Language resource management – Multilingual information framework

Gestion de ressources linguistiques – Modèle pour la représentation d’informations multilingues

Warning

This document is not an ISO International Standard. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an International Standard. Recipients of this document are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

Document type: International standard

Document subtype: Committee draft

Document stage: (30) Committee

Document language: E
Copyright notice

This ISO document is a Committee Draft and is copyright-protected by ISO. Except as permitted under the applicable laws of the user’s country, neither this ISO draft nor any extract from it may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, photocopying, recording or otherwise, without prior written permission being secured. Requests for permission to reproduce should be addressed to ISO at the address below or ISO’s member body in the country of the requester.

Copyright Manager
ISO Central Secretariat
1 rue de Varembé
1211 Geneva 20 Switzerland
tel. + 41 22 749 0111
fax + 41 22 749 0947
internet: iso@iso.org

Reproduction may be subject to royalty payments or a licensing agreement. Violators may be prosecuted.

Foreword

This document has been fully edited in XML using the TEI guidelines. The Word version has been automatically generated using the tools designed by the University of Oxford.

Introduction

2
The scope of activities in localization and translation memory (TM), as well as any type of online multilingual customization (subtitling, iTV, karaoke) is very large, and numerous independent groups are working on these aspects, such as LISA, OASIS, W3C, ISO, etc. Under the guidance of the above-mentioned groups, many formats have been developed. Some of the major formats of specific interest for localization and translation memories are TMX (LISA/OSCAR), XLIFF (OASIS), ITS (W3C). There are many identical requirements for all the formats irrespective of the differences in final output. For example, all the formats aim at being user-friendly, easy-to-learn, and at reusing existing databases or knowledge. All these formats work well in the specific field they are designed for, but they lack a synergy that would make them interoperable when using one type of information in a slightly different context, giving rise to the fear of competition between them.

In this document, we describe in more details the goals of the standardization work, the way to achieve this and a first empirical basis that will serve for proposing a metamodel and data-categories for Multi Lingual Information Framework (MLIF).

The general objective is to reach a better convergence between heterogeneous standardisation activities that are taking place in the domain of data modeling (XML; W3C), text management (TEI; TEI-C), multilingual information (TMX;LISA – XLIFF, OAXAL;OASIS) and multimedia (SMILText;W3C).

1 Scope

MLIF provides a generic platform for modeling and managing multilingual information in various domains: localization, translation, multimedia, document management, digital library, and information or business modeling applications. MLIF provides a metamodel and a set of generic data categories for various application domains. MLIF also provides
strategies for the interoperability and/or linking of models including (but not limited): XLIFF, TMX, SMILText and ITS.

2 Normative references

For this international standard there are seven ISO documents listed as normative references some of which still need to be approved as international standards.

ISO 16642 Computer applications in terminology – Terminological markup framework
ISO DIS 24611 Morphosyntactic annotation framework
ISO CD 24615 Syntactic annotation framework
ISO 639–1:1988 Information technology – Code for the representation of names of languages
ISO 8601:1988 Data elements and interchange formats – Information interchange – Representation of dates and times

3 Terms and definitions

This clause will be completed at the next development stage of the document by accommodating further suggestions from reviewers and member bodies.

... 

... 

... 

4 MLIF Metamodel

The MLIF metamodel (see Figure 1) is made of the following components:
— MLDC (MultiLingual Data Collection) Represents a collection of data containing global information and several multilingual units.
— GI (Global Information) Represents technical and administrative information applying to the entire multilingual data collection.
— GroupC (Grouping component) Represents a sub-collection of multilingual data having a common origin or purpose within a given project.
— MultiC (Multilingual Component) Groups together all variants of a given textual content.
— MonoC (Monolingual Component) Part of a multilingual component (MultiC) containing information related to one language
— HistoC (History Component) This generic component allows to trace modifications on the component it is anchored to (i.e. versioning).
— SegC (Segmentation Component) A recursive component allowing any level of segmentation for textual information.

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

— by fully implementing the MLIF metamodel starting at the level of the <MLDC> component;
— by specifically embedding MLIF compliant information within another model, by implementing one of the lower level MLIF component, namely <GroupC>, <MultiC> or <MonoC>.
5 Core Data Categories for MLIF

5.1 Introduction

All the different models have very similar hierarchical structure but they have different terms and methods of storing metadata relevant to them in particular. MLIF provides a generic structure that can establish basic foundation for all other models. This model provides flexibility to make any element or attribute of this format to be defined explicitly or not. If the value is not defined explicitly it will take default value. Most of the models will also define their own elements and attributes those will fit into this using extensibility that is one of the basic requirements of MLIF model.

The various data categories appearing in Figure 1 are further elicited in the following section as well as in the final specification.
5.2 General principals concerning the use of W3C generic attributes

All MLIF compliant applications must use the following W3C attributes as follows:

- xml:lang must be used in accordance to W3C recommendations to represent the working language of any relevant component. In particular, it should be used systematically for any implementation of the MonoC component.
- xml:id must me used in accordance to W3C recommendations to provide a unique identifier to a component of the MLIF meta model.

5.3 Recommended data categories for the GI component

- domain Specifies the domain on which the MLDC is dependent.
- project Specifies a project within the domain on which the MLDC is dependent.
- source A complete citation of the bibliographic information pertaining to a document or other resource. [ISO 12620:1999]
- sourceType In multilingual and translation-oriented language resource or terminology management, the kind of text used to document the selection of lexical or terminological equivalents, collocations, and the like. [ISO 12620:1999]
- sourceLanguage In a translation-oriented language resource or terminology database, the language that is taken as the language in which the original text is written. [ISO 12620:1999]
— **sourceFormat** Indicates the format from which MLIF data have been generated from an MLIF compliant application

— **targetLanguage** In a translation-oriented language resource or terminology database, the language that is taken as the language into which the original text is translated. [ISO 12620:1999]

— **formatVersion** When applicable, the formatVersion data category should be used in combination to sourceFormat to indicate the corresponding version of the format from which MLIF data have been generated from an MLIF compliant application.

— **creationTool** (Creation tool) Identifies the tool that created the content of the encompassing component.

— **creationToolVersion** (Creation tool version) Identifies the version of the tool that created the monolingual or multilingual content. Its possible values are not specified by the standard but each tool provider should publish the string identifier it uses.

— **creationDate** (Creation date) Specifies the date of creation of the element.

— **creationIdentifier** (Creation identifier) Specifies the identifier of the user who created the encompassing component.

— **changeDate** (Change date) Specifies the date of the last modification of the encompassing component.

— **changeIdentifier** (Change identifier) Specifies the identifier of the user who last modified the encompassing component.

**5.4 Recommended data categories for the GroupC component**

— **groupType** (Group type) Specific the rationale for grouping MultiC information together.

**5.5 Recommended data categories for the MultiC component**

— **class** A hierarchical high level description of the component it is anchored to.

— **changeDate** (Change date) Specifies the date of the last modification of the encompassing component.

— **changeIdentifier** (Change identifier) Specifies the identifier of the user who last modified the encompassing component.

— **creationTool** (Creation tool) Identifies the tool that created the content of the encompassing component.

— **creationToolVersion** (Creation tool version) Identifies the version of the tool that created the monolingual or multilingual content. Its possible values are not specified by the standard but each tool provider should publish the string identifier it uses.

— **creationIdentifier** (Creation identifier) Specifies the identifier of the user who created the encompassing component.
— creationDate (Creation date) Specifies the date of creation of the element.
— translationStatus (Translation status) Indicates whether the MultiC component has a specific status with regards the translation process.

5.6 Recommended data categories for the MonoC component

— translationRole (Translation role) Determines whether the encompassing MonoC component corresponds to a source language or a target language in a translation process
— \texttt{lb/} (Line break) Indicates the beginning of a new line.
— \texttt{att.xlink} Provides the definition of all XLink attributes needed for MLIF

| href | Supplies the data to find a remote resource |
| type | Indicates the XLink element type           |
| label | Label of a resource of locator element   |
| title | Human-readable description               |
| from | What resource an arc comes from           |
| to   | What resource an arc goes to              |

5.7 Recommended data categories for the SegC component

— translationRole (Translation role) Determines whether the encompassing MonoC component corresponds to a source language or a target language in a translation process
— \texttt{beginPairedTag} (Begin paired tag) The <beginPairedTag> data category is used to delimit the beginning of a paired sequence of native codes. Each <beginPairedTag> has a corresponding <endPairedTag> element within the segment.
— \texttt{endPairedTag} (End paired tag) The <endPairedTag> data category is used to delimit the end of a paired sequence of native codes. Each <endPairedTag> has a corresponding <beginPairedTag> element within the segment.
— \texttt{genericGroupPlaceholder} (Generic group placeholder) The <genericGroupPlaceholder> data category is used to replace any inline code of the original document that has a beginning and an end, does not overlap other paired inline codes and can be moved within its parent structural element. [From TMX to be edited] The actual inline data is stored in <tag> elements in the header of the file. The required xid attribute is used to reference the <tag> element that contains the replaced code.
— **placeholder** (Placeholder) The `<placeholder>` data category is used to delimit a sequence of native standalone codes in the segment, or the initial or ending portion of a paired tag that does not have its matching code within the segment, that contains embedded translatable text.

— **genericPlaceholder** (Generic placeholder) The `<genericPlaceholder>` data category is used to replace any inline code of the original document. [From TMX to be edited] The actual inline data is stored in `<tag>` elements in the header of the file. The required xid attribute is used to reference the `<tag>` element that contains the replaced code.

— **translate** The translate data category expresses information about whether the content of encompassing SegC component should be translated or not. The values of this data category are "yes" (translatable) or "no" (not translatable).

— **att.linguistic** Linguistic data categories

<table>
<thead>
<tr>
<th>Pos</th>
<th>(part of speech) indicates the grammatical category of the word being tagged</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lemma</td>
<td>provides an abstract reference to the lexical entry that can be associated to the word being tagged</td>
</tr>
</tbody>
</table>

— **att.xlink** Provides the definition of all XLink attributes needed for MLIF

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>href</td>
<td>Supplies the data to find a remote resource</td>
</tr>
<tr>
<td>type</td>
<td>Indicates the XLink element type</td>
</tr>
<tr>
<td>label</td>
<td>Label of a resource of locator element</td>
</tr>
<tr>
<td>title</td>
<td>Human-readable description</td>
</tr>
<tr>
<td>from</td>
<td>What resource an arc comes from</td>
</tr>
<tr>
<td>to</td>
<td>What resource an arc goes to</td>
</tr>
</tbody>
</table>

### 5.8 Recommended data categories for the HistoC component

The HistoryComponent is a generic component allowing to trace modifications on the component it is anchored to (e.g., creation, modification, validation). It can be anchored onto any component of the metamodel. In MLIF metamodel, the HistoryComponent may be anchored to the GlobalInformation or to the MonoLingualComponent. In the GlobalInformation component, it keeps all information related to any modification on the context or on the domain; in the MonoLingualComponent, it allows keeping all evolutions or any enhancement of the content.

The HistoryComponent is described by four data categories:

— **author** the person responsible for the creation of the content.
— **version** a unique and non-ambiguous number used for versioning purposes.
transaction one of the steps involved in the creation, approval, and use of a specific component (approval, check, exportation, importation, input, modification, origination, standardization, userAccess, withdrawal).

date a date.

5.9 Inline annotation data categories

Multilingual text documents are often only one stage in a complex workflow that involves external document sources baring a wide variety of formats. From these, it is often necessary to keep inline markup that, e.g., indicate presentational features that have to be kept from a source document to a translated target document. To this purpose MLIF compliant applications shall use the following data categories, which, in particular maps onto similar subsets in TMX and XLIFF.

The following data categories should all be used in relation to the <SegC> component:

— beginPairedTag (Begin paired tag) The <beginPairedTag> data category is used to delimit the beginning of a paired sequence of native codes. Each <beginPairedTag> has a corresponding <endPairedTag> element within the segment.
— endPairedTag (End paired tag) The <endPairedTag> data category is used to delimit the end of a paired sequence of native codes. Each <endPairedTag> has a corresponding <beginPairedTag> element within the segment.
— genericGroupPlaceholder (Generic group placeholder) The <genericGroupPlaceholder> data category is used to replace any inline code of the original document that has a beginning and an end, does not overlap other paired inline codes and can be moved within its parent structural element. [From TMX to be edited] The actual inline data is stored in <tag> elements in the header of the file. The required xid attribute is used to reference the <tag> element that contains the replaced code.
— genericPlaceholder (Generic placeholder) The <genericPlaceholder> data category is used to replace any inline code of the original document. [From TMX to be edited] The actual inline data is stored in <tag> elements in the header of the file. The required xid attribute is used to reference the <tag> element that contains the replaced code.
— placeholder (Placeholder) The <placeholder> data category is used to delimit a sequence of native standalone codes in the segment, or the initial or ending portion of a paired tag that does not have its matching code within the segment, that contains embedded translatable text.

[To be removed once Datcats are all set.] <g>, <ph> and <x> elements allow you to encapsulate or replace original native inline codes. The <hi> element allows you to add
extra markup not related to existing inline codes. And the <sub> element, used inside encapsulated inline code, allows you to delimit embedded translatable text.

5.10 Localisation data categories

The following data categories should all be used to provide localisation related information:

- **translationRole** (Translation role) Determines whether the encompassing MonoC component corresponds to a source language or a target language in a translation process.
- **translationStatus** (Translation status) Indicates whether the MultiC component has a specific status with regards the translation process.

5.11 Internationalisation data categories

- **translate** The translate data category expresses information about whether the content of encompassing SegC component should be translated or not. The values of this data category are 'yes' (translatable) or 'no' (not translatable).

5.12 Temporal synchronisation data categories

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

- **duration** The duration of the encompassing component (SegC, MonoC or MultiC) specified as a simple time value.
- **begin** This data category defines an absolute time when the encompassing component becomes active.
- **next** The next data category defines a relative start time, relative to the effective begin time of the parent MonoC component, or the most recently activated SegC component within the parent.

5.13 Augmented computer-aided translation using MLIF

The main objective of adding to MLIF linguistic properties such as **lemma** and **part-of-speech** is to provide computer-aided translation tools with linguistic information in order to improve the performance of the current translation memory systems.

The example below shows an MLIF file that could be used by a computer-aided translation tool based on translation memory, bilingual lexicons and linguistic analysis to translate a sentence similar to the sentence “The tool is described in the following chapters:”. 

The class “word” is used to specify the lemma and the part-of-speech of all the tokens of the sentence ‘The tool is described in the following chapters:’ and its translation ‘L’outil est décrit dans les chapitres suivants:’.

Figure 3 — MLIF file related to a computer-aided translation tool based on translation memory, bilingual lexicons and linguistic analysis.

With the translation memory of this example (Figure 2), the current tools which are based only on word strings such as TRADOS, DÉJÀ VU and SDLX do not propose any translation to the sentence ‘The first chapter treats multimedia.’. However, a computer-aided translation tool using this translation memory, an English-to-French bilingual lexicon and a linguistic analyzer for English and French could propose the sentence ‘Le premier chapitre traite multimédia’ as a translation to the sentence ‘The first chapter treats multimedia.’.
The original examples contained non ISO compliant language codes generated by the tool. These have been corrected in this example.

6 Relation with other standards

As with the “Terminological Markup Framework” TMF (ISO 16642:2003) in terminology, MLIF introduces a metamodel in combination with chosen data categories as a means 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 can be used, we may consider a specific granularity of segmentation and description, built on MAF (ISO DIS 24611), SynAF (ISO CD 24615), TMF respectively, for morphological description, syntactical annotation or terminological description.

Supporting the construction and the interoperability of localization and “Translation Memories” TM resources, MLIF 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, which 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 will not only describes elementary linguistic segments (i.e. sentence, syntactical component, word, part of speech, …), but it may also be used to represent document structure (i.e. title, abstract, paragraph, section, …). In addition, MLIF allows using external and internal links (i.e. annotations and references).

The MLIF is being designed with the objective of providing a common platform for all the existing tools developed by the groups listed in the introduction section. It promotes the use of a common framework for the future development of several different formats: TMX, XLIFF, … MLIF can be considered as a parent for all these formats, since all of them deal with multilingual data expressed in the form of segments or text units. They all can be stored, manipulated and translated in a similar manner.
6.1 TMX

6.1.1 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. In existence since 1998, TMX is a certifiable standard format. TMX is developed and maintained by OSCAR (Open Standards for Container/Content Allowing Re-use), a LISA Special Interest Group.

6.1.2 Mapping of TMX onto MLIF components and data categories

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

- `<tmx>` matches onto the `<MLDC>` component
- `<header>` maps onto the `<GI>` component
- `<body>` is a container for `<tuv>` element and does not map onto any component of the `<MLIF>` metamodel
- `<tu>` maps onto the `<MultiC>` component
- `<tuv>` maps onto the `<MonoC>` component
- `<seg>` maps onto the `<SegC>` component

TMX further elements and attributes map onto MLIF core data categories as follows:

- the `creationtool` attribute maps onto the `<creationTool>` data category
- the `creationdate` attribute maps onto the `<creationDate>` data category
- the `tuid` attribute maps onto the `<creationIdentifier>` data category within the MultiC component
- the `<prop>` element does not match onto any specific data category, as it represents a generic placeholder for application dependant data category. When applicable, a specific `<prop>` element should explicitly be mapped onto MLIF core data categories or standardized ISO/TC 37 data category as available from ISOCat

6.1.3 Example

The following example, based on TMX version 1.4, focuses on the multilingual units of a TMX document. It should be noted that TMX version 2.0 is available as a Working Draft since March 28th, 2007.
Das von ihm entwickelte zehnstufige Qualitätssicherungsverfahren ist über 1300 Jahre alt, jedoch erheblich gründlicher und genauer als die heutigen Verfahren.

Le processus de contrôle de qualité 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.

His 10-stage quality control process initiated more than 1300 years ago is far more thorough and exacting than any existing today.

El proceso de control de calidad en diez pasos que inició hace más de 1300 años es mucho más completo y preciso que los que existen en la actualidad.

Il suo metodo di controllo di qualità in 10 fasi risale a più di 1300 anni fa ed è molto più accurato e preciso di qualsiasi metodo attuale.

그가 1300 여년 전 시작한 10 단계 품질 관리 방법은 현존하는 것보다 훨씬 더 철저하고 정확하다.

多年前，他创立了翻译质量控制的十个步骤，其严谨与精确程度远远超过当今翻译界。

The corresponding representation in MLIF default representation must look as follows:
Das von ihm entwickelte zehnstufige Qualitätssicherungsverfahren ist über 1300 Jahre alt, jedoch erheblich gründlicher und genauer als die heutigen Verfahren.

Le processus de contrôle de qualité 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.

His 10-stage quality control process initiated more than 1300 years ago is far more thorough and exacting than any existing today.

El proceso de control de calidad en diez pasos que inició hace más de 1300 años es mucho más completo y preciso que los que existen en la actualidad.

Il suo metodo di controllo di qualità in 10 fasi risale a più di 1300 anni fa ed è molto più accurato e preciso di qualsiasi metodo attuale.

그가 1300여년 전 시작한 10 단계 품질 관리 방법은 현존하는 것보다 훨씬 더 철저하고 정확하다.

1300多年前，他创立了翻译质量控制的十个步骤，其严谨与精确程度远远超过当今翻译界。
6.2 XLIFF

6.2.1 Introduction

The purpose of the OASIS XLIFF is to define, through extensible XML vocabularies, and promote the adoption of, a specification for the interchange of localizable software and document based objects and related metadata.

6.2.2 Mapping of XLIFF onto MLIF components and data categories

XLIFF differs from the MLIF metamodel in that it makes a clear difference between source and target language for monolingual information. This should be handled through the appropriate use of the <translationRole> data category in <MonoC> components together with the language declarations (<sourceLanguage> and <targetLanguage>) in <G>.

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

- <xliff> matches onto the <MLDC> component
- <++header> maps onto the <GI> component
- <++body> is a container for <tuv> element and does not map onto any component of the MLIF metamodel
- the <phase> element maps onto the <Histoc> component
- <trans-unit> maps onto the <MultiC> component
- <source> maps onto the MonoC component, in combination with setting the value of the translationRole data category to sourceLanguage. The corresponding textual content must be placed in a <SegC> component.
- <target> maps onto the <MonoC> component, in combination with setting the value of the <translationRole> data category to <targetLanguage>. The corresponding textual content must be placed in a <SegC> component.
- <alt-trans> maps onto the <MultiC> component, in combination with setting the value of the <translationStatus> data category to <alternateTranslation>

XLIFF further elements and attributes map onto MLIF core data categories as follows:

- the tool attribute maps onto the <creationTool> data category

6.2.3 Example

The following example, based on XLIFF version 1.0, focuses on the bilingual part of an XLIFF document. The current version of XLIFF is 1.1 (October 31st, 2003).

<xliff xmlns="urn:oasis:names:tc:xliff:document:1.1"
<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">&amp;Path:</source>
        <target xml:lang="fr">&amp;Chemin :</target>
      </trans-unit>
      <trans-unit id="3" restype="check" resname="IDC_CHECK1"
        coord="8;40;41;10">
        <source xml:lang="en">&amp;Validate</source>
        <target xml:lang="fr">&amp;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>
The corresponding representation in MLIF default representation must look as follows (work in progress):

```xml
<MLDC xmlns="http://www.iso.org/ns/MLIF">
  <sourceFormat>XLIFF</sourceFormat>
  <formatVersion>1.1</formatVersion>
  <xliff xmlns="urn:oasis:names:tc:xliff:document:1.1">
    xml:lang="en"
    schemaLocation="urn:oasis:names:tc:xliff:document:1.1 xliff-core-schema-1.1.xsd"/>
  </xliff>
  <GroupC>
    <GI>
      <sourceLanguage>en</sourceLanguage>
      <targetLanguage>fr</targetLanguage>
    </GI>
    <groupType>file</groupType>
    <file datatype="winres" original="Sample1.rc"/>
  </GroupC>
  <GroupC>
    <groupType>body</groupType>
    <group>
      <restype="dialog">
        <resname>"IDD_DIALOG1"</resname>
        <coord>"0;0;186;57"</coord>
        <font>"MS Sans Serif;8"</font>
      </group>
      <MultiC xml:id="1" restype="caption">
        <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="2" restype="label">
        <MonoC xml:lang="en">
          <translationRole>sourceLanguage</translationRole>
          <SegC>&amp;Path:</SegC>
        </MonoC>
        <MonoC xml:lang="fr">
          <translationRole>targetLanguage</translationRole>
          <SegC>&amp;CheminÂ :</SegC>
        </MonoC>
      </MultiC>
      <MultiC xml:id="3" restype="check">
        ...
      </MultiC>
    </group>
  </GroupC>
</MLDC>
```
6.3 Example: TMX and MLIF interaction

Figure 1 illustrates the interaction between TMX and MLIF. This diagram includes the following steps: extraction, translation, merging. The starting point is a TMX document which linguistic content is in English (en) and in German (de). The extraction process (1) allows to obtain in one side a “Skeleton File” (2) which contains all TM formatting information and in another part a MLIF file (3) in which only relevant linguistic information is stored. As most translators (human or automatic) work with TMX software oriented-tools, a XSL style-sheet allows to transform a MLIF document into a TMX
document or into any ML2 document. This file does not contain any formatting information. Once the translator (human or automatic) has added the related Japanese translation, another XSL style-sheet allows to transform a TMX document into a MLIF document (4). Finally, the new MLIF document (this containing the Japanese translation) is merged with the "Skeleton File" in order to obtain a new TMX formatted document (5).

![Figure 5 — TMX and MLIF Interaction.](image)

6.4 smilText

6.4.1 Introduction

smilText is a module defined in the context of the new SMIL 3.0 W3C recommendation. It has the potential to be an important application context for MLIF as it associates and synchronizes multimedia and textual content.

6.4.2 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 synchronisation mechanisms of textual content. The following attributes are thus integrated in the overall MLIF specification:
6.4.3 Simplified mapping of monolingual content

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

In this context, the core mappings between MLIF and the SMIL Text specification would be the following ones:

- <MonoC> components should map onto <smilText> elements, together with the corresponding attributes (in particular, language)
- <SegC> components should 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 should be based on the a <switch> constructs inkeeping with the following skeleton:

```xml
<switch>
  <par systemLanguage="en">
    <smilText
      xml:id="TE30"
      region="Contents"
      dur="12s"
      its:dir="ltr"
      xml:lang="en"
      its:translate="yes"> ...
    </smilText>
  </par>
  <par systemLanguage="fr">
    <smilText
      xml:id="TF30"
      region="Contents"
      dur="12s"
      its:dir="ltr"
      xml:lang="fr"
      its:translate="yes"> ...
    </smilText>
  </par>
</switch>
```

Further ITS data categories will be described in the MLIF document after more elaborate discussion with the W3C colleagues.
Other non temporal attributes such as, e.g. region are not covered by MLIF specifications and should thus be created independently of the information present in the MLIF compliant structure.

This equivalence can be used to conversaly 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).

6.4.4 Mappings of SMIL 3.0 and smilText objects onto MLIF components and data categories

The core elements of smilText map to MLIF components as follows:

— The <smilText> element functions as a logical and temporal structuring element that allows the inclusion of in-line text content into a SMIL presentation. SmilText can also be used as an external, stand-alone timed text format. This is accomplished 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 at which the associated text content (up to the following tev or clear element or the end of the smilText element) is rendered.
— [Mapping on 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. Depending on the values of the begin or next attributes, it also determines a scheduling time at which the associated text content (up to the following tev or clear element or the end of the smilText element) is rendered. This element is functionally equivalent to the tev element, except that it has a side-effect of clearing the rendering area before any new content is rendered.

Besides, the following SMIL attributes maps as follows:

— dur maps onto the <duration> data category
— begin maps onto the <begin> data category
— next maps onto the <next> data category
7 Using MLIF for dealing with video and multilingual subtitles (captioning).

7.1 Introduction

Subtitles are textual versions of the dialog in films and television programs, usually displayed at the bottom of the screen. They can either be a form of written translation of a dialog in a foreign language, or a written rendering of the dialog in the same language, with or without added information to help viewers who are deaf and hard-of-hearing to follow the dialog.

Today professional subtitlers usually work with specialized computer software and hardware where the video is digitally stored on a hard disk, making each individual frame instantly accessible. Besides creating the subtitles, the subtitler usually also tells the computer software the exact positions where each subtitle should 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 where each subtitle should appear and disappear. These markers are usually based on timecode if it is a work for electronic media (e.g. TV, video, DVD), or on film length (measured in feet and frames) if the subtitles are to be used for traditional cinema film.

The finished subtitle file is used to add the subtitles to the picture, either directly into the picture (open subtitles); embedded in the vertical interval and later superimposed on the picture by the end user with the help of an external decoder or a decoder built into the TV (closed subtitles on TV or video); or converted to tiff or bmp graphics that are later superimposed on the picture by the end user (closed subtitles on DVD).

7.2 Using MLIF to represent subtitling information

There are several formats that may be used for subtitles. Some of them are real standards (e.g. ISO/IEC MPEG-4 Timed-Text – MPEG-4 part 17), and some others, that while not being real standards, are currently being used by an important number of people all around 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 to synchronize video frames with subtitles. Obviously, synchronization means associating temporal markers to textual information.

Here is a very simple example of a part of a SRT file:

---

In this annex we want to show how MLIF may be used for subtitles. Examples 3 and 4 have been built by following the latest SMIL (Synchronized Multimedia Integration Language) specification (http://www.w3.org/TR/2008/REC-SMIL3-20081201), in particular smilText (http://www.w3.org/TR/2008/REC-SMIL3-20081201/smil-text.html).

As we may see, using MLIF for dealing with multilingual subtitles is really straightforward. It should be very easy, by parsing any of the proposed MLIF documents, 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 proposal (example 3) defines one single <MultiC> element, and inside this element, two <MonoC> elements. This corresponds to the following outline:

```xml
<MLDC xmlns="http://www.iso.org/ns/MLIF"
>
 <GI/>
 <GroupC>
  <MultiC>
   <MonoC/>
   <MonoC/>
  </MultiC>
 </GroupC>
</MLDC>
```

The second MLIF proposal (example 4) defines two <MultiC> elements, and inside each one of them, one single <MonoC> element, with the corresponding outline:

```xml
<MLDC xmlns="http://www.iso.org/ns/MLIF"
>
 <GI/>
 <GroupC>
  <MultiC>
   <MonoC/>
  </MultiC>
  <MultiC>
   <MonoC/>
  </MultiC>
 </GroupC>
</MLDC>
```
The first MLIF proposal may be more convenient for a pair-to-pair translation process (i.e., xlink should be very useful here), while the second proposal may be more convenient for filtering and selection of one language (e.g., a monolingual bloc can easily be isolated).

Further implementation variations may occur depending on how one want to elicit temporal information associated with the presentation of subtitles. For instance, the following example uses the SMIL attributes in two complementary ways, with either an \texttt{<end>} or a \texttt{<duration>}:

\begin{verbatim}
<MLDC xmlns="http://www.iso.org/ns/MLIF" >
  <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>
<!-- Variant with duration -->
  <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>
</MLDC>
\end{verbatim}

\section{Full example}

\subsection{Introduction}

We present here a complete example associating an SRT representation to a compliant MLIF based format.
7.3.2 SRT source files

7.3.2.1 English subtitles

1
00:00:32,560 --> 00:00:35,119
The world is changed.

2
00:00:35,640 --> 00:00:38,200
I feel it in the water.

3
00:00:39,280 --> 00:00:41,740
I feel it in the earth.

4
00:00:43,600 --> 00:00:46,130
I smell it in the air.

5
00:00:46,840 --> 00:00:48,450
Much that once was...

6
00:00:48,880 --> 00:00:50,179
...is lost.

7
00:00:51,039 --> 00:00:53,600
For none now live who remember it.

8
00:01:06,319 --> 00:01:10,010
It began with the forging
of the Great Rings.

9
00:01:10,719 --> 00:01:12,329
Three were given to the Elves:

10
00:01:12,719 --> 00:01:15,909
Immort al, wisest and fairest
of all beings.

11
00:01:17,359 --> 00:01:19,109
Seven to the Dwarf-Lords:

28
Great miners and craftsmen
of the mountain halls.

And nine...

...nine rings were gifted
to the race of Men...

...who, above all else, desire power.

For within these rings was bound
the strength and will to govern each race.

But they were all of them deceived.

For another ring was made.

in the land of Mordor,
in the fires of Mount Doom...

...the Dark Lord Sauron forged
in secret a Master Ring...

...to control all others.

And into this Ring he poured
his cruelty, his malice...
...and his will to dominate all life.

One Ring to rule them all.

7.3.2.2 French subtitles

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.

3
00:00:39,280 --> 00:00:41,740
Je le ressens dans la terre.

4
00:00:43,600 --> 00:00:46,130
Je le sens dans l'air.

5
00:00:46,840 --> 00:00:48,450
Beaucoup de ce qui existait jadis est perdu.

6
00:00:51,039 --> 00:00:53,600
Car aucun de ceux qui vivent aujourd'hui ne s'en souvient.

8
00:01:06,319 --> 00:01:10,010
Tout commença lorsque les Grands Anneaux furent forgés.

9
00:01:10,719 --> 00:01:12,329
Trois furent donnés aux Elfes :

10
00:01:12,719 --> 00:01:15,909
immortels, les plus sages et les plus respectables de tous les êtres.

Sept aux Seigneurs Nains :

Grands mineurs et sculpteurs de la montagne.

Et neuf, neuf anneaux furent donnés à la race des Hommes qui par dessus tout désirait le pouvoir.

Mais ils furent tous dupés.

Car un autre anneau fût forgé.

Sur les terres du Mordor, dans les flammes de la Montagne du Destin, Sauron le Seigneur des ténèbres forgea en secret le Maître Anneau.
pour gouverner tous les autres.
22
Dans cet Anneau il deversa sa cruauté,
sa malveillance,
et sa volonté de gouverner toute vie.
24
Un Anneau pour les gouverner tous.

7.3.3 MLIF representation – paired sentences

```xml
<MLDC xmlns="http://www.iso.org/ns/MLIF"/>
<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="1en">
        <SegC class="sentence" begin="00:00:32.560" end="00:00:35.119">
          The world is changed.
        </SegC>
      </SegC>
      <SegC xml:id="2en">
        <SegC begin="00:00:35.640" end="00:00:38.200"> I feel it in the water.
        </SegC>
      </SegC>
      <SegC xml:id="3en">
        <SegC begin="00:00:39.280" end="00:00:41.740"> I feel it in the earth.
        </SegC>
      </SegC>
      <SegC xml:id="4en">
        <SegC begin="00:00:43.600" end="00:00:46,130"> I smell it in the air.
        </SegC>
      </SegC>
      <SegC xml:id="5en">
        </SegC>
    </MonoC>
  </MultiC>
</GroupC>
```
Much that once was... is lost. For none now live who remember it.

It began with the forging of the Great Rings.

Three were given to the Elves: Immortal, wisest and fairest of all beings.

Seven to the Dwarf-Lords: Great miners and craftsmen of the mountain halls.

And nine... nine rings were gifted to the race of Men... who, above all else, desire power.

For within these rings was bound the strength and will to govern each race.

But they were all of them deceived.

For another ring was made.

In the land of Mordor, in
the fires of Mount Doom... </SegC>
<SegC xml:id="20en">
  <SegC begin="00:01:50.359" end="00:01:54.120"> ...the Dark Lord Sauron forged in secret a Master Ring... </SegC>
</SegC>
<SegC xml:id="21en">
  <SegC begin="00:01:54.840" end="00:01:56.560"> ...to control all others. </SegC>
</SegC>
<SegC xml:id="22en">
  <SegC begin="00:01:57.000" end="00:02:00.329"> And into this Ring he poured his cruelty, his malice... </SegC>
</SegC>
<SegC xml:id="23en">
  <SegC begin="00:02:01.000" end="00:02:04.549"> ...and his will to dominate all life. </SegC>
</SegC>
<SegC xml:id="24en">
  <SegC begin="00:02:05.959" end="00:02:09.219"> One Ring to rule them all. </SegC>
</SegC>
</MonoC>
<MonoC xml:lang="fr">
  <SegC xml:id="1fr">
    <SegC class="sentence" begin="00:00:32.560" end="00:00:35.119">
      Le monde a changé. </SegC>
  </SegC>
  <SegC xml:id="2fr">
    <SegC begin="00:00:35.640" end="00:00:38.200"> Je le vois dans l’eau. </SegC>
  </SegC>
  <SegC xml:id="3fr">
    <SegC begin="00:00:39.280" end="00:00:41.740"> Je le ressens dans la terre. </SegC>
  </SegC>
  <SegC xml:id="4fr">
    <SegC begin="00:00:43.600" end="00:00:46,130"> Je le sens dans l’air. </SegC>
  </SegC>
  <SegC xml:id="5fr">
    <SegC begin="00:00:46.840" end="00:00:48.450"> Beaucoup de ce qui existait jadis </SegC>
  </SegC>
  <SegC xml:id="6fr">
    <SegC begin="00:00:48.88" end="00:00:50.179"> est perdu. </SegC>
  </SegC>
  <SegC xml:id="7fr">
    <SegC begin="00:00:51.039" end="00:00:53.600"> Car aucun de ceux qui vivent aujourd’hui ne s’en souvient. </SegC>
  </SegC>
</SegC>

7.3.4 MLIF representation – on the fly filtering scenario

The world is changed.

I feel it in the water.

I feel it in the earth.

I smell it in the air.

Much that once was...
...is lost. For none now live who remember it.
It began with the forging of the Great Rings.
Three were given to the Elves:
Immortal, wisest and fairest of all beings.
Seven to the Dwarf-Lords:
Great miners and craftsmen of the mountain halls.
And nine...
...nine rings were gifted to the race of Men...
...who, above all else, desire power.
For within these rings was bound the strength and will to govern each race.
But they were all of them deceived.
For another ring was made.
In the land of Mordor, in the fires of Mount Doom...
the Dark Lord Sauron forged in secret a Master Ring...

to control all others.

And into this Ring he poured his cruelty, his malice...

...and his will to dominate all life.

One Ring to rule them all.

Le monde a changé.

Je le vois dans l'eau.

Je le ressens dans la terre.

Je le sens dans l'air.

Beaucoup de ce qui existait jadis est perdu.

Car aucun de ceux qui vivent aujourd'hui ne s'en souvient.
Tout commença lorsque les Grands Anneaux furent forgés.

Trois furent données eux Elfes :

Immortels, les plus sages et les plus respectables de tous les êtres.

Sept aux Seigneurs Nains :

Grand mineurs et sculpteurs de ma montagne.

Et neuf, neuf anneaux furent donnés à la race des Hommes

qui par dessus tout désirait le pouvoir.

Car à travers ces anneaux furent transmises la force et la volonté de gouverner chaque race.

Mais ils furent tous dupés.

Car un autre anneau fût forgé.

Sur les terres du Mordor, dans les flammes de la Montagne du Destin,

Sauron le Seigneur des ténèbres forgea en secret le Maître Anneau

pour gouverner tous les
autres. </SegC>
</SegC>
<SegC xml:id="22fr">
<SegC begin="00:01:57.000" end="00:02:00.329"> Dans cet Anneau il deversa sa cruauté, sa malveillance, </SegC>
</SegC>
<SegC xml:id="23fr">
<SegC begin="00:02:01.000" end="00:02:04.549"> et sa volonté de gouverner toute vie. </SegC>
</SegC>
<SegC xml:id="24fr">
<SegC begin="00:02:05.959" end="00:02:09.219"> Un Anneau pour les gouverner tous. </SegC>
</SegC>
</MonoC>
</MultiC>
</GroupC>
</MLDC>

8 Specification

8.1 Macros

8.2 Model classes

8.2.1 model.GIPart

<table>
<thead>
<tr>
<th>model.GIPart</th>
<th>Groups data categories that can be attached at GI level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Used by</td>
<td>GI</td>
</tr>
<tr>
<td>Members</td>
<td>changeDate, changeId, creationDate, creationId, creationTool, creationToolVersion, domain, formatVersion, project, source, sourceFormat, sourceLanguage, sourceType, targetLanguage</td>
</tr>
</tbody>
</table>

8.2.2 model.GroupCPart

<table>
<thead>
<tr>
<th>model.GroupCPart</th>
<th>Groups data categories that can be attached at MultiC level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Used by</td>
<td>GroupC</td>
</tr>
<tr>
<td>Members</td>
<td>groupType</td>
</tr>
</tbody>
</table>

8.2.3 model.HistoCPart

<table>
<thead>
<tr>
<th>model.HistoCPart</th>
<th>Groups data categories that can be attached at GI level</th>
</tr>
</thead>
</table>

8.2.4 model.MonoCPart

model.MonoCPart
Groups data categories that can be attached at MonoC level

<table>
<thead>
<tr>
<th>Used by</th>
<th>MonoC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Members</td>
<td>lb translationRole</td>
</tr>
</tbody>
</table>

8.2.5 model.MultiCPart

model.MultiCPart
Groups data categories that can be attached at MultiC level

<table>
<thead>
<tr>
<th>Used by</th>
<th>MultiC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Members</td>
<td>changeDate changeIdentifier creationDate creationIdentifier creationTool creationToolVersion translationStatus</td>
</tr>
</tbody>
</table>

8.2.6 model.SegCPart

model.SegCPart
Groups data categories that can be attached at SegC level

<table>
<thead>
<tr>
<th>Used by</th>
<th>SegC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Members</td>
<td>model.inline [beginPairedTag endPairedTag genericGroupPlaceholder genericPlaceholder placeholder] lb translate</td>
</tr>
</tbody>
</table>

8.2.7 model.i18n

model.i18n
Groups together all data categories that may be used for internationalisation applications.

<table>
<thead>
<tr>
<th>Used by</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Members</td>
<td>Translate</td>
</tr>
</tbody>
</table>
8.2.8 model.inline

<table>
<thead>
<tr>
<th>model.inline</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Inline data categories) This class groups data categories that can appear inside a SegC component. With the exception of <code>&lt;hi&gt;</code> and <code>&lt;sub&gt;</code> element, they all encode or replace any formatting or control codes that is not text but resides within the SegC component.</td>
</tr>
<tr>
<td><strong>Used by</strong></td>
</tr>
<tr>
<td><strong>Members</strong></td>
</tr>
<tr>
<td><strong>Note</strong></td>
</tr>
</tbody>
</table>

8.2.9 model.l18n

<table>
<thead>
<tr>
<th>model.l18n</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Localisation related data categories) data categories used for localisation purposes</td>
</tr>
<tr>
<td><strong>Used by</strong></td>
</tr>
<tr>
<td><strong>Members</strong></td>
</tr>
<tr>
<td><strong>Note</strong></td>
</tr>
</tbody>
</table>

8.2.10 model.temporal

<table>
<thead>
<tr>
<th>model.temporal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Groups together all data categories provided synchronisation or temporal information to textual content.</td>
</tr>
<tr>
<td><strong>Used by</strong></td>
</tr>
<tr>
<td><strong>Members</strong></td>
</tr>
<tr>
<td><strong>Note</strong></td>
</tr>
</tbody>
</table>

8.2.11 model.workflow

<table>
<thead>
<tr>
<th>model.workflow</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Workflow related data categories) Meta data categories related to the creation and management of content</td>
</tr>
<tr>
<td><strong>Used by</strong></td>
</tr>
<tr>
<td><strong>Members</strong></td>
</tr>
</tbody>
</table>
### 8.3 Attribute classes

#### 8.3.1 att.classed

<table>
<thead>
<tr>
<th>att.classed</th>
<th>Member</th>
<th>MonoC MultiC SegC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attributes</td>
<td>class</td>
<td></td>
</tr>
<tr>
<td>Status</td>
<td>Optional</td>
<td></td>
</tr>
<tr>
<td>Datatype</td>
<td>text</td>
<td></td>
</tr>
</tbody>
</table>

#### 8.3.2 att.id

<table>
<thead>
<tr>
<th>att.id</th>
<th>Provides a general definition of xml:id for the unique identification components in the MLIF meta model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Members</td>
<td>HistoC MonoC MultiC SegC</td>
</tr>
<tr>
<td>Attributes</td>
<td>id</td>
</tr>
<tr>
<td>Datatype</td>
<td>text</td>
</tr>
</tbody>
</table>

#### 8.3.3 att.lang

<table>
<thead>
<tr>
<th>att.lang</th>
<th>Provides a general definition of xml:lang for the description of working language in the MLIF meta model and corresponding data categories when applicable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Members</td>
<td>MonoC</td>
</tr>
<tr>
<td>Attributes</td>
<td>lang</td>
</tr>
<tr>
<td>Datatype</td>
<td>text</td>
</tr>
</tbody>
</table>

#### 8.3.4 att.linguistic

<table>
<thead>
<tr>
<th>att.linguistic</th>
<th>Linguistic data categories</th>
</tr>
</thead>
<tbody>
<tr>
<td>Members</td>
<td>SegC</td>
</tr>
<tr>
<td>Attributes</td>
<td>pos</td>
</tr>
<tr>
<td></td>
<td>Status</td>
</tr>
<tr>
<td></td>
<td>Datatype</td>
</tr>
<tr>
<td></td>
<td>lemma</td>
</tr>
<tr>
<td></td>
<td>Status</td>
</tr>
<tr>
<td></td>
<td>Datatype</td>
</tr>
</tbody>
</table>
8.3.5 att.temporal

<table>
<thead>
<tr>
<th>Members</th>
<th>MonoC SegC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attributes</td>
<td></td>
</tr>
<tr>
<td>dur</td>
<td>Status: Optional, Datatype: text</td>
</tr>
<tr>
<td>begin</td>
<td>Status: Optional, Datatype: text</td>
</tr>
<tr>
<td>end</td>
<td>Status: Optional, Datatype: text</td>
</tr>
<tr>
<td>next</td>
<td>Status: Optional, Datatype: text</td>
</tr>
</tbody>
</table>

8.3.6 att.xlink

<table>
<thead>
<tr>
<th>Members</th>
<th>MonoC SegC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attributes</td>
<td></td>
</tr>
<tr>
<td>label</td>
<td>Label of a resource of locator element, Status: Optional, Datatype: text</td>
</tr>
<tr>
<td>href</td>
<td>Supplies the data to find a remote resource, Status: Optional, Datatype: text, Note: XLink traversal attribute</td>
</tr>
<tr>
<td>type</td>
<td>Indicates the XLink element type, Values are: simple Creates a simple link, extended Creates an extended link, locator Creates a locator 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</td>
</tr>
</tbody>
</table>
8.4 Elements

8.4.1 <GI>

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

<table>
<thead>
<tr>
<th>Used by</th>
<th>GroupC, MLDC</th>
</tr>
</thead>
<tbody>
<tr>
<td>May contain</td>
<td>HistoC, changeDate, changeIdentifier, creationDate, creationIdentifier, creationTool, creationToolVersion, domain, formatVersion, project, source, sourceFormat, sourceLanguage, sourceType, targetLanguage</td>
</tr>
<tr>
<td>Declaration</td>
<td>element GI { ( model.GIPart</td>
</tr>
<tr>
<td>Note</td>
<td>Example: title of the data collection, revision history, context...</td>
</tr>
</tbody>
</table>

8.4.2 <GroupC>

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

<table>
<thead>
<tr>
<th>Used by</th>
<th>GroupC, MLDC</th>
</tr>
</thead>
<tbody>
<tr>
<td>May contain</td>
<td>GI, GroupC, MultiC, groupType</td>
</tr>
<tr>
<td>Declaration</td>
<td>element GroupC { GI?, model.GroupCPart*, ( GroupC*</td>
</tr>
<tr>
<td>Note</td>
<td>The model for GroupC does not allow for a mixture of GroupC and SegC.</td>
</tr>
</tbody>
</table>
### 8.4.3 `<HistoC>`

(*History Component*) This generic component allows to trace modifications on the component it is anchored to (i.e. versioning).

<table>
<thead>
<tr>
<th>Attributes</th>
<th>att.id@id</th>
</tr>
</thead>
<tbody>
<tr>
<td>Used by</td>
<td>GI MonoC MultiC</td>
</tr>
<tr>
<td>May contain</td>
<td>author date transaction version</td>
</tr>
<tr>
<td>Declaration</td>
<td><code>element HistoC { model.HistoCPart* }</code></td>
</tr>
<tr>
<td>Note</td>
<td>Example: title of the data collection, revision history, context,…</td>
</tr>
</tbody>
</table>

### 8.4.4 `<MLDC>`

(*MultiLingual Data Collection*) Represents a collection of data containing global information and several multilingual units.

<table>
<thead>
<tr>
<th>Used by</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>May contain</td>
<td>GI GroupC</td>
</tr>
<tr>
<td>Declaration</td>
<td><code>element MLDC { GI?, GroupC* }</code></td>
</tr>
</tbody>
</table>

### 8.4.5 `<MonoC>`

(*Monolingual Component*) Part of a multilingual component (`MultiC`) containing information related to one language

<table>
<thead>
<tr>
<th>Attributes</th>
<th>att.lang@lang att.id@id att.xink@href@type@title@from@to att.classed@class att.temporal@dur@begin@end@next</th>
</tr>
</thead>
<tbody>
<tr>
<td>Used by</td>
<td>MultiC</td>
</tr>
<tr>
<td>May contain</td>
<td>HistoC SegC b translationRole</td>
</tr>
<tr>
<td>Declaration</td>
<td>`element MonoC { HistoC*, ( SegC</td>
</tr>
</tbody>
</table>

### 8.4.6 `<MultiC>`

(*Multilingual Component*) Groups together all variants of a given textual content.

<table>
<thead>
<tr>
<th>Attributes</th>
<th>att.d@id att.classed@class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Used by</td>
<td>GroupC MultiC</td>
</tr>
<tr>
<td>May contain</td>
<td>HistoC MonoC MultiC changeDate changeIdentifier class creationDate creationIdentifier creationTool creationToolVersion translationStatus</td>
</tr>
</tbody>
</table>
8.4.7 `<SegC>`

`<SegC>` (Segmentation Component) A recursive component allowing any level of segmentation for textual information.

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Used by</th>
<th>May contain</th>
<th>Declaration</th>
</tr>
</thead>
<tbody>
<tr>
<td>att.id@id</td>
<td>MonoC SegC</td>
<td><code>SegC</code> beginPairedTag endPairedTag genericGroupPlaceholder genericPlaceholder lb placeholder translate</td>
<td>element <code>SegC</code> { ( text</td>
</tr>
</tbody>
</table>

8.4.8 `<author>`

`<author>` the person responsible for the creation of the content.

<table>
<thead>
<tr>
<th>Used by</th>
<th>May contain</th>
<th>Declaration</th>
</tr>
</thead>
<tbody>
<tr>
<td>model.HistoCPart</td>
<td>Character data only</td>
<td>element <code>author</code> { text }</td>
</tr>
</tbody>
</table>

8.4.9 `<begin>`

`<begin>` This data category defines an absolute time when the encompassing component becomes active.

<table>
<thead>
<tr>
<th>Used by</th>
<th>May contain</th>
<th>Declaration</th>
</tr>
</thead>
<tbody>
<tr>
<td>model.temporal</td>
<td>Character data only</td>
<td>element <code>begin</code> { text }</td>
</tr>
</tbody>
</table>

8.4.10 `<beginPairedTag>`

`<beginPairedTag>` (Begin paired tag) The `<beginPairedTag>` data category is used to delimit the beginning of a paired sequence of native codes. Each `<beginPairedTag>` has a corresponding `<endPairedTag>` element within the segment.

<table>
<thead>
<tr>
<th>Used by</th>
<th>May contain</th>
<th>Declaration</th>
</tr>
</thead>
<tbody>
<tr>
<td>model.inline</td>
<td>Character data only</td>
<td>element <code>beginPairedTag</code> { text }</td>
</tr>
</tbody>
</table>
Note | Implemented as `<bpt>` in TMX and XLIFF

### 8.4.11 `<changeDate>`

| `<changeDate>` | (Change date) Specifies the date of the last modification of the encompassing component. |
| Used by | model.GIPart model.MultiCPart model.workflow |
| May contain | Character data only |
| Declaration | element changeDate { text } |
| Note | cf. creationDate |

### 8.4.12 `<changeIdentifier>`

| `<changeIdentifier>` | (Change identifier) Specifies the identifier of the user who last modified the encompassing component. |
| Used by | model.GIPart model.MultiCPart model.workflow |
| May contain | Character data only |
| Declaration | element changeIdentifier { text } |

### 8.4.13 `<class>`

| `<class>` | A hierarchical high level description of the component it is anchored to. |
| Used by | model.MultiCPart |
| May contain | Character data only |
| Declaration | element class { text } |

### 8.4.14 `<creationDate>`

| `<creationDate>` | (Creation date) Specifies the date of creation of the element. |
| Used by | model.GIPart model.MultiCPart model.workflow |
| May contain | Character data only |
| Declaration | element creationDate { text } |
Note | Date in [ISO 8601] Format. The recommended pattern to use is: YYYYMMDDThhmmssZ Where: YYYY 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.

8.4.15 <creationIdentifier>

<creationIdentifier> (Creation identifier) Specifies the identifier of the user who created the encompassing component.

<table>
<thead>
<tr>
<th>Used by</th>
<th>model.GIPart model.MultiCPart model.workflow</th>
</tr>
</thead>
<tbody>
<tr>
<td>May contain</td>
<td>Character data only</td>
</tr>
<tr>
<td>Declaration</td>
<td>element creationIdentifier { text }</td>
</tr>
</tbody>
</table>

8.4.16 <creationTool>

<creationTool> (Creation tool) Identifies the tool that created the content of the encompassing component.

<table>
<thead>
<tr>
<th>Used by</th>
<th>model.GIPart model.MultiCPart model.workflow</th>
</tr>
</thead>
<tbody>
<tr>
<td>May contain</td>
<td>Character data only</td>
</tr>
<tr>
<td>Declaration</td>
<td>element creationTool { text }</td>
</tr>
</tbody>
</table>

8.4.17 <creationToolVersion>

<creationToolVersion> (Creation tool version) Identifies the version of the tool that created the monolingual or multilingual content. Its possible values are not specified by the standard but each tool provider should publish the string identifier it uses.

<table>
<thead>
<tr>
<th>Used by</th>
<th>model.GIPart model.MultiCPart model.workflow</th>
</tr>
</thead>
<tbody>
<tr>
<td>May contain</td>
<td>Character data only</td>
</tr>
<tr>
<td>Declaration</td>
<td>element creationToolVersion { text }</td>
</tr>
</tbody>
</table>

8.4.18 <date>

$date$ a date.

<table>
<thead>
<tr>
<th>Used by</th>
<th>model.HistoCPart</th>
</tr>
</thead>
<tbody>
<tr>
<td>May contain</td>
<td>Character data only</td>
</tr>
<tr>
<td>Declaration</td>
<td>element date { text }</td>
</tr>
<tr>
<td>Note</td>
<td>The date is encoded according to a profile of [ISO8601] as described in [W3CDTF] and follows the YYYYMMDDThhmmssZ format.</td>
</tr>
</tbody>
</table>
8.4.19 <domain>

Specifies the domain on which the MLDC is dependent.

- **Used by:** model.GIPart
- **May contain:** Character data only
- **Declaration:** element domain { text }

8.4.20 <duration>

The duration of the encompassing component (SegC, MonoC or MultiC) specified as a simple time value.

- **Used by:** model.temporal
- **May contain:** Character data only
- **Declaration:** element duration { text }

8.4.21 <end>

This data category defines an absolute time, when the encompassing component stops being active.

- **Used by:** model.temporal
- **May contain:** Character data only
- **Declaration:** element end { text }
- **Note:** Does this category exist in SMIL?

8.4.22 <endPairedTag>

(End paired tag) The <endPairedTag> data category is used to delimit the end of a paired sequence of native codes. Each <endPairedTag> has a corresponding <beginPairedTag> element within the segment.

- **Used by:** model:inline
- **May contain:** Character data only
- **Declaration:** element endPairedTag { text }
- **Note:** Implemented as <ept> in TMX and XLIFF

8.4.23 <formatVersion>

When applicable, the formatVersion data category should be used in combination to sourceFormat to indicate the corresponding version of the format from which MLIF data have been generated from an MLIF compliant application.

- **Used by:** model.GIPart
- **May contain:** Character data only
- **Declaration:** element formatVersion { text }
Example
The following example illustrate how TMX originayi ng data may be marked within the GI component

```xml
<GI>
<sourceFormat>TMX</sourceFormat>
<formatVersion>1.4</formatVersion>
</GI>
```

### 8.4.24 <genericGroupPlaceholder>**(**Generic group placeholder)**

The `<genericGroupPlaceholder>` data category is used to replace any inline code of the original document that has a beginning and an end, does not overlap other paired inline codes and can be moved within its parent structural element. [From TMX to be edited] The actual inline data is stored in `<tag>` elements in the header of the file. The required `xid` attribute is used to reference the `<tag>` element that contains the replaced code.

<table>
<thead>
<tr>
<th>Used by</th>
<th>model:inline</th>
</tr>
</thead>
<tbody>
<tr>
<td>May contain</td>
<td>Character data only</td>
</tr>
<tr>
<td>Declaration</td>
<td>element genericGroupPlaceholder { text }</td>
</tr>
<tr>
<td>Note</td>
<td>Implemented as <code>&lt;g&gt;</code> in TMX and XLIFF</td>
</tr>
</tbody>
</table>

### 8.4.25 <genericPlaceholder>**(**Generic placeholder)**

The `<genericPlaceholder>` data category is used to replace any inline code of the original document. [From TMX to be edited] The actual inline data is stored in `<tag>` elements in the header of the file. The required `xid` attribute is used to reference the `<tag>` element that contains the replaced code.

<table>
<thead>
<tr>
<th>Used by</th>
<th>model:inline</th>
</tr>
</thead>
<tbody>
<tr>
<td>May contain</td>
<td>Character data only</td>
</tr>
<tr>
<td>Declaration</td>
<td>element genericPlaceholder { text }</td>
</tr>
<tr>
<td>Note</td>
<td>Implemented as <code>&lt;x&gt;</code> in TMX and XLIFF</td>
</tr>
</tbody>
</table>

### 8.4.26 <groupType>**(**Group type)**

Specific the rationale for grouping MultiC information together.

<table>
<thead>
<tr>
<th>Used by</th>
<th>model:GroupCPart</th>
</tr>
</thead>
<tbody>
<tr>
<td>May contain</td>
<td>Character data only</td>
</tr>
<tr>
<td>Declaration</td>
<td>element groupType { text }</td>
</tr>
<tr>
<td>Note</td>
<td>A further version of this data category should probably define recommended values for this data category.</td>
</tr>
</tbody>
</table>

### 8.4.27 <lb/>**(**Line break)**

Indicates the beginning of a new line.

<table>
<thead>
<tr>
<th>Used by</th>
<th>model:SegCPart model:MonoCPart</th>
</tr>
</thead>
<tbody>
<tr>
<td>May contain</td>
<td>Empty element</td>
</tr>
<tr>
<td>Declaration</td>
<td>element lb { empty }</td>
</tr>
</tbody>
</table>
### 8.4.28 <matchQuality>

**<matchQuality>** (Match quality) match quality of the alternateTranslation element is tool specific and can be a score expressed in percentage or an arbitrary value (e.g. `match-quality="high"`).

<table>
<thead>
<tr>
<th>Used by</th>
<th>model.l18n</th>
</tr>
</thead>
<tbody>
<tr>
<td>May contain</td>
<td>Character data only</td>
</tr>
<tr>
<td>Declaration</td>
<td>element matchQuality { text }</td>
</tr>
</tbody>
</table>

### 8.4.29 <next>

**<next>** The next data category defines a relative start time, relative to the effective begin time of the parent MonoC component, or the most recently activated SegC component within the parent.

<table>
<thead>
<tr>
<th>Used by</th>
<th>model.temporal</th>
</tr>
</thead>
<tbody>
<tr>
<td>May contain</td>
<td>Character data only</td>
</tr>
<tr>
<td>Declaration</td>
<td>element next { text }</td>
</tr>
</tbody>
</table>

### 8.4.30 <placeholder>

**<placeholder>** (Placeholder) The <placeholder> data category is used to delimit a sequence of native standalone codes in the segment, or the initial or ending portion of a paired tag that does not have its matching code within the segment, that contains embedded translatable text.

<table>
<thead>
<tr>
<th>Used by</th>
<th>model.inline</th>
</tr>
</thead>
<tbody>
<tr>
<td>May contain</td>
<td>Character data only</td>
</tr>
<tr>
<td>Declaration</td>
<td>element placeholder { text }</td>
</tr>
</tbody>
</table>

| Note       | Implemented as <ph> in TMX and XLIFF |

### 8.4.31 <project>

**<project>** Specifies a project within the domain on which the MLDC is dependent.

<table>
<thead>
<tr>
<th>Used by</th>
<th>model.GIPart</th>
</tr>
</thead>
<tbody>
<tr>
<td>May contain</td>
<td>Character data only</td>
</tr>
<tr>
<td>Declaration</td>
<td>element project { text }</td>
</tr>
</tbody>
</table>

### 8.4.32 <source>

**<source>** A complete citation of the bibliographic information pertaining to a document or other resource. [ISO 12620:1999]
### 8.4.33 `<sourceFormat>`

`<sourceFormat>` Indicates the format from which MLIF data have been generated from an MLIF compliant application.

<table>
<thead>
<tr>
<th>Used by</th>
<th>model.GIPart</th>
</tr>
</thead>
<tbody>
<tr>
<td>May contain</td>
<td>Character data only</td>
</tr>
<tr>
<td>Declaration</td>
<td>element sourceFormat { text }</td>
</tr>
<tr>
<td>Example</td>
<td>The following example illustrate how TMX originating data may be marked within the GI component</td>
</tr>
<tr>
<td></td>
<td><code>&lt;GI&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;sourceFormat&gt;</code>TMX<code>&lt;sourceFormat&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;formatVersion&gt;</code>1.4<code>&lt;formatVersion&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;/GI&gt;</code></td>
</tr>
</tbody>
</table>

### 8.4.34 `<sourceLanguage>`

`<sourceLanguage>` In a translation-oriented language resource or terminology database, the language that is taken as the language in which the original text is written. [ISO 12620:1999]

<table>
<thead>
<tr>
<th>Used by</th>
<th>model.GIPart</th>
</tr>
</thead>
<tbody>
<tr>
<td>May contain</td>
<td>Character data only</td>
</tr>
<tr>
<td>Declaration</td>
<td>element sourceLanguage { text }</td>
</tr>
</tbody>
</table>

### 8.4.35 `<sourceType>`

`<sourceType>` In multilingual and translation-oriented language resource or terminology management, the kind of text used to document the selection of lexical or terminological equivalents, collocations, and the like. [ISO 12620:1999]

<table>
<thead>
<tr>
<th>Used by</th>
<th>model.GIPart</th>
</tr>
</thead>
<tbody>
<tr>
<td>May contain</td>
<td>Character data only</td>
</tr>
<tr>
<td>Declaration</td>
<td>element sourceType { text }</td>
</tr>
</tbody>
</table>
### Note

Both parallel and background texts serve as sources for information used in documenting multilingual terminology entries. [ISO 12620]

---

#### 8.4.36 `<targetLanguage>`

`<targetLanguage>` in a translation-oriented language resource or terminology database, the language that is taken as the language into which the original text is translated. [ISO 12620:1999]

<table>
<thead>
<tr>
<th>Used by</th>
<th>model.GIPart</th>
</tr>
</thead>
<tbody>
<tr>
<td>May contain</td>
<td>Character data only</td>
</tr>
<tr>
<td>Declaration</td>
<td>element <code>targetLanguage</code> { text }</td>
</tr>
</tbody>
</table>

#### 8.4.37 `<transaction>`

`<transaction>` one of the steps involved in the creation, approval, and use of a specific component (approval, check, exportation, importation, input, modification, origination, standardization, userAccess, withdrawal).

<table>
<thead>
<tr>
<th>Used by</th>
<th>model.HistoCPart</th>
</tr>
</thead>
<tbody>
<tr>
<td>May contain</td>
<td>Character data only</td>
</tr>
<tr>
<td>Declaration</td>
<td>element <code>transaction</code> { text }</td>
</tr>
</tbody>
</table>

#### 8.4.38 `<translate>`

`<translate>` The translate data category expresses information about whether the content of encompassing SegC component should be translated or not. The values of this data category are "yes" (translatable) or "no" (not translatable).

<table>
<thead>
<tr>
<th>Used by</th>
<th>model.I18n model.SegCPart</th>
</tr>
</thead>
<tbody>
<tr>
<td>May contain</td>
<td>Empty element</td>
</tr>
<tr>
<td>Declaration</td>
<td>element <code>translate</code> { text }</td>
</tr>
</tbody>
</table>

#### 8.4.39 `<translationRole>`

`<translationRole>` (Translation role) Determines whether the encompassing MonoC component corresponds to a source language or a target language in a translation process.

<table>
<thead>
<tr>
<th>Used by</th>
<th>model.I18n model.MonoCPart</th>
</tr>
</thead>
<tbody>
<tr>
<td>May contain</td>
<td>Empty element</td>
</tr>
<tr>
<td>Declaration</td>
<td>element <code>translationRole</code> { text }</td>
</tr>
</tbody>
</table>

#### 8.4.40 `<translationStatus>`

`<translationStatus>` (Translation status) Indicates whether the MultiC component has a specific status with regards the translation process.
8.4.41 <version>

<table>
<thead>
<tr>
<th>Used by</th>
<th>model.18n model:MultiCPart</th>
</tr>
</thead>
<tbody>
<tr>
<td>May contain</td>
<td>Empty element</td>
</tr>
<tr>
<td>Declaration</td>
<td>element translationStatus { text }</td>
</tr>
</tbody>
</table>

<version> a unique and non-ambiguous number used for versioning purposes.

9 Issues

Line breaking conventions (between segments or within segments by means of a specific inline element <lineBreak>)

Data categories to be defined

— matchQuality (cf. match-quality in XLIFF)

XLIFF

— What to do with the notion of <file> => match to <GroupC>
— inline elements: <g>, <x>, <bpt>, <ept>, <it>