Proceedings of the 13th Conference on Language Resources and Evaluation (LREC 2022), pages 2865–2873
Marseille, 20-25 June 2022
© European Language Resources Association (ELRA), licensed under CC-BY-NC-4.0

An Architecture of resolving a multiple link path in a standoff-style data format to enhance the mobility of language resources

Kazushi Ohya
Tsurumi University
Tsurumi 2-1-3, Tsurumi-ku
Yokohama, Japan
oya-k@tsurumi-u.ac.jp

Abstract

The present data formats proposed by authentic organizations are based on a so-called standoff-style data format in XML, which represents a semantic data model through an instance structure and a link structure. However, this type of data formats intended to enhance the power of representation of an XML format injures the mobility of data because an abstract data structure denoted by multiple link paths is hard to be converted into other data structures. This difficulty causes a problem in the reuse of data to convert into other data formats especially in a personal data management environment. In this paper, in order to compensate for the drawback, we propose a new concept of transforming a link structure to an instance structure on a new marked-up scheme. This approach to language data brings a new architecture of language data management to realize a personal data management environment in daily and long-life use.

1. Background

Socially and diachronically sharing language resources is the ultimate goal in an activity of language documentation, To do so, we have engaged in sharing applications or using shared data formats. As the former way, ToolBox(Summer Institute of Linguistics, 2021b), FLEx(Summer Institute of Linguistics, 2021a), ELAN(Max Planck Institution, 2021), Audacity(Audacity, 2021), Praat(Boersman and Weenink, 2011), and other kinds of applications such as text editors and tabular software have been shared by language researchers. As the latter way, many authentic standards have been proposed(Ide and Pustejovsky, 2017), e.g. TEI (Text Encoding Initiative, 1994), CES (Ide, 1998), Annotation Graph (Bird and Liberman, 1999), ISO Feature Structure(FS) (ISO, 2006) (ISO, 2011), ISO LAF(ISO, 2012) and so on. However, in terms of computer science, sharing applications is not a good way for sharing data in a long span of time. Thus, data format sharing is the best way for data preservation. The data formats proposed from the authorities adopt a so-called standoff-style data format, that is, contain multiple link paths to represent a semantic model over an instance structure in the XML format. For example, TEI, CES, ISO FS and LAF, all the formats use a link structure to represent an alternative semantic structure that cannot be represented by a scheme of an XML data, or an instance structure. This kind of strategy aiming at a syntax with the intensification of expressiveness is good for global-scale data repositories because it ensures data conversion from a variety of applications or formats used in a personal environment into the format used in the archive system. However, from the viewpoint of an individual language researcher, this expressiveness causes harm to the data conversion from the data in archives to one for a personal data management system. Usually a link structure consisting of multiple link paths is hard to convert into another data structure because of lack of applications handling multiple link paths and fundamental difficulties to handle an abstract link-ladder in the instance(s) (Ohya, 2008)(Ohya, 2009)(Ohya, 2015a). A standoff-style format can absorb any scheme into their own format but not vice versa. A multiple-link path has a unilateral convenience of data conversion. Thus this format is not suitable for data management systems used in personal research activities.

As a preparation of seeking the solution to this problem, in order to define requirements for data structures or data management systems in the personal data management system, we had surveyed them(Ohya, 2016), and got the following list of requirements; (1) an arbitrary hierarchy of data structure, (2) a long lifespan of data as much as the researchers themselves, and (3) an intuitive data structure similar to an interlinear data format.

Depending on languages and researchers' preferences, the number of abstraction level varies, and so does the hierarchy of data structures. Thus, (a1) the data format must be flexible in the hierarchy, or scheme-less if it is possible. Needless to say, personal research activities are foundations of every organized activity, and according to the present sense of value, the Procrustean bed is nonsense. Individual language researcher shall not be obliged to fit their data into the one for archives or repositories. Language researchers do not have to use the data formats used in archives or repositories for their personal research activities[1], and (a2) they have a choice to use a data format matching their needs in their studies all their life. As such a data format, language researchers have used a co-called interlinear data format, which is, for ex-

[1]It does not deny the existence of archives and repositories as the commons for language communities, nor the formats used in them. The data conversion from personal data to archival data is another theme and out of the target of this paper(Ohya, 2011).

ample, adopted in ToolBox. (a3) It should be respected for attaching importance to the interlinear data format.
To sum up, we are required to solve a problem of multi-link paths in a proposed data formats in order to response to the requirements, i.e. (a1) flexible hierarchy, (a2) life-log personalized data formats, and (a3) an interlinear style.

2. Outcomes/Proposals

As a way of a solution to the difficulties in the multi-link paths, the following new ideas are introduced.

1. converting a link structure to an instance structure
2. root-element-less scheme in XML data
3. scheme-less data model in a data structure
4. the new pivot data format for language resources proposed as the name MDM; Meta/Marked-up Data Model

As a way to resolve the multi-link path problems, we introduce a new strategy to make a new instance from the link-ladder in a link structure. In a standoff-style data, the element with an ID attribute that is referred to from an element with a href or IDREF attribute has its own IDREF attribute to refer to the other element, and this kind of a link-to-link structure or a link-ladder is hard to predict the number of paths in this ladder in advance to trace. Thus, we convert the link-ladder into an instance structure, and call the conversion the L2I conversion in this paper. The L2I conversion helps us reach the terminal data on a link-ladder easily though the instance structure and get a new base data structure that can be converted into the other data format by using existing data conversion methods such DSSSL, XSLT, XQuery and DOM API, which means the resultant instance can be easily converted into the data for a personal data management system and public archives or repositories.
The resultant instance of the L2I conversion is added into the original XML data, which means the total instance violates the scheme of the original instance. To avoid this inconvenience, we introduce a new policy of a root-element-less scheme of an XML data. This allows the resultant instance that incorporates the original and the new instance made from a link-ladder.
As a natural consequence of the above proposals, we need another policy of scheme-less data model in XML data. Since a link structure is usually not defined in a metadata format, the scheme of the resultant of the L2I conversion cannot be determined before the L2I conversion, nor follows to the pre-defined scheme. This principle of scheme-less data model is, as a matter of fact, the principal policy for annotating in humanities. The data unit annotated by humanities researchers does not exist a priori. The annotation itself is a result of an intellectual exploration, thus impossible to be defined before it. Therefore, a scheme-less data model is ideal for personal activities of language documentation.

In order to implement the L2I conversion, we invent a pivot data format named MDM (Meta/Marked-up Data Model). The MDM is expected to make the L2I conversion easy. Syntactically the MDM is a set of list for representing a graph data. On the MDM, a link structure can be represented as a link edge, which is the target of the L2I conversion, thus easily changed into a node edge of a new instance. The details are in the section 3.
To cover the requirement (a3), we need more tools and ideas rather than these proposals. The details are in the section 5.1.

3. MDM

3.1. Requirements

The MDM should have an ability to be a general format for an XML data especially with a link structure on its own instance. The MDM should be a simple data format, and be equipped with a mechanism to change the link structure into an instance structure with ease.

3.2. Recall: Data Structure

Any data format consits of data and the data structure that is a relation of data. A primitive relation is a pair which consists of two data and the order. This primitive structure can represent any data structure. Practically, a more abstract relation or data structure is adopted in handling a data format. As such a relation, there are four represetntatives: List, Table, Tree and Graph in a level of programming or data management including language documentation. The definitions are as follows.

Pair $(x, y) : x, y \in Data$

List $(x_1, x_2, ..., x_n) : x_n \in Data$

Table $(l_1, l_2, ..., l_n) : l_n \in List$

Graph $\{Node, Edge\}$
$Node := \{x : x \in Data\}$
$Edge := \{(x, y) : x, y \in Node\}$

Tree $\{Node, Edge\}$
$Node := \{Root, NTNode\}$
$Root := \{\exists! x : x \in Data\}$
$NTNode := \{x : x \in Data - Root\}$
$Edge := \{(x, y) : x \in Node, y \in NTNode\}$
CONSTRAINT: There is a unique sequence of edges from the Root to an NTNode.

3.3. Definitions of MDM

The MDM is a graph data structure that is defined in a representational or syntax level and a categorical or semantic level.

3.3.1. Format/Syntax

In a representational level, MDM is defined as a tetra-term or a four-tuple data unit as in Fig.1.
The TYPE is a name of semantic categories defined in the next subsection.

$$MDM := (ID, TYPE, Value1, Value2)+;$$

Figure 1: Syntax of MDM

3.3.2. Semantics/Categories

In a semantic level, MDM is defined as a type of graph as in Fig.2.[2]

$$MDM := \{Node, Path\}$$
$$Node := \{eN, aN, tN, dN\}$$
$$Path := \{eP, aP, tP, lP\}$$

Figure 2: Semantics of MDM

Each semantic unit consists of sub-categories as in Fig.3.

$eN :=$ {ID, "EN", TagType, NodeName}
$aN :=$ {ID, "AN", AttName, AttValue}
$tN :=$ {ID, "TN", "TXT", TextString}
$dN :=$ {ID, "DN", "DOC", FileName}
$eP :=$ {ID, "EP", IdOfElementNode, IdOfElementNode}
$aP :=$ {ID, "AP", IdOfElementNode, IdOfAttNode}
$tP :=$ {ID, "TP", IdOfElementNode, IdOfTextNode}
$lP :=$ {ID, "LP", IdOfElementNode, IdofElementNode}
TagType := {"DCL", "PI", "TAG", "EMP"}

Figure 3: Categories of Marked-up Data

A category of Node in MDM has four groups; Element Node(eN), Attribute Node(aN), Text Node(tN), and Document Node(dN). As a category of Path in MDM, there are four groups; Element Path(eP), Attribute Path(aP), Text Path(tP), and Link Path(lP). The dN is a unique and it can be a member of Value1 of element paths(eP) and text paths(tP).
The element node has further subcategories; a declaration element(DCL), a processing instruction(PI), a normal element(TAG), and an empty element(EP). For the present, the experimental software[3] does not suppose a full-spec XML data as an input data. For example, it does not process entities and regards complex definitions such as CDATA, comments, and parameter entities just as a declaration.
A link path(lP) is an ordered pair of values of IDREF and ID attributes. [4] The argument of $Value2$ can be not only an element with an ID attribute but also an object indicated by a URL format.[5]
Provided an XML data with an instance structure(Fig.4), it is converted onto the MDM as Fig.5.

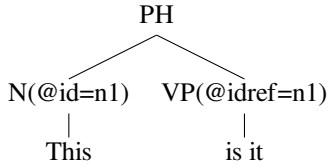

Figure 4: An Instance Structure

```
1,DN,DOC,test.txt
2,EN,TAG,PH
3,EP,1,2
4,EN,TAG,N
5,EP,2,4
6,AN,id,n1
7,AP,4,6
8,TN,TXT,This
9,TP,4,8
10,EN,TAG,VP
11,EP,2,10
12,AN,idref,n1
13,AP,10,12
14,TN,TXT,is it
15,TP,10,14
16,LP,10,4
```

Figure 5: MDM output of Fig. 4

MDM starts from a document node with a file name as $Value2$. The order of each element is indicated by the order of ID number. A link path(e.g. the data unit with ID 16 in Fig.5) indicates a relation from the ID of the element with an IDREF type attribute.[6] to the ID of the element with an ID type attribute[7]

3.4. New Concept of Marked-up Data Syntax

We introduce a new marked-up data format that does not require a root element in their scheme, which means a marked-up data can be inserted within any place of text data. [8] The root-element-less scheme permits a text-centric data model with a variety of marked-up data schemes defined by multiple authorities. In a text data there can be multiple partial marked-up instances with a variety of schemes. This feature is appropriate for a pivot

[2]The idea of MDM was introduced in LingDy Project (Ohya, 2009). This paper is the first to define the definition.

[3]The details are in the section 7.

[4]This simple link structure indicated by IDREF and ID attributes defined in SGML(Goldfarb, 1990) has been taken over through a complex link system defined in HyTime(ISO, 1992a) to the present HTML5, and this directed edge is now a common and only model of a link on the web. Our proposals obey this policy, which means not to seek a possibility of an undirected link adopted in HyTime.

[5]As mentioned later, the experimental software does not support URL resolving for now.

[6]In this case the ID 10 is for this element.

[7]In this case the ID 4 is for this element

[8]This new syntax is based on a tree structure in the same way of an XML syntax. This idea was proposed at the TEI Day in Kyoto (Ohya, 2006). In the paper, four guidelines for marked-up data syntax and scheme used in digital humanities are proposed; (1) no root element, (2) introducing anonymous element(nameless tag), (3) well-formedness is only required, (4) flat scheme.

data format for multiple conversions. And, as explained later, we need this feature for the L2I conversion.
In the humanities including linguistics, researchers make annotated data at all kinds of intellectual level such as observing phenomena, reading texts, writing thinking down, editing data, writing articles and so on. These kinds of annotations can be regarded as a trajectory of intellectual activities. A scheme of the data with such annotations usually can not be defined before starting research activities. In a domain of digital humanities, this kind of process of defining data units is called a text-centric approach, and the defined is called a posteriori scheme. On the other hand, especially in a computer science or engineer, a data unit is defined beforehand in processing, which can be called a data-centric approach, and a priori scheme. In an XML world, for the former a well-formed document and for the latter a valid document are presented respectively.
For MDM to be in a more flexible environment in writing marked-up data than one with a well-formed instance, we introduce an idea of scheme-less data model. This philosophy gives researchers a chance to annotate any comments into any place in a body text. [9]
As an example for a root-element-less and scheme-less data model, we can make a data like Fig.6.

```
The world tour
<PH><N id="n1">This</N>
<VP idref="n1">is it</VP></PH>
```

Figure 6: Instance without a Root Element

The MDM of that instance is like Fig.7.

3.5. L2I Conversion on MDM

A directed link can be an ordered pair. We use this nature as a structural information of an instance. That is, an IDREF-ID relation can be converted to a super-sub relation in an instance structure. The element with an ID becomes a child element of the element with an IDREF of which value is the same as the ID's. For example, the data of Fig.6 can be converted into the marked-up instance in Fig.8
Because of a root-element-less and scheme-less policy, the converted partial instance can be added into the original instance in any place. In this example, the converted partial instance emanated from a link structure appears in the end of the original instance. In MDM, a link is indicated as a link path(lP), thus the L2I conversion is done just by changing the type of MDM units from lP to eP.[10]
As you can see in Fig.8, the new instance consists of the original instance and the resultant of the L2I conversion, thus the referent element with a referred value of ID occurs at least twice in the total resultant instance. And, if the resultant element has a complex content model, the converted resultant also takes over the complexity. Therefore, the resultant of the L2I conversion may be ugly and difficult to be a target for a new conversion.[11] The degree of usability of the L2I conversion depends on an instance structure of a link model.

```
1,DN,DOC,test.txt
2,TN,TXT,The World tour \n
3,TP,1,2
4,EN,TAG,PH
5,EP,1,4
6,EN,TAG,N
7,EP,4,6
8,AN,id,n1
9,AP,6,8
10,TN,TXT,This
11,TP,8,10
12,TN,TXT,\n
13,TP,1,12
14,EN,TAG,VP
15,EP,4,14
16,AN,idref,n1
17,AP,14,16
18,TN,TXT,is it
19,TP,14,18
20,LP,14,6
```

Figure 7: MDM output of Fig. 6

```
The world tour
<PH><N id="n1">This</N>
<VP idref="n1">is it</VP></PH>

<VP idref="n1">is it
 <N id="n1">This</N>
</VP>
```

Figure 8: Instance made from Fig.4

4. Applications of MDM

MDM is originated for the purpose of a pivot data format of marked-up data conversion. In this paper there are two examples of the usage of MDM as a pivot for data conversion; ELAN data and ToolBox data.

4.1. ELAN

ELAN adopts a standoff-style XML data format for its output data format such as Fig.9. In this instance, the

[9] As a result, researchers can write down annotation as their own way and, if needed, they define a scheme for the annotated data from the instance after the validation. The cost of learning the schemes proposed as standards or in guidelines usually expands beyond the benefit language researchers expect. Thus, the proposed schemes have remained in the format at institutions for archives. Individual researchers seek a way to record their data on their own way, and the data is ensured to be converted easily to a format expected in a social world. This order of preference is very important, i.e. a personal data and an archival data.

[10] And, in order to get the new resultant such as Fig.8, we have to add a new element path (eP) from the document node(dN) to the resultant instance(eN) of the L2I conversion.

[11] For example, if there is a loop path in a link structure, the L2I conversion collapses.

```
<ANNOTATION_DOCUMENT>
  <TIME_ORDER>
    <TIME_SLOT TIME_SLOT_ID="ts1"
               TIME_VALUE="180">
    </TIME_SLOT>
    <TIME_SLOT TIME_SLOT_ID="ts2"
               TIME_VALUE="350">
    </TIME_SLOT></TIME_ORDER>
  <TIER>
    <ANNOTATION>
      <ALIGNABLE_ANNOTATION
          ANNOTATION_ID="a1"
          TIME_SLOT_REF1="ts1"
          TIME_SLOT_REF2="ts2">
        <ANNOTATION_VALUE>
          the</ANNOTATION_VALUE>
      </ALIGNABLE_ANNOTATION></ANNOTATION></TIER>
</ANNOTATION_DOCUMENT>
```

Figure 9: Input ELAN data(part)

length of the link-ladder is one, but the starting element with "a1" as the value of the attribute ANNOTATION_ID has two IDREFs.Thus, the L2I conversion makes an instance of Fig.10. The resultant instance is

```
<ALIGNABLE_ANNOTATION
     ANNOTATION_ID="a1"
     TIME_SLOT_REF1="ts1"
     TIME_SLOT_REF2="ts2">
  <ANNOTATION_VALUE>the</ANNOTATION_VALUE>
  <TIME_SLOT TIME_SLOT_ID="ts1"
             TIME_VALUE="180"></TIME_SLOT>
  <TIME_SLOT TIME_SLOT_ID="ts2"
             TIME_VALUE="350"></TIME_SLOT>
</ALIGNABLE_ANNOTATION>
```

Figure 10: Resultant of L2I conversion(part)

easy to check the time information recorded distantly from the annotated text in the ELAN format. This kind of gathering information is useful especially in personal data handling as their own research activities.

4.2. ToolBox

ToolBox permits users to set any number of data records and the structure of which can be also defined as they like. And, the total set of the data records can form an interlinear data structure, which is familiar to language researchers especially descriptive linguists. For example, the data of ToolBox is in Fig.11.[12]

```
\ref,ShAG19970726T.001
\speaker-id,ShAG
\tx-cyr,Пулундиэ тэрикиэдиэньэ модоҥи.
\tx-ipa,pulundie terikiedien'e modo ŋi.
\ps,n,,n,,vi
\kw,S,,###,,Vi
\mr,pulun,-die,terikie, -die -n'e,modo,-ŋ i
\u,pulut,-die,terike, -die -n'e1,modo,-ŋi 2
\gl_eng,old.man, -DIM:NOM,old.woman,-DIM-COM,sit,
-PL.IND.INTR.3
\ft_eng,An old man and an old woman lived together.
\ft_rus,Старичок со старушкой жили.
```

Figure 11: ToolBox data

In order to handle the data made on ToolBox, we set new semantic categories of MDM for it such as Fig.12. Due

$eN :=$ {ID, "EN", TagName, TagValue}

$eP :=$ {ID, "EP", IdOfElementNode, IdOfElementNode}

Figure 12: Categories for ToolBox data

to the categories, the data of Fig.11 can be represented on MDM like in Fig.13. Once the data is denoted on

```
1 D doc /home/user/sampleToolBox2.csv
2 EN ref ['ShAG19970726T.001']
3 EP 1 2
4 EN speaker-id ['ShAG']
5 EP 2 4
6 EN tx-cyr ['Пулундиэ тэрикиэдиэньэ модоҥи.']
7 EP 2 6
8 EN tx-ipa ["pulundie terikiedien'e modoŋi."]
9 EP 2 8
10 EN ps ['n', '', 'n', '', 'vi']
11 EP 2 10
12 EN kw ['S', '', '###', '', 'Vi']
13 EP 2 12
14 EN mr ['pulun', '-die', 'terikie', " -die -n'e", 'modo', '-ŋi']
15 EP 2 14
16 EN u ['pulut', '-die', 'terike', " -die -n'e1", 'modo', '-ŋi2']
17 EP 2 16
18 EN gl_eng ['old.man', ' -DIM:NOM', 'old.woman', '-DIM-COM', 'sit',
'-PL.IND.INTR.3']
19 EP 2 18
20 EN ft_eng ['An old man and an old woman lived together.']
21 EP 2 20
22 EN ft_rus ['Старичок со старушкой жили.']
23 EP 2 22
```

Figure 13: MDM of Toolbox data

MDM, it can be converted into the other data format with ease. For example, the present experimental application mentioned afterward converts the data in Fig.13 into the following three types of data formats.

The first type is in Fig.14 is a simple data model that reflects directly the original data structure. Each record in ToolBox is converted into an element with the name same as the tag name.

```
<doc idref="/home/user/code/sampleToolBox2.csv">
 <ref id="ShAG19970726T.001">
  <speaker-id class="ShAG19970726T.001">ShAG</speaker-id>
  <tx-cyr class="ShAG19970726T.001">Пулундиэ тэрикиэдиэньэ модоҥи.</tx-cyr>
  <tx-ipa class="ShAG19970726T.001">pulundie terikiedien'e modoŋi.
</tx-ipa>
  <ps class="ShAG19970726T.001">n n vi</ps>
  <kw class="ShAG19970726T.001">S ### Vi</kw>
  <mr class="ShAG19970726T.001">pulun -die terikie -die -n'e modo -ŋi</mr>
  <u class="ShAG19970726T.001">pulut -die terike -die -n'e1 modo -ŋi2</u>
  <gl_eng class="ShAG19970726T.001">old.man -DIM:NOM old.woman -DIM-COM
sit -PL.IND.INTR.3</gl_eng>
  <ft_eng class="ShAG19970726T.001">An old man and an old woman lived
together.</ft_eng>
  <ft_rus class="ShAG19970726T.001">Старичок со старушкой жили.</ft_rus>
 </ref>
```

Figure 14: Simple output

The second type is a more complex data model that reflects sub-units in each record as an independent element like in Fig.15. This type can be regarded as an interlinear data in an XML format.

The third type is a tree structure reflacting multiple hierarchies in language units such as Fig.16. This tree model presupposes the definition of the semantics of data units. However, in ToolBox, there is no processible description or information in the data itself. Thus, this conversion needs additional instructions. For example, the information which unit is a super element and which unit is the sub element is needed. The point is MDM works as the base of data conversion for this kind of type.

[12]The original language data of this example is made by Iku Nagasaki.

```
<doc idref="/home/user/sampleToolBox2.csv">
 <ref id="ShAG19970726T.001">
  <speaker-id class="ShAG19970726T.001">ShAG</speaker-id>
  <tx-cyr class="ShAG19970726T.001">
Пулундиэ тэрикиэдиэньэ модоӈи.
</tx-cyr>
  <tx-ipa class="ShAG19970726T.001">
   pulundie terikiedien'e modoŋi.</tx-ipa>
  <ps class="ShAG19970726T.001">
   <seg n="1">n</seg>
   <seg n="2"></seg>
   <seg n="3">n</seg>
   <seg n="4"></seg>
   <seg n="5">vi</seg>
  </ps>
  <kw class="ShAG19970726T.001">
   <seg n="1">S</seg>
   <seg n="2"></seg>
   <seg n="3">###</seg>
   <seg n="4"></seg>
   <seg n="5">Vi</seg>
  </kw>
```

Figure 15: Complex output

```
<doc idref="/home/user/sampleToolBox2.csv">
 <ref value="ShAG19970726T.001">
  <speaker-id value="ShAG"></speaker-id>
  <tx-cyr value="Пулундиэ тэрикиэдиэньэ модоӈи."></tx-cyr>
  <tx-ipa value="pulundie terikiedien'e modoŋi."></tx-ipa>
  <ft_eng value="An old man and an old woman lived together."></ft_eng>
  <ft_rus value="Старичок со старушкой жили."></ft_rus>
  <ann value="">
   <ps value="n">
    <kw value="S">
     <mr value="pulun">
      <u value="pulut">
       <gl_eng value="old.man"></gl_eng>
      </u>
     </mr>
     <mr value="-die">
      <u value="-die">
       <gl_eng value=" -DIM:NOM"></gl_eng>
      </u>
     </mr>
    </kw>
   </ps>
```

Figure 16: Tree output

4.3. Note: Tree Structure and Language Data

MDM supports the L2I conversion because of being a general data format for any marked-up data especially with a link information. This mechanism ensures the data in a standoff-style format can be easily reused in a personal data management system. However, if there is no philosophies or semantic rules of the link structure itself, the resultant instance of the L2I conversion becomes something unnatural for our intuition or naive understanding. And, this is not so rare at present since there is no guidelines or standards for a metadata defining a link used in marked-up data. For example, the way of representing a link in the instance of ELAN is defined based on authentic traditional Link strategies(DeRose and Durand, 1994)(ISO, 1992a). On the other hand, the XML data generated by FLEx adopts a reversed tree model on a link structure. That is, a morpheme can be a starting unit of a link and a word becomes the target or the referent of the link, which makes an instance unnatural to our intuition of an image of tree. If a link is used as an alternative way to represent another data structure than one the instance has, the usage of a directed link should be considered more carefully. [13]

[13]It seems to be a critical issue of lacking standards for a metadata to define a link used in an XML instance. The consideration under a link system in establishing HyTime is still informative.

5. Constellation of Language Data; Models and Formats

Following sections are introductions to research underway to establish a digital environment for a data management system for language documentation with MDM.

MDM and the L2I conversion compensate for deficiencies of data in archives or repositories. They contribute to extracting language resources from a standoff-style data. However, they alone does not satisfy a solution to a personal data management environment. We need more mechanisms for language researchers' requirements (a1) and (a3). As reconfirmed before, sharing a data format is a good way for data preservation. But we also confirmed the existing standoff-style data format may not work well for it. Thus, an alternative shareable data format is expected. For now, as requirements for such a format, we presuppose as follows; (b1) a flat structure as much as possible, (b2) no complex link information in the language data content itself, and (b3) an interlinear format. The (b1) contradicts the (a1)[14] at first glance. The requirement (a1) comes from observation of fieldnotes and rambling notes(Ohya, 2016). But this (b1) comes from our experience in handling data in language documentation. In this paper, the existance of the requirement (b1) is confirmed. We have to seek the sublation of (a1) and (b1). The (b2) must be carefully understood: avoidance of using a link in a language data content does not mean a prohibition on a link in data.[15] The reason of (b3) is explained before.[16]

As such a data model, we define CORPUS(Ohya, 2015b)(Ohya, 2016). And, in order to handle time information that is represented in a link structure we demand to remove from an archival data, we devise GIST as a way to describe time information(Ohya, 2015b)(Ohya, 2016).

5.1. General Corpus Format:CORPUS

CORPUS is a data format for an interlinear description style of language data, which is a reflection of a ToolBox format. Thus, for those who are familiar with ToolBox, this format is easy to recognize. The syntax of CORPUS is in Fig.17.[17]

A language data or corpus(CORPUS) consists of sentences(SNT), which is a set of annotations that includes one target data(annotation line;AL) and multiple additional annotations(alternative annotation line; AAL, and annotation with its own structures; ANN).[18] A data in CORPUS is not a marked-up data, but just a plain text data. [19]

[14]flexible hierarchy

[15]A link information is expected in a metadata for connecting multiple data contents.

[16]Due to the limitation of the page length, we can not explain the details on this topic in this paper.

[17]This definition is a slightly revised version of (Ohya, 2016).

[18]The ANN is different from the ToolBox format in order to be equipped with one more structural hierarchy.

[19]Therefore, CORPUS takes over the drawback or restric-

$CORPUS := SNT+;$
$SNT := ID, AL, (ALL|ANN)+, ER;$
$AL|AAL :=$ {any strings};
$ANN :=$ ({any strings}, $DLM)+;$
$ER :=$ {empty record};
$DLM :=$ {delimiters such as comma or space};

Figure 17: Syntax of CORPUS

5.2. General Time-Span Format:GIST

A standoff-style is based on the policy that one referent is shared with multiple references by links. In the case of ELAN and FLEx adopting this style, time point information as a referent is shared on a link structure as an semantic model other than an instance structure. As a type of time information there are absolute and relative time information. We do not find a good solution of a way to denote a relative time information, but, as for an absolute time information, we devise the GIST(general information of sub-time) format. The syntax of GIST is in Fig.18.[20]

$GIST := I+;$
$I := (Name|TP)+;$
$Name :=$ {any strings};
$TP := (TIME, TIME);$
$TIME := hh : mm : ss([.,]d+)?;$

Figure 18: Syntax of GIST

Each record includes time information(I) that can be represented by a name, usually a file name, or a pair of time-point that indicate a time period(TP). The semantics of the record I is defined in Fig.19.

$$[\![I_1 I_2]\!] := I_1 \supseteq I_2$$

Figure 19: Semantics of GIST

In a record I, more than one time information can appear in a sub-set relation, that is, a time period of I_1 is the same as or larger than I_2. The combinations of a sequence of a name and a pair of time-point are as Fig.20. An example of GIST descriptions is as Fig.21.[21]

$[\![Name_1, Name_2]\!] := Name_1 \equiv Name_2$
$[\![Name, TP]\!] := Name \supseteq TP$
$[\![TP, Name]\!] := TP \equiv Name$
$[\![TP_1, TP_2]\!] := TP_1 \supseteq TP_2$

Figure 20: Semantic Categories of GIST

```
00:00:01.2,00:00:50.3
00:00:01.2,00:00:50.3,file1.wav
00:00:01.2,00:00:50.3,00:00:00,00:00:15
00:00:01.2,00:00:50.3,00:00:00,00:00:15,file2.wav
00:00:02.2,00:00:50.3,file4.wav,00:00:01.2,00:00:10.32
```

Figure 21: Sample descriptions of GIST

The nest of time information in a record helps indicate a position of the unit in a partial domain of language sound, e.g. a word in a sentence, a syllable in a morpheme, and so on. This notation permits redundancy of a time information for a specific time period, but it is entirely within users' decision.
A time information should be recorded independently from a language data transcribed from the sound data. This stance comes from the observation in making digital documents for Digital Humanities, especially a field of Digital Histories. In historical records there are many types of words to represent time information; e.g. a phrase for a specific time point, an ambiguous time point, a period of time, a vague period of time, and so on. In order to handle a variety of time indications, it is a best way to make a matching table for the descriptions an independent file, which can be separated from the processing mechanisms. The matching file itself is a result of historical studies and could be updated whenever the research progresses. [22] [23]

6. Strategies for Data Storage

In order to realize a personal data management environment for language documentation, it is requisite to investigate the sublation between (a1) a flexible scheme and (b1) a flat scheme under the condition of avoiding application sharing but adopting a format sharing. It has puzzled us for over a decade. Now, we re-evaluate the proposition of avoiding application sharing.
Many language researchers satisfy the extent of structural hierarchy in a ToolBox style, but there is no flexibility of the data scheme in its hierarchy. However both data schemes are on a tree structural data model. And as a result of making time information independent from the language data itself, language researchers have to ensure the connections that are kept by applications such as ELAN, FLEx and so on. If these are set as requirements for a new application that supports a personal data management system, we can discover a hierarchical database

tion ToolBox has; a definition of a component of a record is left for users, and the degree of hierarchy is at best practically the two in each record. We are studying the solution to them as mentioned afterward.

[20] This definition is slightly revised to be more simple from one in (Ohya, 2016).

[21] In this example, an indication for an input sound data is omitted. In the present experimental application, the processed sound source is indicated as the argument at the call.

[22] It implies that this matching table will ever change as research continues.

[23] This way can be a solution to handle a relative time information.

system as a candidate to solve the problems.

A hierarchical database has longer history than the present popular database system based on a relational model, and also has the properties such as (c1) a tree structural data model, (c2) schema-less, (c3) a flexible data structure, (c4) text- or document-centric data, (c5) the longest life-span of data in database systems, and (c6) fast. MUMPS(ANSI, 1984)(ISO, 1992b) has been well known, and GT.M(fis-gtm)(GT.M, 2021), YottaDB(YottaDB, 2021) and Caché(InterSystems, 2021) are the well-known successors. And, MongoDB(MongoDB, 2021), as one of the new hopes for database systems, can also be regarded as the member. As drawbacks the hierarchical database has there are (d1) no mechanism to keep relations between data sets[24] and (d2) difficulty to keep the consistency of stored data. The (d1) is not a disadvantage but rather a natural property for language documentation and Digital Humanities. Keeping the matching table is a research activity itself. The (d2) seems to be a trade-off between flexibility language researchers want and the present rigidness a relational database has. As far as we know there is no guideline that language researchers can refer to in using a hierarchical database system for their own language storage. We are engaged in evaluating GT.M and MongoDB as a personal data management system for language documentation activities(Grants-in-Aid for Scientific Research, Japan, No.20K00619).

7. Appendix A: Experimental Applications

To test the feasibility and usability of MDM and the L2I conversion, we have implemented an experimental application named docsci.mdm as a Python package, which can be installed with "`pip ␣ install ␣ -i ␣ https://test.pypi.org/simple/ ␣ docsci==0.1.4`".[25] This package supports an XML and a CSV file as the input data. In the case of an XML data, after importing the package (`import docsci.mdm as mdm`), making an object (`obj = mdm.MDM()`) and reading an input data (`obj.read_file(FILE_NAME)`), we can call parsing (`obj.parse()`), making the MDM data model (`obj.make_mdm()`), and making a new output XML data (`obj.make_instance()`).[26] The L2I conversion can be done by calling the `add_linkpath()` method before calling the `make_instance()`.[27] In the case of a CSV file supposed to be on a CORPUS model, we can call the functions `read_csvFile(FILE_NAME)`, `parse_csv()`, and `make_dmd_csv()`. The MDM of a CSV data can be transformed into the three types of data formats shown in the section 4.2.[28] This package supports a small subset of XPath, by the function `get_instance_from_xpath(XPATH)` to MDM.

As an application to handle a GIST formatted data, we made a Java-based program named Sclip(sound clip). [29]

[24]The data set is called, in RDB a table, in MUMPS a global variable, and in MongoDB a document respectively.

[25]The ␣ indicates spaces. Please note that the documentations of this package is of very poor quality and quantity now.

[26]The instance made through the above steps is stored in `obj.mdi`, then you can check the content with `print(obj.mdi)`.

[27]This package supports idref, url, href, src, and link as default attribute names for IDREFs. You can add a new name by the function `obj.idrefs.add(NEW_NAME)`.

[28]`make_instance_simple(1)`, `make_instance_complex(1)`, `make_instance_tree(1)`

[29]`https://docsci.infon.org/stack/Sclip.zip`

8. Appendix B: Acknowledgements

This research has been supported by the following research funds: Japan Society for Promotion of Science Grants-in-Aid for Scientific Research, No. 23401025, 26370512, 17K02749, and 20K00619, which are titled "A study of digital archive environment and language documentation for minority languages in North-East Eurasia", "A study of documentation theories and practices on minority languages in Siberia", "A study of language documentation for descriptive or field linguistics to establish a theory for recording of emerging data", and " A study of language documentation with development of a text-based database system for long-term use and maintenance" respectively. I am deeply grateful to Iku Nagasaki, Professor of Nagoya University, Chikako Ono, Professor of Hokkai-Gakuen University, Ritsuko Kikusawa, Professor of National Museum of Ethnology, and Itsuji Tangiku, Professor of Hokkaido University, Japan.

9. Bibliographical Reference

ANSI. (1984). Programming Language MUMPS, ANSI X11.1-1984. Technical report, ANSI.

Audacity. (2021). Audacity [computer software]. `https://www.audacityteam.org/`.

Bird, S. and Liberman, M. (1999). A formal framework for linguistic annotation. Technical Report MS-CIS-99-01, University of Pennsylvania.

Boersman, P. and Weenink, D. (2011). Praat [computer software]. `https://www.fon.hum.uva.nl/praat/`.

DeRose, S. J. and Durand, D. G. (1994). *Making Hypermedia Work: A User's Guide to HyTime*. Kluwer Academic Publishers.

Goldfarb, C. F. (1990). *The SGML Handbook*. Oxford University Press.

GT.M. (2021). fis-gtm [Computer Software]. `https://sourceforge.net/projects/fis-gtm/`.

Ide, N. and Pustejovsky, J. (2017). *Handbook of Linguistic Annotation Vol.1*. Springer Nature.

Ide, N. (1998). Corpus encoding standard: SGML guidelines for encoding linguistic corpora. In *In Proceedings of the First International Language Resources and Evaluation Conference*, pages 463–70.

InterSystems. (2021). Caché [Computer Software]. https://www.intersystems.com/products/cache/.

ISO. (1992a). ISO 10744 Hypermedia/Time-based Structuring Language (HyTime). Technical report, ISO.

ISO. (1992b). ISO/IEC 11756 International technology –Programming language– MUMPS. Technical report, ISO.

ISO. (2006). ISO/DIS 24610-1 Language Resource Management – Feature Structures – Part1: Feature Structure Representation. Technical report, ISO.

ISO. (2011). Language resource management – feature structures – part 2: Feature system declaration. Technical report, ISO.

ISO. (2012). Language resource management – linguistic annotation framework (LAF). Technical report, ISO.

Max Planck Institution. (2021). ELAN [Computer Software. https://archive.mpi.nl/tla/elan.

MongoDB. (2021). MongoDB [Computer Software]. https://www.mongodb.com/.

Ohya, K. (2006). Markup problems: Syntactical analysis and steps to their resolution (in Japanese). In *Report of TEI Day in Kyoto 2006*, pages 29–39. Kyoto University, 12.

Ohya, K. (2008). Management of links between link elements to represent correlation on link structures (in Japanese). *IPSJ SIG technical reports*, Vol.2008, No.100:15–22.

Ohya, K. (2009). Data structure for minority language corpora (in Japanese). *IPSJ symposium series*, Vol.2009, No.16:115–122.

Ohya, K. (2011). Missing services in language documentation in terms of information processing – a report of lingdy project – (in Japanese). *IPSJ symposium series*, Vol.2011, No.8:59–66.

Ohya, K. (2015a). Corpus sharing strategy for descriptive linguistics. *Journal of JADH*, Vol.1, No.1:68–85.

Ohya, K. (2015b). A general format for time information to the first-class data of general linguistics. Technical report, ICLDC4. http://hdl.handle.net/10125/25368.

Ohya, K. (2016). Data formats and management strategies from the perspective of language resource producers –personal diachronic and social synchronic data sharing. In *Proceedings of LREC 2016*, pages 3243–3248. LREC2016.

Summer Institute of Linguistics. (2021a). FLEx [Computer Software]. https://software.sil.org/fieldworks/.

Summer Institute of Linguistics. (2021b). ToolBox [Computer software]. https://software.sil.org/toolbox/,.

Text Encoding Initiative. (1994). *The TEI Guidelines P3*. TEI.

YottaDB. (2021). YottaDB[Computer Software]. https://yottadb.com/.