Re-use), a LISA Special Interest Group.

B.2 Mapping TMX to MLIF

TMX is nearly isomorphic to the MLIF metamodel. The core elements of the TMX macro-structure map to MLIF as follows:

- <tmx> maps onto the <MLDC> element;
- <header> maps onto the <GI> element;
- <body> is a container for the <tuv> element and maps onto the <GroupC> element;
- <tu> maps onto the <MultiC> element;
- <tuv> maps onto the <MonoC> element;
- <seg> maps onto the <SegC> element;
- <hi> of type term maps onto the <SegC> element of type term.

Further TMX elements and attributes map onto MLIF elements as follows:

- The "creationtool" attribute maps onto the <creationTool> element;
- The "creationdate" attribute maps onto the <creationDate> element;
- The "tuid" attribute maps onto the <creationIdentifier> element within MultiC.
- The <prop> element does not map onto any specific element as it represents a generic placeholder for application-dependent data. When applicable, a specific <prop> element is explicitly mapped onto MLIF elements or onto a standardized ISO/TC 37 data category as available from ISOCat.

B.3 Example of data

The following example, based on TMX version 1.4, focuses on the multilingual units of a TMX document and does not translate all the details of the header.

```
<tmx version="1.4">
 <header
   adminlang="en"
   creationdate="20040731T164933Z"
   creationtool="Heartsome TM Server"
   creationtoolversion="1.0.1"
   datatype="xml"
   o-tmf="unknown"
   segtype="block"
   srclang="*all*"/>
 <body>
  <tu creationdate="20020930T004233Z" tuid="1091303313515">
   <tuv xml:lang="fr">
    <seg>Le processus de <hi xml:id="X3" type="term">contrôle de
           qualité</hi> en dix étapes qu'il a créé il y a plus
         de 1300 ans est beaucoup plus complet et précis que ceux
         existant aujourd'hui.</seg>
   </tuv>
   <tuv xml:lang="en">
    <seg>His 10-stage <hi corresp="#X3" type="term">quality
           control</hi> process initiated more than 1300 years
         ago is far more thorough and exacting than any existing
         today.</seg>
   </tuv>
   <tuv xml:lang="es">
    <seg>El proceso de <hi corresp="#X3" type="term">control de
           calidad</hi> en diez pasos que inició hace más de
         1300 años es mucho más completo y preciso que los que
         existen en la actualidad.</seg>
   </tuv>
   <tuv xml:lang="it">
    <seg>Il suo metodo di <hi corresp="#X3" type="term">controllo di qualità</hi> in 10 fasi risale a più
         di 1300 anni fa ed è molto più accurato e preciso di
         qualsiasi metodo attuale.</seg>
   </tuv>
   <tuv xml:lang="ko">
    <seg>그가 1300 여년 전 시작한 10 단계 <hi corresp="#X3" type="term">품질
           관리</hi> 방법은 현존하는 것보다 훨씬 더 철저하고 정확하다.</seg>
   </tuv>
  </tu>
 </body>
</tmx>
```

The corresponding representation in MLIF default representation is as follows:

```
<MLDC>
 <GI>
  <sourceFormat>TMX</sourceFormat>
  <formatVersion>1.4</formatVersion>
  <creationDate>20040731T164933Z</creationDate>
  <creationTool>Heartsome TM Server</creationTool>
  <creationToolVersion>1.0.1</creationToolVersion>
 </GI>
 <GroupC>
  <MultiC>
```

```
   <creationIdentifier>1091303313515</creationIdentifier>
   <creationDate>20020930T004233Z</creationDate>
   <MonoC xml:lang="fr">
    <SegC>Le processus de <SegC xml:id="X3" type="term">contrôle
         de qualité</SegC> en dix étapes qu'il a créé il y a
       plus de 1300 ans est beaucoup plus complet et précis que
       ceux existant aujourd'hui.</SegC>
   </MonoC>
   <MonoC xml:lang="en">
    <SegC>His 10-stage <SegC corresp="#X3" type="term">quality
         control</SegC> process initiated more than 1300
       years ago is far more thorough and exacting than any
       existing today.</SegC>
   </MonoC>
  </MultiC>
 </GroupC>
</MLDC>
```

B.4 Example of TMX and MLIF interaction

Figure B.1 illustrates the interaction between TMX and MLIF. This process involves subsequent steps of extraction, translation and merging. The process begins with a TMX document containing linguistic content in English (en) and German (de). The extraction process (1) generates a “Skeleton File” (2) containing all TM formatting information, and an MLIF Document Linguistic Content (3) in which only relevant linguistic information is stored. As most translators (human beings or automatic software modules) work with TMX software-oriented tools, an XSL style-sheet makes it possible to transform an MLIF document into a TMX document. This file does not contain any formatting information. Once the translator has added the appropriate Japanese (ja) translation, another XSL style-sheet transforms the TMX document into an MLIF document (4). Finally, the new MLIF document (containing the Japanese translation) is merged with the “Skeleton File” to produce a new TMX formatted document (5).

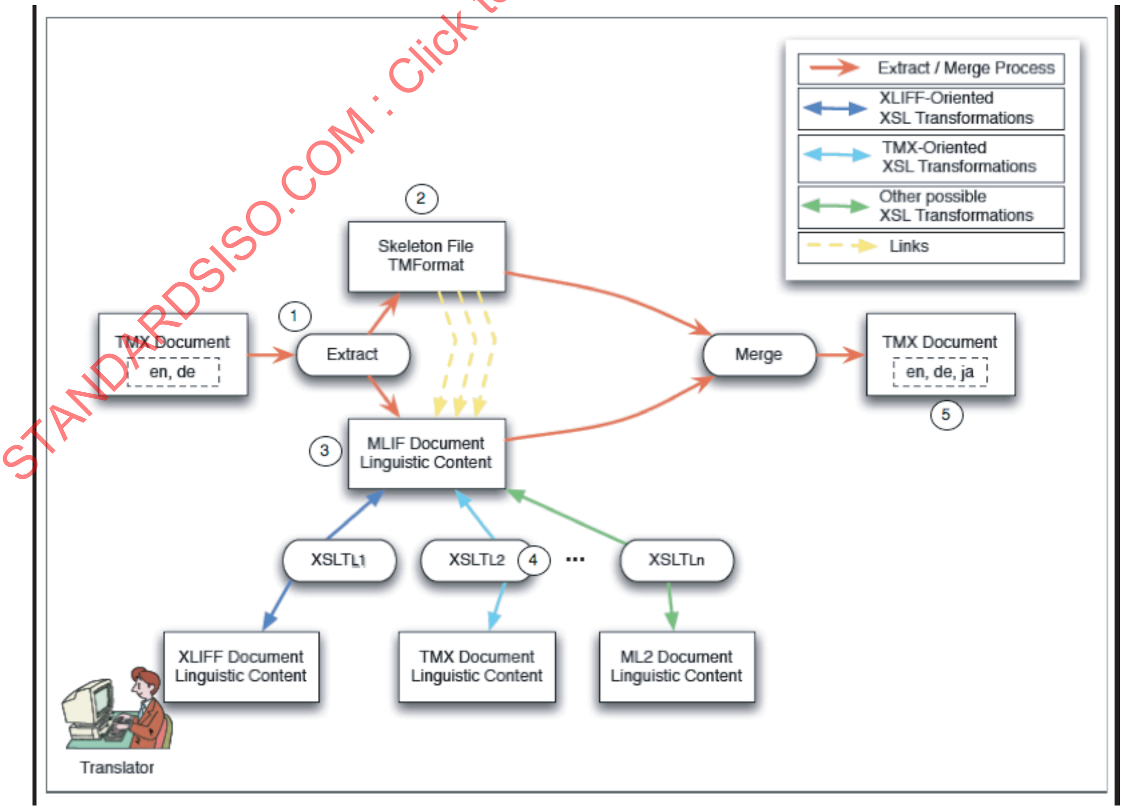
Extract / Merge Process
XLIFF-Oriented XSL Transformations
TMX-Oriented XSL Transformations
Other possible XSL Transformations
Links
2
Skeleton File TMFormat
1
TMX Document
en, de
Extract
Merge
TMX Document
en, de, ja
5
3
MLIF Document Linguistic Content
XSLTL1
XSLTL2
4
...
XSLTLn
XLIFF Document Linguistic Content
TMX Document Linguistic Content
ML2 Document Linguistic Content
Translator

Figure B.1 — TMX and MLIF interaction

Annex C
(informative)

Example of XLIFF data representation

C.1 Introduction

The purpose of the XLIFF is to define and promote the adoption of a specification for the interchange of localizable software- and document-based objects and related metadata.

C.2 Mapping XLIFF to MLIF

XLIFF differs from the MLIF metamodel in that it draws a clear distinction between source and target language for monolingual information. This is handled through the appropriate use of the <translationRole> data category in <MonoC> together with the language declarations (<sourceLanguage> and <targetLanguage>) in <GI>.

The core elements of the XLIFF macro-structure map to MLIF as follows:

- <xliff> maps onto the <MLDC> element;
- <header> maps onto the <GI> element;
- <body> is a container for the <tuv> element and maps onto the <GroupC> element;
- the <phase> element maps onto the <HistoC> element;
- <trans-unit> maps onto the <MultiC> element;
- <source> maps onto the <MonoC> element and simultaneously sets the value of the <translationRole> element to <sourceLanguage>. The corresponding textual content is placed in a <SegC> element;
- <target> maps onto the <MonoC> element and simultaneously sets the value of the <translationRole> element to <targetLanguage>. The corresponding textual content is placed in a <SegC> element;
- <alt-trans> maps onto the <MultiC> element and simultaneously sets the value of the <translationStatus> element to alternate.

XLIFF further elements and attributes map onto MLIF elements as follows:

- The XLIFF tool attribute maps onto the <creationTool> element.

C.3 Example of data

The following example, based on XLIFF version 1.2, focuses on the bilingual part of an XLIFF document:

```
<xliff
  xmlns="urn:oasis:names:tc:xliff:document:1.2"
  version="1.2"
  xml:lang="en"
  xsi:schemaLocation="urn:oasis:names:tc:xliff:document:1.2 xliff-core-schema-1.2.xsd"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <file
   source-language="en"
   target-language="fr"
   datatype="winres"
   original="Sample1.rc">
  <header />
  <body>
   <group
     restype="dialog"
     resname="IDD_DIALOG1"
     coord="0;0;186;57"
     font="MS Sans Serif;8">
    <trans-unit
     id="1" restype="caption">
     <source
      xml:lang="en">Title</source>
     <target
      xml:lang="fr">Titre</target>
    </trans-unit>
    <trans-unit
      id="2"
      restype="label"
      resname="IDC_STATIC"
      coord="8;4;19;8">
     <source
      xml:lang="en">Path</source>
     <target
      xml:lang="fr">Chemin</target>
    </trans-unit>
    <trans-unit
      id="3"
      restype="check"
      resname="IDC_CHECK1"
      coord="8;40;41;10">
     <source
      xml:lang="en">Validate</source>
     <target
      xml:lang="fr">Valider</target>
    </trans-unit>
    <trans-unit
      id="4"
      restype="button"
      resname="IDOK"
      coord="129;7;50;14">
     <source
      xml:lang="en">OK</source>
     <target
      xml:lang="fr">OK</target>
    </trans-unit>
    <trans-unit
      id="5"
      restype="button"
      resname="IDCANCEL"
      coord="129;24;50;14">
     <source
      xml:lang="en">Cancel</source>
     <target xml:lang="fr">Annuler</target>
    </trans-unit>
   </group>
  </body>
 </file>
</xliff>
```

The corresponding representation in MLIF default representation is as follows:

```
<MLDC>
 <GI>
  <sourceFormat>XLIFF</sourceFormat>
  <formatVersion>1.2</formatVersion>
 </GI>
 <GroupC>
  <GI>
   <sourceLanguage>en</sourceLanguage>
   <targetLanguage>fr</targetLanguage>
  </GI>
  <groupType>file</groupType>
  <GroupC>
   <groupType>body</groupType>
   <MultiC xml:id="id1" >
    <MonoC xml:lang="en">
     <translationRole>sourceLanguage</translationRole>
     <SegC>Title</SegC>
    </MonoC>
    <MonoC xml:lang="fr">
     <translationRole>targetLanguage</translationRole>
     <SegC>Titre</SegC>
    </MonoC>
   </MultiC>
   <MultiC xml:id="id2">
    <MonoC xml:lang="en">
     <translationRole>sourceLanguage</translationRole>
     <SegC>Path</SegC>
    </MonoC>
    <MonoC xml:lang="fr">
     <translationRole>targetLanguage</translationRole>
     <SegC>Chemin</SegC>
    </MonoC>
   </MultiC>
   <MultiC xml:id="id3">
    <MonoC xml:lang="en">
     <translationRole>sourceLanguage</translationRole>
     <SegC>Validate</SegC>
    </MonoC>
    <MonoC xml:lang="fr">
     <translationRole>targetLanguage</translationRole>
     <SegC>Valider</SegC>
    </MonoC>
   </MultiC>
   <MultiC xml:id="id4">
    <MonoC xml:lang="en">
     <translationRole>sourceLanguage</translationRole>
     <SegC>OK</SegC>
    </MonoC>
    <MonoC xml:lang="fr">
     <translationRole>targetLanguage</translationRole>
     <SegC>OK</SegC>
    </MonoC>
   </MultiC>
   <MultiC xml:id="id5">
    <MonoC xml:lang="en">
     <translationRole>sourceLanguage</translationRole>
     <SegC>Cancel</SegC>
    </MonoC>
    <MonoC xml:lang="fr">
     <translationRole>targetLanguage</translationRole>
     <SegC>Annuler</SegC>
    </MonoC>
   </MultiC>
  </GroupC>
 </GroupC>
</MLDC>
```

Annex D
(informative)

Example: representing smilText data

D.1 Introduction

Within the SMIL 3.0 W3C recommendation (http://www.w3.org/TR/2008/REC-SMIL3-20081201), the smilText modules provide a text container element with an explicit content model for defining timed text (http://www.w3.org/TR/2008/REC-SMIL3-20081201/smil-text.html). smilText has the potential to be an important application context for MLIF as it associates and synchronizes multimedia and textual content.

D.2 Using generic SMIL attributes in MLIF

General timing mechanisms from the SMIL (Synchronized Multimedia Integration Language) recommendation may be used in MLIF-compliant content to provide synchronization mechanisms for textual content. The following SMIL elements are thus integrated in the overall MLIF specification: "begin", "next" and "dur".

D.3 Simplified mapping of monolingual content

The basic use case for articulating MLIF and SMIL involves producing a monolingual SMIL output from a multilingual representation expressed in an MLIF-compliant format. This results from the selection of the content corresponding to a selected language and its integration into one or several <smilText> containers, for instance embedded in a <seq> construct. When applicable, the existing timing information is propagated into the SMIL representation.

In this context, the core mappings between MLIF and the smilText specification are as follows:

— <MonoC> elements map onto <smilText> elements, together with the corresponding attributes (in particular, language);

— <SegC> elements map univocally onto <tev> elements, together with the corresponding descriptors (in particular, temporal ones).

The actual embedding of multilingual content within a single SMIL representation is based on the <switch> constructs within the following skeleton:

```
<switch xmlns:its="http://www.w3.org/2005/11/its">
 <par systemLanguage="en">
  <smilText
    xml:id="TE30"
    region="Contents"
    dur="12s"
    its:dir="ltr"
    xml:lang="en"
    its:translate="yes"> This is a sentence.</smilText>
 </par>
 <par systemLanguage="fr">
  <smilText
    xml:id="TF30"
    region="Contents"
    dur="12s"
    its:dir="ltr"
    xml:lang="fr"
    its:translate="yes">Ceci est une
     phrase.</smilText>
 </par>
</switch>
```

Other non-temporal attributes such as region are not covered by the MLIF specification, and should therefore be created separately from the MLIF-compliant structure.

This mapping can be used conversely to generate MLIF-compliant content from a SMIL representation. The associated use case is typically the preparation of an MLIF-compliant structure that will later contain further translation(s).

D.4 Mapping smilText to MLIF

The core elements of smilText map to MLIF elements as follows[2]:

- the <smilText> element functions as a logical and temporal structuring element that allows the inclusion of inline text content in a SMIL presentation. smilText can also be used as an external, stand-alone timed text format. This is achieved by using the SMIL 3.0 smilText profile;
- the <tev> element defines a "temporal moment" within a block of smilText content; depending on the values of the begin or next attributes, it determines a scheduling time during which the associated text content (up to the following <tev> or <clear> element or the end of the smilText element) is rendered;
- mapping to <SegC>, the <clear> element defines a "temporal moment" within a block of smilText content at which the full contents of the rendering area are cleared.

The following SMIL attributes also map as follows:

- the "dur" attribute maps onto the <duration> element;
- the "begin" attribute maps onto the <begin> element;
- the "next" attribute maps onto the <next> element.

[2] These definitions are taken from the W3C SMIL recommandation: http://www.w3.org/TR/2008/REC-SMIL3-20081201/smil-text.html.

Annex E
(informative)

Example of MLIF usage for subtitles (captioning)

E.1 Introduction

Subtitles are textual versions of the dialog in films, television programs, video games, etc., and are usually displayed at the bottom of the screen. They can be either, a written rendering of the dialog in the same language, or a written rendering of the dialog in a language other than that of the dialog itself. Additional information may be included in order to help viewers who are deaf and hard-of-hearing to follow the dialog [SUB].

Professional subtitlers usually work with specialized computer software and hardware where the video is stored digitally, making each individual frame instantly accessible. In addition to creating the subtitles, the subtitler usually specifies the exact positions where each subtitle shall appear and disappear. For cinema film, this task is traditionally done by separate technicians. The end result is a subtitle file containing the actual subtitles as well as position markers indicating when each subtitle appears and disappears. These markers are usually based on timecode if the work is for the electronic media (e.g. television, videos and DVDs) and on film length (measured in feet and frames) if the subtitles are to be used for traditional cinema film.

E.2 Using MLIF to represent subtitling information

There are several formats that may be used for subtitles. Some of them are de jure standards (e.g. [MPEG-4 TT]), while some others, although not being de jure standards, are currently being used by a large number of people all over the world (e.g. SRT Format – SubRip). SRT is probably the most popular external subtitle file format.

All subtitle formats have to provide a way of synchronizing video frames with subtitles. It goes without saying, that synchronization means associating temporal markers to textual information.

The following example is a very small fragment of a part of an SRT file:

Fragment-1:

```
1
00:00:20,000 --> 00:00:24,400
Subtitle number one…
2
00:00:24,600 --> 00:00:27,800
Subtitle number two…
```

This annex demonstrates how MLIF may be used for subtitles. Fragment-2 and Fragment-3 have been built in compliance with the latest SMIL specification, in particular smilText.

The use of MLIF for dealing with multilingual subtitles is straightforward. It is easy to parse any of the proposed MLIF documents in order to obtain SRT files.

However, depending on the underlying scenario (or workflow), the subtitling information may be represented in two different ways.

The first MLIF way (Fragment-2) defines a single <MultiC> element, and inside this element, two <MonoC> elements are embedded as follows:

Fragment-2:

```
<MLDC>
 <GI/>
 <GroupC>
  <MultiC>
   <MonoC/>
   <MonoC/>
  </MultiC>
 </GroupC>
</MLDC>
```

The second MLIF way (Fragment-3) defines two <MultiC> elements, each containing a single <MonoC> element, with the corresponding outline:

Fragment-3:

```
<MLDC>
 <GI/>
 <GroupC>
  <MultiC>
   <MonoC/>
  </MultiC>
  <MultiC>
   <MonoC/>
  </MultiC>
 </GroupC>
</MLDC>
```

The first approach may be more convenient for a pair-to-pair translation process, while the second way may be more convenient for filtering and selecting of one language (for example, a monolingual block can easily be isolated).

Other implementations may occur depending on how one wants to elicit temporal information associated with the presentation of subtitles. For instance, the following examples use the SMIL attributes in two different ways, with either an <end> (Fragment-4) or a <duration> (Fragment-5).

Fragment-4

```
<MLDC>
 <GroupC>
  <MultiC>
   <MonoC xml:lang="en">
    <begin>00:12:28,928</begin>
    <end>00:12:32,515</end>
    <SegC>- Good morning.</SegC>
    <SegC>- Dr Lecter, my name is Clarice Starling.</SegC>
   </MonoC>
   <MonoC xml:lang="fr">
    <begin>00:12:01,800</begin>
    <end>00:12:05,270</end>
    <SegC>- Bonjour.</SegC>
    <SegC>- Dr Lecter, je m'appelle Clarice Starling.</SegC>
   </MonoC>
  </MultiC>
 </GroupC>
</MLDC>
```

Fragment-5

```
<MLDC>
 <GroupC>
  <MultiC>
   <MonoC xml:lang="en">
    <begin>00:12:28,928</begin>
    <duration>3.607</duration>
    <SegC>- Good morning.</SegC>
    <SegC>- Dr Lecter, my name is Clarice Starling.</SegC>
   </MonoC>
   <MonoC xml:lang="fr">
    <begin>00:12:01,800</begin>
    <duration>3.47</duration>
    <SegC>- Bonjour.</SegC>
    <SegC>- Dr Lecter, je m'appelle Clarice Starling.</SegC>
   </MonoC>
  </MultiC>
 </GroupC>
</MLDC>
```

E.3 Full example

E.3.1 Introduction

The following example associates an SRT representation with a compliant MLIF-based format.

E.3.2 SRT source files

E.3.2.1 English subtitles

The English subtitles are as follows:

```
1
00:00:32,560 --> 00:00:35,119
The world is changed.
2
00:00:35,640 --> 00:00:38,200
I see it in the water.
```

E.3.2.2 French subtitles

The French subtitles are as follows:

```
1
00:00:32,560 --> 00:00:35,119
Le monde a changé.
2
00:00:35,640 --> 00:00:38,200
Je le vois dans l'eau.
```

E.4 MLIF representation - paired sentences

Based on Fragment-2 structure, the resulting MLIF data are as follows:

```
<MLDC>
 <GI>
  <HistoC>
   <date>2008-11-30T17:31:57+01:00</date>
   <author>Samuel CRUZ-LARA</author>
   <version>0.1</version>
  </HistoC>
 </GI>
 <GroupC>
  <MultiC class="subtitles">
   <MonoC xml:lang="en">
    <SegC xml:id="id1en">
     <SegC class="sentence">
      <begin>00:00:32.560</begin>
      <end>00:00:35.119</end>
      <SegC class="word">The</SegC>
      <SegC class="word">world</SegC>
      <SegC class="word">is</SegC>
      <SegC class="word">changed</SegC>
      <SegC class="punctuation">.</SegC>
     </SegC>
    </SegC>
    <SegC xml:id="id2en">
     <SegC class="sentence">
      <begin>00:00:35.640</begin>
      <end>00:00:38.200</end>
      <SegC class="word">I</SegC>
      <SegC class="word">feel</SegC>
      <SegC class="word">it</SegC>
      <SegC class="word">in</SegC>
      <SegC class="word">the</SegC>
      <SegC class="word">water</SegC>
      <SegC class="punctuation">.</SegC>
     </SegC>
    </SegC>
   </MonoC>
   <MonoC xml:lang="fr">
    <SegC xml:id="id1fr">
     <SegC class="sentence">
      <begin>00:00:32.560</begin>
      <end>00:00:35.119</end>
      <SegC class="word">Le</SegC>
      <SegC class="word">monde</SegC>
      <SegC class="word">a</SegC>
      <SegC class="word">changé</SegC>
      <SegC class="punctuation">.</SegC>
     </SegC>
    </SegC>
    <SegC xml:id="id2fr">
     <SegC class="sentence">
      <begin>00:00:35.640</begin>
      <end>00:00:38.200</end>
      <SegC class="word">Je</SegC>
      <SegC class="word">le</SegC>
      <SegC class="word">vois</SegC>
      <SegC class="word">dans</SegC>
      <SegC class="word">l'</SegC>
      <SegC class="word">eau</SegC>
      <SegC class="punctuation">.</SegC>
     </SegC>
    </SegC>
   </MonoC>
  </MultiC>
 </GroupC>
</MLDC>
```

E.5 MLIF representation - captioning scenario

Based on Fragment-3 structure, the resulting MLIF data are as follows:

```
<MLDC>
 <GI>
  <HistoC>
   <date>2008-11-30T17:31:57+01:00</date>
   <author>Samuel CRUZ-LARA</author>
   <version>0.1</version>
  </HistoC>
 </GI>
 <GroupC>
  <MultiC class="subtitles">
   <MonoC xml:lang="en">
    <SegC xml:id="id1en">
     <SegC class="sentence">
      <begin>00:00:32.560</begin>
      <end>00:00:35.119</end>
      <SegC class="word">The</SegC>
      <SegC class="word">world</SegC>
      <SegC class="word">is</SegC>
      <SegC class="word">changed</SegC>
      <SegC class="punctuation">.</SegC>
     </SegC>
    </SegC>
    <SegC xml:id="id2en">
     <SegC class="sentence">
      <begin>00:00:35.640</begin>
      <end>00:00:38.200</end>
      <SegC class="word">I</SegC>
      <SegC class="word">feel</SegC>
      <SegC class="word">it</SegC>
      <SegC class="word">in</SegC>
      <SegC class="word">the</SegC>
      <SegC class="word">water</SegC>
      <SegC class="punctuation">.</SegC>
     </SegC>
    </SegC>
   </MonoC>
  </MultiC>
  <MultiC class="subtitles">
   <MonoC xml:lang="fr">
    <SegC xml:id="id1fr">
     <SegC class="sentence">
      <begin>00:00:32.560</begin>
      <end>00:00:35.119</end>
      <SegC class="word">Le</SegC>
      <SegC class="word">monde</SegC>
      <SegC class="word">a</SegC>
      <SegC class="word">changé</SegC>
      <SegC class="punctuation">.</SegC>
     </SegC>
    </SegC>
    <SegC xml:id="id2fr">
     <SegC class="sentence">
      <begin>00:00:35.640</begin>
      <end>00:00:38.200</end>
      <SegC class="word">Je</SegC>
      <SegC class="word">le</SegC>
      <SegC class="word">vois</SegC>
      <SegC class="word">dans</SegC>
      <SegC class="word">l'</SegC>
      <SegC class="word">eau</SegC>
      <SegC class="punctuation">.</SegC>
     </SegC>
    </SegC>
   </MonoC>
  </MultiC>
 </GroupC>
</MLDC>
```

Annex F
(informative)

Using MLIF for MAF data

MLIF may be used to include pre-existent non-MLIF data like those produced by a Natural Language processor. The aim of the following example is to show how to represent a standoff annotation coming from a morphosyntactic process. The example input is "to eventually decide", in which the verbal form is discontinuous. The challenge is to preserve the ordering position of each element in the flow of word forms. In the following MLIF XML fragment, the MAF XML elements are included without any transformation.

```
<MLDC
   xmlns:maf="http://www.iso.org/ns/MAF">
 <GroupC>
  <MultiC>
   <MonoC>
    <maf:token xml:id="t1">to</maf:token>
    <maf:token xml:id="t2">eventually</maf:token>
    <maf:token xml:id="t3">decide</maf:token>
    <maf:wordForm lemma="to decide" tokens="#t1 #t3"/>
    <maf:wordForm lemma="eventually" tokens="#t2"/>
   </MonoC>
  </MultiC>
 </GroupC>
</MLDC>
```

Annex G
(normative)

Detailed specification

G.1 General

The main body of this International Standard specifies the general principles of the structure and the application of MLIF. This annex specifies in detail the different MLIF classes and the relationships to which they belong.

In this specification, each component of the meta-model is implemented as a specific element bearing the appropriate relations to the other elements implementing the MLIF metamodel.

This specification also describes a series of data categories which should be seen as normative for any MLIF application. It may also be found in the ISO/TC 37 datacategory registry (under www.isocat.org).

G.2 Model classes

G.2.1 model.GIPart

model.GIPart

Groups elements that can be attached at GI level.

Used by <GI>

Members <changeDate> <changeIdentifier> <creationDate> <creationIdentifier> <creationTool> <creationToolVersion> <domain> <formatVersion> <legalStatus> <project> <source> <sourceFormat> <sourceLanguage> <sourceType> <targetLanguage>

G.2.2 model.GroupCPart

model.GroupCPart

Groups elements that can be attached at GroupC level.

Used by <GroupC>

Members <groupType>

G.2.3 model.HistoCPart

model.HistoCPart

Groups elements that can be attached at HistoC level.

Used by <HistoC>

Members <author> <date> <transaction> <version>

G.2.4 model.I18N

model.I18N

Groups together all the information that may be used for internationalization applications.

Used by

Members <translate>

G.2.5 model.L10N

model.L10N

(Localization related elements) Represents information for localization purposes.

Used by

Members <matchQuality> <translationRole> <translationStatus>

Note The main input comes from the XLIFF specification

G.2.6 model.MonoCPart

model.MonoCPart

Groups elements that can be attached at MonoC level.

Used by <MonoC>

Members <segmentation> <translationRole>

G.2.7 model.MultiCPart

model.MultiCPart

Groups elements that can be attached at MultiC level.

Used by <MultiC>

Members <changeDate> <changeIdentifier> <class> <creationDate> <creationIdentifier> <creationTool> <creation ToolVersion> <translationStatus>

G.2.8 model.SegCPart

model.SegCPart

Groups elements that can be attached at SegC level.

Used by <SegC>

Members model.inline
[<beginPairedTag> <endPairedTag> <genericGroupPlaceholder> <genericPlaceholder> <placeholder>]
<segmentation> <translate>

G.2.9 model.inline

model.inline

(Inline elements) Groups information that can appear inside a SegC component. With the exception of <hi> and <sub> element, they all enclose or replace any formatting or control codes that is not text but resides within the SegC component.

Used by model.SegCPart

Members <beginPairedTag> <endPairedTag> <genericGroupPlaceholder> <genericPlaceholder> <placeholder>

Note Source: TMX, also used in XLIFF

G.2.10 model.temporal

model.temporal

Groups together all elements and attributes required for synchronizing temporal information and textual content.

Used by model.MonoCPart model.SegCPart

Members <begin> <duration> <end> <next>

Note Most of the elements described for this class are attributes in the SMIL specification.

G.2.11 model.workflow

model.workflow

(Workflow related elements) Relates to the creation and management of content.

Used by

Members <changeDate> <changeIdentifier> <creationDate> <creationIdentifier> <creationTool> <creationToolVersion>

G.3 Attribute classes

G.3.1 att.classed

att.classed

Specifies a hierarchical description of the component to which it is anchored.

Members <MonoC> <MultiC> <SegC>

Attributes class **Status** Optional

Datatype text

G.3.2 att.id

att.id

Provides a general definition of xml:id for the unique identification of components in the MLIF metamodel.

Members <HistoC> <MonoC> <MultiC> <SegC>

Attributes xml:id **Status** Optional

Datatype text

corresp Points to an equivalent textual segment in another language

Status Optional

Datatype text

Note This attribute is equivalent to the TEI corresp attribute.

G.3.3 att.lang

att.lang

Provides a general definition of xml:lang for the description of working language in the MLIF metamodel and corresponding data categories when applicable.

Members <MonoC>

Attributes xml:lang **Status** Optional

Datatype text

G.3.4 att.linguistic

att.linguistic

Determines linguistic attributes.

Members <SegC>

Attributes pos (part of speech) Indicates the grammatical category of the word being tagged.

Status Optional

Datatype text

lemma Provides an abstract reference to the lexical entry that can be associated to the word being tagged.

Status Optional

Datatype text

tag Specifies the morphological feature(s) attribute that relates to the word form being tagged. The value shall point to the definition of feature structures.

Status Optional

Datatype text

G.3.5 att.xlink

att.xlink

Provides the definition of all XLink attributes needed for MLIF.

Members <MonoC> <SegC>

Attributes

label Labels the resource of locator element.

Status Optional

Datatype text

Note XLink traversal attribute

href Supplies the data to find a remote resource.

Status Optional

Datatype text

Note Specifies a document (URI) and an XPointer

type Indicates the XLink element type.

Status Optional

Sample values include

simple	Creates a simple link.
extended	Creates an extended link.
locator	Creates a locater link that points to a resource.
arc	Creates an arc with multiple resources and various traversal paths.
resource	Creates a resource link, which indicates a specific resource.
title	Creates a title link. Such elements are useful for internationalization purposes.

title Allows a human-readable description.

Status Optional

Datatype text

from Identifies the origin resource of an arc.

Status Optional

Datatype text

Note XLink transversal attribute

to Identifies the destination resource of an arc.

Status Optional

Datatype text

Note XLink traversal attribute

G.4 Elements

G.4.1 <GI>

<GI> (Global Information) Represents technical and administrative information applying to the entire multilingual data collection.

Used by <GroupC> <MLDC>

May contain <HistoC> <changeDate> <changeIdentifier> <creationDate> <creationIdentifier> <creationTool> <creationToolVersion> <domain> <formatVersion> <legalStatus> <project> <source> <sourceFormat> <sourceLanguage> <sourceType> <targetLanguage>

Declaration element GI { (model.GIPart | <HistoC>)* }

Note Example: title of the data collection, revision history, context,…

G.4.2 <GroupC>

<GroupC> (Grouping components) Represents a sub-collection of multilingual data having a common origin or purpose within a given project.

Used by <GroupC> <MLDC>

May contain <GI> <GroupC> <MultiC> <groupType>

Declaration element GroupC { <GI>?, model.GroupCPart*, (<GroupC>* | <MultiC>*) }

Note The model for GroupC does not allow for a mixture of GroupC and SegC.

G.4.3 <HistoC>

<HistoC> (History Component) Traces modifications on the component to which it is anchored (i.e. versioning).

In addition to global attributes att.id (@id, @corresp)

Used by <GI> <MonoC> <MultiC>

May contain <author> <date> <transaction> <version>

Declaration element HistoC { att.id, model.HistoCPart* }

Note Example: author of the modification, date of the modification, version number,…

G.4.4 <MLDC>

<MLDC> (Multilingual Data Collection) Represents a collection of data containing global information and several multilingual units.

Used by

May contain <GI> <GroupC>

Declaration element MLDC { <GI>?, <GroupC>* }

G.4.5 <MonoC>

<MonoC> (Monolingual Component) Groups information related to one language and is part of a multilingual component (MultiC).

In addition to global attributes att.lang (@lang) att.id (@id, @corresp) att.xlink (@label, @href, @type, @title, @from, @to) att.classed (@class)

Used by <MultiC>

May contain <HistoC> <SegC> <segmentation> <translationRole>

Declaration

```
element MonoC
{
  att.lang,
  att.id,
  att.xlink,
  att.classed,
  <HistoC>*,
 ( <SegC> | model.MonoCPart )*
}
```

G.4.6 <MultiC>

<MultiC> (Multilingual Component) Groups together all variants of a given textual content.

In addition to global attributes att.id (@id, @corresp) att.classed (@class)

Used by <GroupC> <MultiC>

May contain <HistoC> <MonoC> <MultiC> <changeDate> <changeIdentifier> <class> <creationDate> < creationIdentifier> <creationTool> <creationToolVersion> <translationStatus>

Declaration

```
element MultiC
{
  att.id,
  att.classed,
  <HistoC>*,
  model.MultiCPart*,
  <MonoC>*,
  <MultiC>*
}
```

G.4.7 <SegC>

<SegC> (Segmentation Component) Allows any level of segmentation for textual information, possibly in a recursive manner.

In addition to global attributes	att.id (@id, @corresp) att.xlink (@label, @href, @type, @title, @from, @to) att.classed (@class) att.linguistic (@pos, @lemma, @tag)
Used by	<MonoC> <SegC>
May contain	<SegC> <beginPairedTag> <endPairedTag> <genericGroupPlaceholder> <genericPlaceholder> <placeholder> <segmentation> <translate>

Declaration	element SegC
	{
	att.id,
	att.xlink,
	att.classed,
	att.linguistic,
	(text | <SegC> | model.SegCPart)*
	}

G.4.8 <author>

<author> Names the person responsible for the creation of the content. This element maps onto the "originator" data category [ISO 12620:2009; TC37 DCR] as available from ISOCat.

Used by	model.HistoCPart
May contain	Character data only
Declaration	element author { text }

G.4.9 <begin>

<begin> Defines an absolute time when the encompassing component becomes active.

Used by	model.temporal
May contain	Character data only
Declaration	element begin { text }

G.4.10 <beginPairedTag>

<beginPairedTag> (Begin paired tag) Delimits the beginning of a paired sequence of inline codes. Each <beginPairedTag> has a corresponding <endPairedTag> element within the segment.

Used by	model.inline
May contain	Character data only
Declaration	element beginPairedTag { text }
Note	Implemented as <bpt> in TMX and XLIFF