

# *Ontologies for a Global Language Infrastructure*

Yoshihiko Hayashi

Osaka University and NICT Language Grid Project

mailto:hayashi@lang.osaka-u.ac.jp

## 1 Introduction

With the recent developments of the Semantic Web and progresses of the associated methodologies and standards, demands for an open and distributed infrastructure for sharing language resources and technologies can be addressed now on a new basis (Buitelaar et al., 2003; Calzolari, 2008). In this article, we call such an infrastructure a *global language infrastructure* (GLI). GLI should accommodate language resources and technologies world-wide. A GLI thus should inherently address multilingual and multicultural issues.

More precisely, a GLI is an open and web-based software platform to which resources can be easily plugged in, and on which tailored *language services* can be efficiently composed, disseminated and consumed. Here a language service simply means a web service whose functionalities are generally related to human language; it can range from simple dictionary access to more complicated linguistic analysis, as well as conversion of linguistic expressions such as translation or paraphrasing.

We can mention the following initiatives/projects as examples of an obvious effort towards such a language infrastructure:

- *CLARIN*<sup>1</sup> is committed to establish an integrated and interoperable research infrastructure of language resources and technology. It aims at addressing the current fragmentation by offering a stable, persistent, accessible and extendable infrastructure that will enable the development of “e-Humanities”.
- *Language Grid*<sup>2</sup> provides a language infrastructure on which language services that are useful in intercultural collaboration can be composed, delivered, and utilized. On the Language Grid, existing language data resources, NLP tools/systems and newly created community-based resources can be efficiently and effectively combined (Ishida, 2006). In addition, the Language Grid presents an operation model to address complicated issues associated with intellectual property rights and contracts (Ishida et al., 2008).

These two initiatives share issues of interoperability and reusability of language data resources and NLP tools/systems, even though their primary objectives are totally different. This calls for an opportunity to work out a common strategy for these crucial issues.

With this background, this article argues that a GLI should be ontology-based, and presents a high-level configuration of the ontologies, which are integrated into a

---

<sup>1</sup> <http://www.clarin.eu/>

<sup>2</sup> <http://langrid.nict.go.jp/>

comprehensive *language service ontology*. This article also examines relevant ongoing international standards, and discusses how these frameworks can be ontologized and incorporated into the comprehensive language service ontology.

## **2 Dimensions of GLI**

### **Objectives of a GLI**

Needs for a language infrastructure have originally emerged from research fields including NLP and a range of e-sciences, which require mining from textual resources. For example, Klein and Potter (2004) presented two use cases; one is a workbench for NLP researchers, and the other is a text-mining tool for e-science researchers who are not necessarily NLP experts.

More recently, CLARIN explicitly targets its users to communities of e-humanities, and tries to offer its services to:

- The different communities of linguists to optimize their models and tools to the benefit of all who are using language material,
- Humanities scholars in the broad sense to facilitate access to language resources and technology, and
- The society as a whole to enable lower thresholds to multicultural and multilingual content.

In contrast, the Language Grid has been launched for providing a language infrastructure for supporting verbal, particularly cross-language, communications that are observed in activities of intercultural collaboration. To achieve this goal, the Language Grid provides an environment in which existing NLP tools/systems and newly created community-based language data resources can be efficiently combined. A number of communication tools are publicized on the project web site.

Here we should remark that (1) the user of a GLI is not necessarily an NLP expert, and (2) not only language data resources but NLP tools/technologies and their useful combinations are involved in a GLI.

### **Types of users in a GLI**

Users, or participants, of a GLI can be classified into the following types:

- A language resource provider who disseminates a language resource or NLP functionality in the form of a language service by wrapping it as a web service,
- A language service composer who composes a composite web service by combining atomic language services, and
- A language service end user who simply consumes a language service.

From a language infrastructure perspective, it is of crucial importance to provide useful support for a language resource provider in creating the wrappers, and for language service composers in authoring composite language services. To these ends, a standardized framework for describing language data resources and NLP tools/systems is strongly required (Hayashi, 2007a).

## 2.1 Technical ingredients of a GLI

As implied from the discussions so far, technical ingredients of a GLI are: (1) NLP tools/systems ranging from dictionary access systems and linguistic analyzers to machine translation systems, and (2) language data resources, such as lexicons or corpora. In addition to these, a GLI has to be aware of abstract linguistic objects such as linguistic expression, linguistic annotation or even linguistic meaning, because these types of abstract objects comprise the data to/from NLP tools/systems, as well as content of language data resources.

### Technical architecture of a GLI

The ontologies described in this article have not been intended for a particular language infrastructure. However we expect that the ontologies should be first introduced to an infrastructure like the Language Grid, because it, unlike other research-oriented infrastructures, tries to incorporate a wide range of NLP tools and community-based language resources (Ishida, 2006) in order to be useful for a range of intercultural collaboration activities.

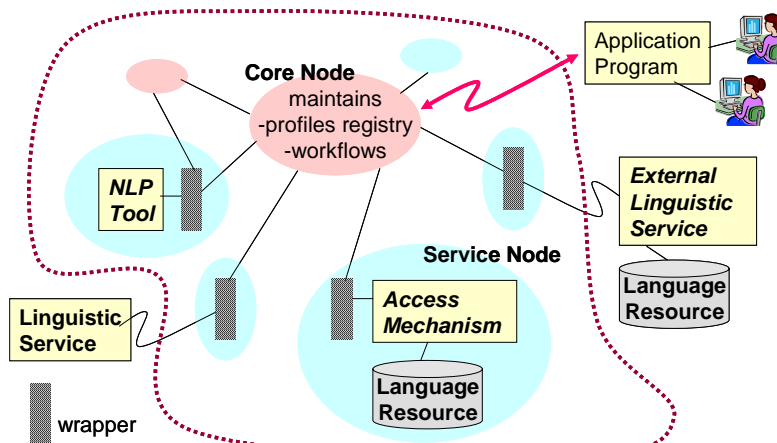


Figure 1: Technical architecture of the Language Grid.

The fundamental technical components in the Language Grid are: (a) external web-based services, (b) on-site NLP core functions, (c) static language resources, and (d) wrapper programs. Figure 1 depicts the general architecture of the Language Grid infrastructure. The technical components listed above are deployed as shown in the figure. Computational nodes in the Language Grid are classified into the following two types as described in (Murakami et al., 2006).

- A service node accommodates atomic language services that provide functionalities of the NLP tool/system running on a node, or they can simply have a wrapper program that consults an external web-based language service.
- A core node maintains a repository of the known atomic language services, and provides service discovery functionality to the possible users/applications. It also maintains a workflow repository for composite language services, and is equipped with a workflow engine.

Given a technical architecture like this, the ontologies will serve as a basis for composition of composite language services, and efficient wrapper generation. A language service ontology will facilitate the wrapper generation processes, which are unavoidable during incorporation of existing general language services or dissemination of newly created community-based language resources. For example, a language service provider who wants to disseminate a new language resource can benefit from the ontology; the relevant part of the taxonomy in the ontology provides guidelines for placing the service in the whole universe of linguistic services. Furthermore, the infrastructure may be able to provide a ‘skeleton code’ of the wrapper program for a major linguistic service type that has a position in the taxonomy. The service provider would be able to develop his/her own wrapper program code by simply refining the provided skeleton program. The most important desideratum for the ontology, therefore, is that it be able to specify the input/output constraints of a language service properly. Such input/output specifications enable us to derive a taxonomy of language service and the associated language resources.

### 3 Ontologies for a GLI

#### Necessity of a comprehensive and standardized ontology

In principle, most of the existing language data resources and NLP tools/systems have been created independently, resulting in a situation where data format, annotation scheme, access method and other features are all idiosyncratic. This obviously will be a burden for establishing a GLI which ensures interoperability and reusability of language data resources and NLP tools/systems. To address this issue, standardization is inevitable: standardized APIs are necessary for NLP tools/systems; standardized data semantics as well as data format are required for language data resources. In addition and importantly, these standards should be designed based on a comprehensive shared ontology, or a set of ontologies, which covers all possible elements of a GLI.

#### Triangular view of a language service

In order to facilitate the development of a comprehensive ontology, it should be divided into appropriate sub-ontologies, each covering a grouped set of elements.

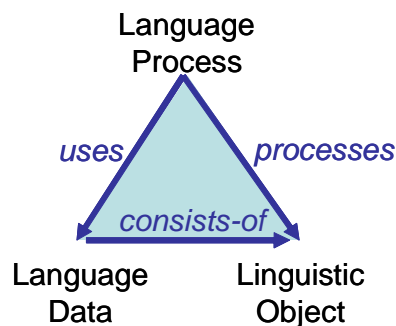


Figure 2: Triangular view of a language service.

Figure 2 shows a triangular view of a language service. Note that a language service is provided by a *language process*, not solely by language data or linguistic objects. Therefore language processes should be placed at the vertex of the triangle. A language process, in general, processes a linguistic expression which may or may not be linguistically annotated. We denote abstract objects such as linguistic expression or linguistic annotation as *linguistic object*. Linguistic objects may comprise a *language data* resource such as a corpus or lexicon; hence it would be utilized by a language process. This triangular view of a language service gives us a foundation on which necessary sub-ontologies are developed.

### Top-level of the language service ontology

We have developed the upper part of the service ontology so far and have been working on detailing some of its core parts. We have adopted OWL (McGuinness and Harmelen, 2004) as the working description language for the moment. We use Protégé<sup>3</sup> as the main tool for working with the service ontology.

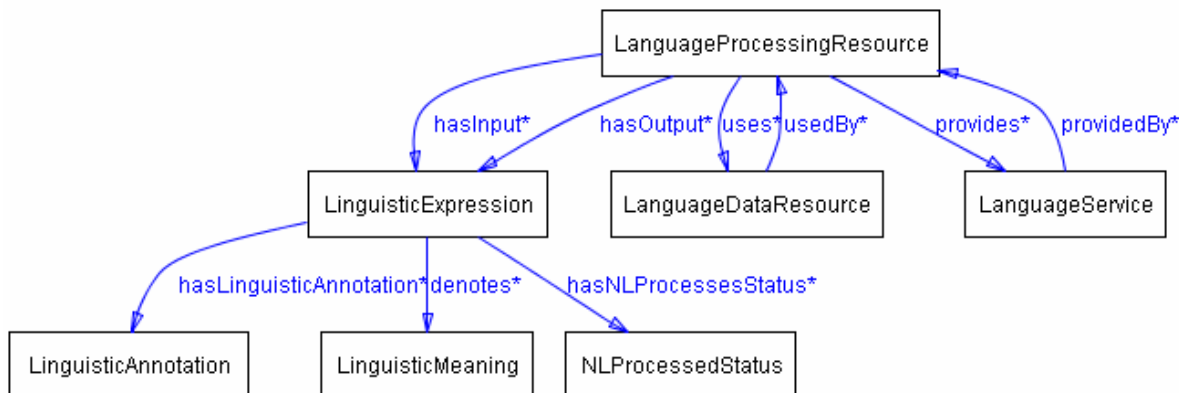


Figure 3: Top-level of the language service ontology.

Figure 3 overviews the top-level of our language service ontology that is configured according to the language service triangle depicted in Figure 2. In the figure and alike in this paper, a rectangle denotes a class, and a directed edge connecting classes indicates the relation between the classes. The label attached to an edge represents the type of the relation; among them, black edges with ‘isa’ labels depict the class hierarchy.

The topmost concept is **LanguageProcessingResource**, which is partitioned into **LanguageProcessingResource** and **LanguageDataResource**. Here, as in GATE (Cunningham et al. 2002), language processing resource refers to programmatic or algorithmic resources, while language data resource refers to data-only static resources such as lexicons or corpora. Although the details have not yet been worked out, meta-information for these two types of resources should be available. The innate relation between these two classes is: a processing resource can use language

<sup>3</sup> All the figures (except Fig.1 and Fig.2) were produced with the OntoViz plugin of the Protégé ontology editor. These tools can be obtained from: <http://protegewiki.stanford.edu/>.

resources. This relationship is specifically introduced to properly define linguistic services that are intended to provide language resource access functions.

Each box in the figure denotes a top-level class in the ontology, which is defined in further detail by a sub-ontology. Among these top concepts, **LanguageService** is the top-most concept. As discussed with the language service triangle, a language service is provided by **LanguageProcessingResource** which takes **LinguisticExpression** as input/output and uses **LanguageDataResource**. Note that a language data resource does not provide a language service by itself; it is always used through an access mechanism which is an instance of some sub-class of the processing resource class.

The fundamental classes, **LinguisticExpression**, **LinguisticMeaning**, and **LinguisticAnnotation** and the innate relations among them are also illustrated in Fig. 2. These play crucial roles in defining functionalities of some types of processing resources and associated language resources. As shown in Fig. 2, a language expression may denote a meaning, and can be annotated by linguistic annotations. Respecting the recent direction of 'NLP as document annotation' (Ide and Romary, 2006), (Klein and Potter, 2004), we also consider that any linguistic analysis is additive.

In addition to these, **NLPProcessedStatus** is introduced especially to represent the so-called IOPE (Input-Output-Precondition-Effect) parameters of a linguistic processor, which is a subclass of language processing resource. The results of a linguistic analysis are encoded as instances of a linguistic annotation class, and the instances are attached to the instance of the expression class. That is, an expression can be annotated by linguistic annotations.

In further detailing the sub-ontologies, we believe it important to incorporate related international standards. In this sense, we have been looking at frameworks for linguistic annotation and lexicon modeling that have been discussed in international standardization bodies. The frameworks for linguistic annotation are incorporated into our ontology not only for specifying the input/output data of NLP tools, but also for defining the content of corpora. On the other hand, the framework for lexicon modeling is introduced to have a formal foundation for developing a taxonomy of lexicon, which obviously forms a subpart of the language data resource ontology.

## 4 Ontology for Linguistic Annotations

### Importance of standard

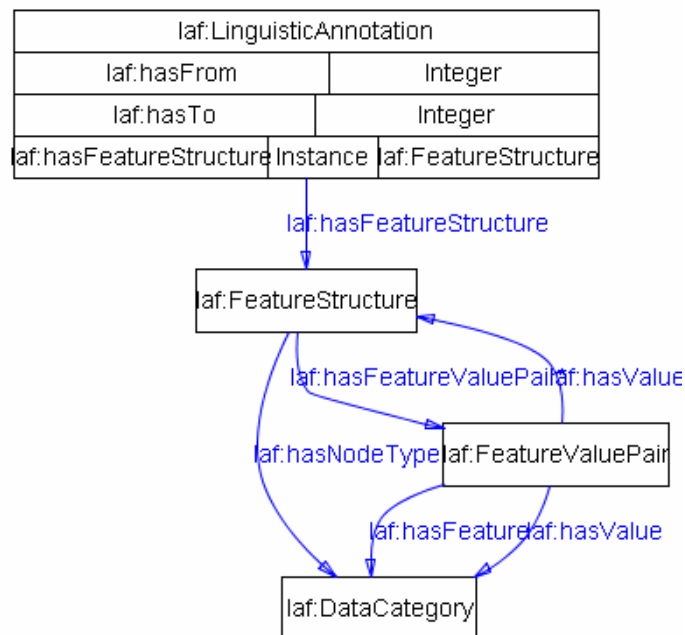
As depicted in Fig 3, a linguistic expression can be multiply annotated by instances of **LinguisticAnnotation** class. This is of crucial importance, because any framework for linguistic annotation has to be able to accommodate multiply layered annotations, given the possibility that the target linguistic expression would be annotated by more than one analyzer, each of which possibly doing its job on a different linguistic level. Among the linguistic objects, ontological configuration for the linguistic annotation should be most carefully designed with respect to the interoperability and reusability of language data resources and NLP tools, because the data to/from a linguistic

analyzer, as well as the content of a language data resource should be represented as linguistic annotation.

Frameworks that are necessary for standardized linguistic annotations have been actively developed and disseminated by the ISO TC37/SC4<sup>4</sup> committee; these include LAF (Linguistic Annotation Framework) (Ide and Romary, 2006), MAF for morphosyntactic annotation (Clément and de la Clergerie, 2005), SynAF for syntactic annotation (Declerck, 2006), and others. Among these, the LAF is the most general *umbrella* framework, and the other frameworks inherit the basic properties of LAF. As these frameworks have not been defined in the form of an ontology, we decided to *ontologize* these frameworks and incorporate them into the language service ontology. Here to ontologize simply means to give OWL specifications to relevant parts of the framework.

### Ontologization of ISO standards: LAF, MAF and SynAF (Hayashi et al. 2008a)

Figure 4 illustrates a high-level configuration of the sub-ontology for linguistic annotations. This configuration corresponds to the LAF framework. As shown in the figure, a linguistic annotation has a start position and an end position for designating the span of annotation in the target linguistic expression<sup>5</sup>. This allows us to implement so-called stand-off annotation, and hence enables multiple annotations on the same data set. It also accommodates a feature structure for representing the



annotation content.

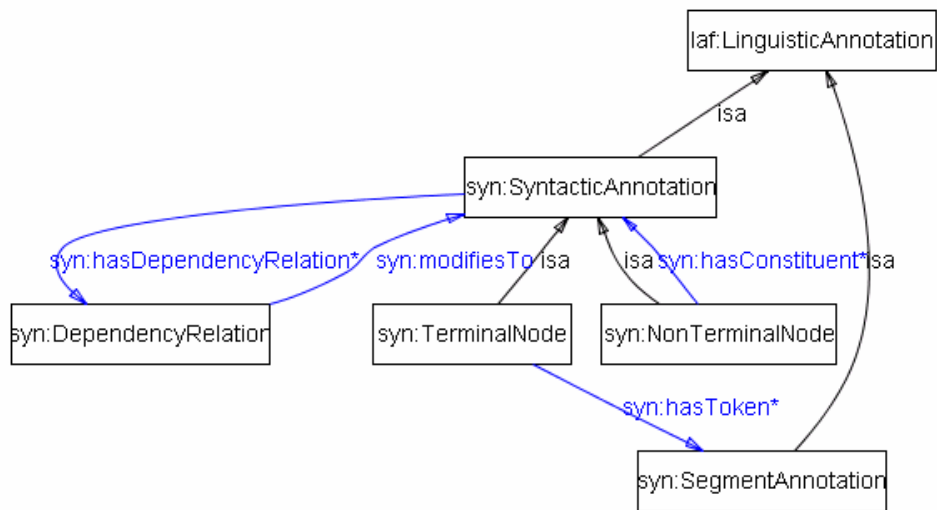
Figure 4: Configuration for LAF.

<sup>4</sup> <http://www.tc37sc4.org/>

<sup>5</sup> In LAF, this is called *primary data*.

As noted in (Declerck et al., 2007), the LAF does not provide specifications for content categories; instead it includes a DCR (Data Category Registry) (Wright, 2004) that contains pre-defined data elements and schemas that may be used in annotations. Current configuration of the data categories does not induce taxonomical structure. Nevertheless the linguistic annotation class should be further organized into subclasses based on which data categories should be included.

Figure 5 summarizes the ontological configuration for MAF and SynAF, introducing classes for segment (**SegmentAnnotation**), syntactic constituent (**SyntacticAnnotation**), and dependency relation (**DependencyRelation**). Note that these classes have been explicitly introduced, although these, in principle, should be represented with the feature structures. Although it is not depicted in the figure, the feature structure for representing morpho-syntactic annotation attached to a segment should be restricted to only include MAF conformant data categories. A similar story should apply to SynAF. As proposed in (Declerck, 2006), SynAF is designed to be able to represent two syntactic properties of a human language: *constituency* and *dependency*. Therefore the syntactic annotation class should be defined to have a specialized feature structure whose node type is restricted to the categories defined in the data category sub-profiles for constituency relation or



dependency relation.

Figure 5: Configuration for MAF and SynAF.

With the ontology described so far, any linguistic expression in the proposing language service ontology can be typed according to the type of linguistic annotation it has. This type information can be effectively utilized in dynamic composition of composite services, in which checking of the input/output constraints given in the meta-description of a processing resource is necessary.

## 5 Ontology for Language Data Resources

The language resource class currently is partitioned into subclasses for **Corpus** and **Dictionary**. Among these, the most interesting class may be a class of **Dictionary**,

which will be further developed as an individual sub-ontology. The remainder of this subsection will concentrate on a sub-ontology for lexicon and dictionary. The immediate subclasses of the dictionary class are: (1) **MonolingualDictionary**, (2) **BilingualDictionary**, (3) **MultilingualTerminology**, and (4) **ConceptLexicon**.

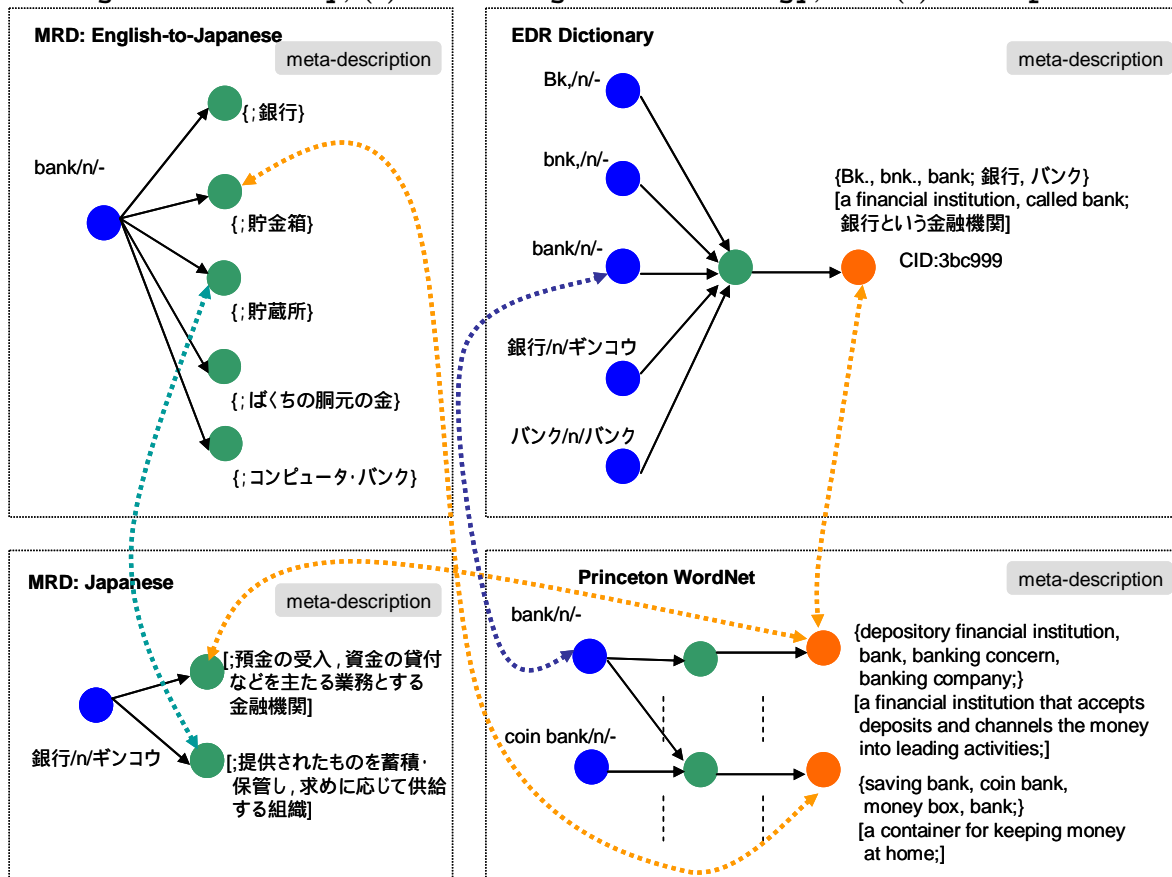


Figure 6: An example of dictionary modeling (Hayashi and Ishida, 2006).

### Dictionary Modeling (Hayashi and Ishida, 2006)

The major instances of (1) and (2) are so-called machine-readable dictionaries (MRDs). Many of the community-based special language resources should fall into (3), including multilingual terminology lists specialized for some application domains. For subclass (4), we consider the computational concept lexicons, which can be modeled by a WordNet-like encoding framework (Hayashi and Ishida, 2006). Figure 6 exemplifies the modeling framework: it presents entries from four dictionaries/lexicons, which are in some way related to the English word “bank” in the form of a graph structure. The entire model space is divided into individual dictionary spaces. Each space represents an instance from a dictionary/lexicon and is summarized by a dictionary meta-description (as shown in gray boxes). The meta-description has not been provided in detail in this paper; however, it should include information such as dictionary ID (probably given in a URI (Uniform Resource Identifier)), type (monolingual, bilingual, concept, etc.), application domain, language(s), character encoding scheme, and

inventory of lexical/semantic/conceptual relations. In addition, administrative information about the resource should also be included here. This framework has been successfully considered in defining a set of Language Grid APIs for dictionary access.

### LMF-based sub-ontology for lexicons

In the language service ontology, the sub-ontology or taxonomy of lexicons may be defined based on a service-oriented perspective, resulting in a situation where the taxonomy may not be linguistically or lexicographically motivated. However it would be far better to ground the service-oriented taxonomy to some lexicon ontology that is based on shared linguistic and lexicological principles.

We have employed LMF (Lexical Markup Framework) (Francopoulo et al., 2006) as such a framework. As known, LMF, worked out by the ISO TC37/SC4 community, is in the final stage of the international standardization process. The specification of LMF (ISO24613, 2008) states that *the ultimate goal of LMF is to create a modular structure that will facilitate true content interoperability across all aspects of electronic lexical resources*. Given this goal, the proposed modular structure of LMF consists of a core package and a number of extensions for modeling a range of lexicons including machine readable dictionaries (MRDs) and NLP lexicons. These LMF extensions are presented by extending the LMF core package, encouraging us to ontologize them by organizing the classes defined in the core package as subclasses of the top LMF class. Here *to ontologize* simply means to give a corresponding OWL representation to the constructs in the framework.

Figure 7 illustrates the ontological configuration for the LMF core model. Although the specifications of LMF are given by using UML (Unified Modeling Language) diagrams, we can convert these diagrams relatively straight forward on OWL by applying some conversion conventions; for example, we have converted the *aggregation* in UML into *hasxxx* property. As stated in the figure, we have defined the LMF core package as an independent sub-ontology; the namespace `lmfcore` prefixed to the entities indicates this situation. All the other extensions are defined in another sub-ontology which imports the LMF core ontology; the namespace `lmfall` represents the whole sub-ontology. This account is somehow different from the original LMF specification, where, managing all types of lexicon in a single ontological space is not considered. However this account gives us an opportunity to stipulate a range of lexical resources in a unique ontological space, and this is mandatory in a service-oriented language infrastructure.

Figure 8 shows a part of the LMF NLP Semantics extension, which is associated in particular with the lexical semantic notions of the extension. Note that this extension has been defined by sub-classing the classes in the LMF core package. The point is certain sub-class of the lexicon class is defined so as to have a particular type of the lexical entry. For example, in Figure 8, `lmf.Sem.Lexicon`, as a sub-class of `lmfcore:Lexicon`, is defined as having `lmf.Sem.LexicalEntry` that is, in turn, a sub-class of `lmfcore:LexicalEntry`. Again, this account is somehow different from the original LMF specification, where, for example, sub-classing of the lexicon class is not allowed.

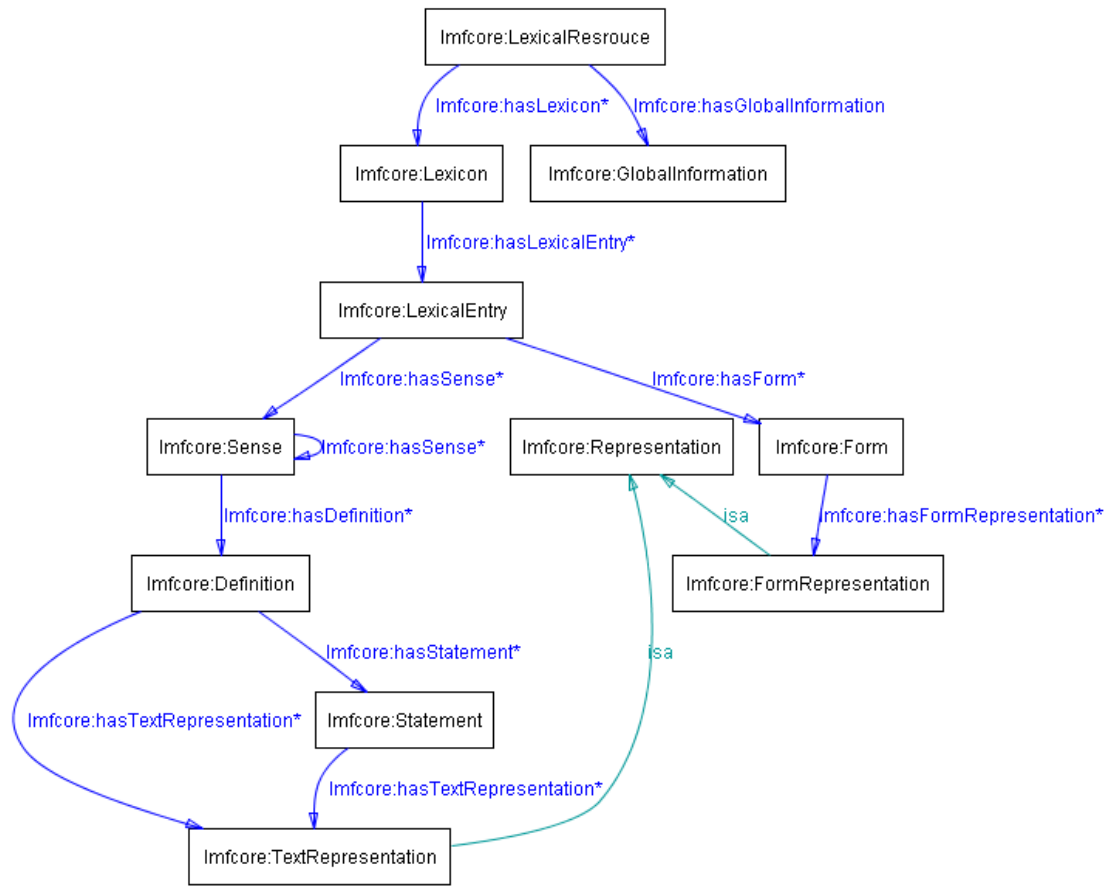


Figure 7: Configuration for LMF core package.

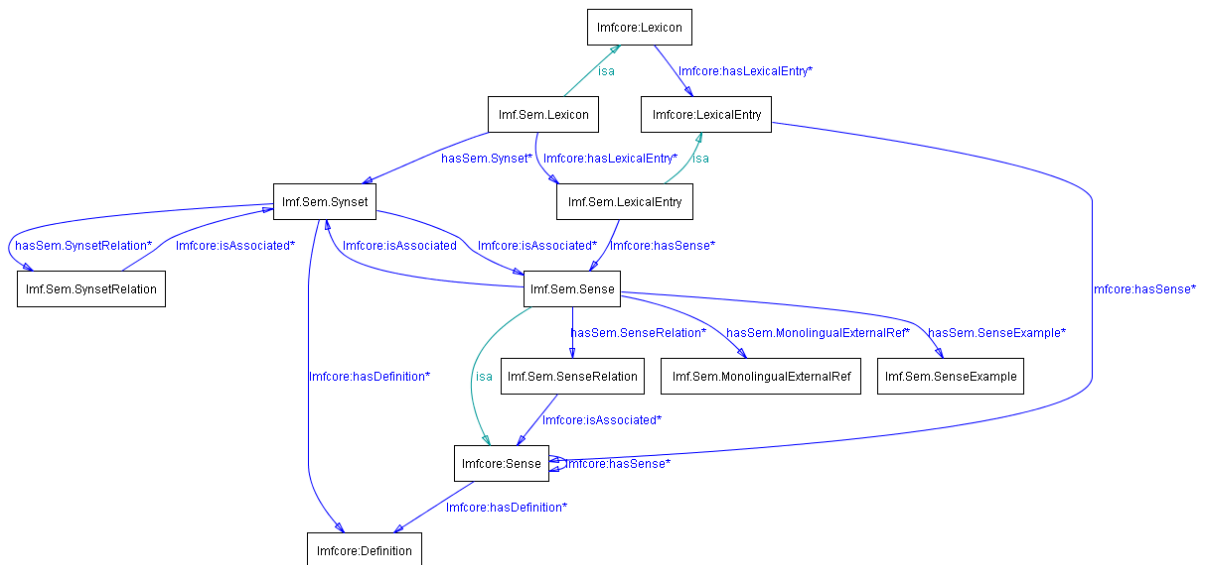


Figure 8: Configuration for LMP NLP Semantics extension.

## Service-oriented Taxonomy and its grounding

Shown in Figure 9 is an extremely simplified view of the lexicon taxonomy, which is presented just to show the notion of service-oriented lexicon taxonomy; where the top-level class `Lexicon` is first divided into `DictionaryForHumanUse` and `LexiconForNLP`. The former includes a class for so-called machine readable dictionaries (`MRD`), which is further divided into `MonolingualDictionary` and `BilingualDictionary`. The latter, on the other hand, derives a class for computational concept lexicon (`ConceptLexicon`), which has been introduced in order to stipulate WordNet-type lexical resources.

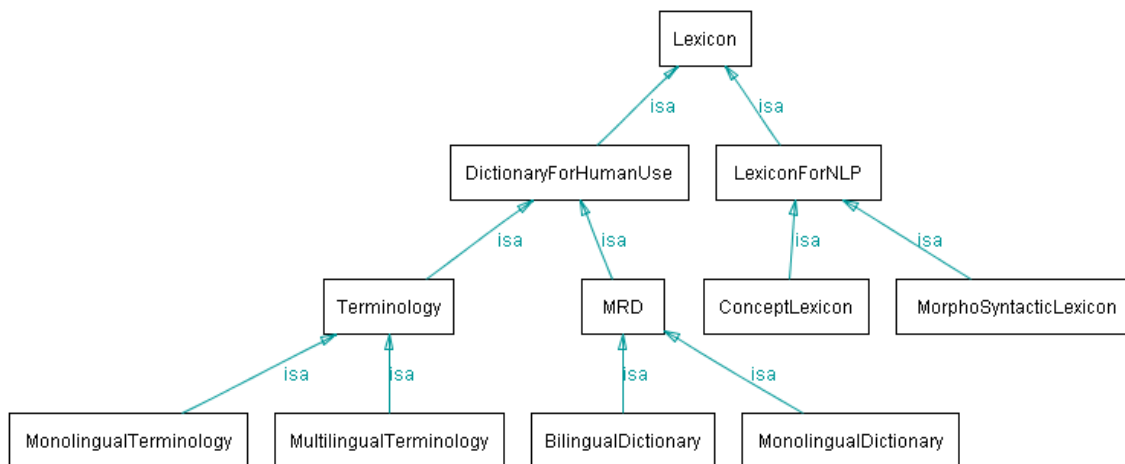


Figure 9: Service-oriented Lexicon Taxonomy (simplified).

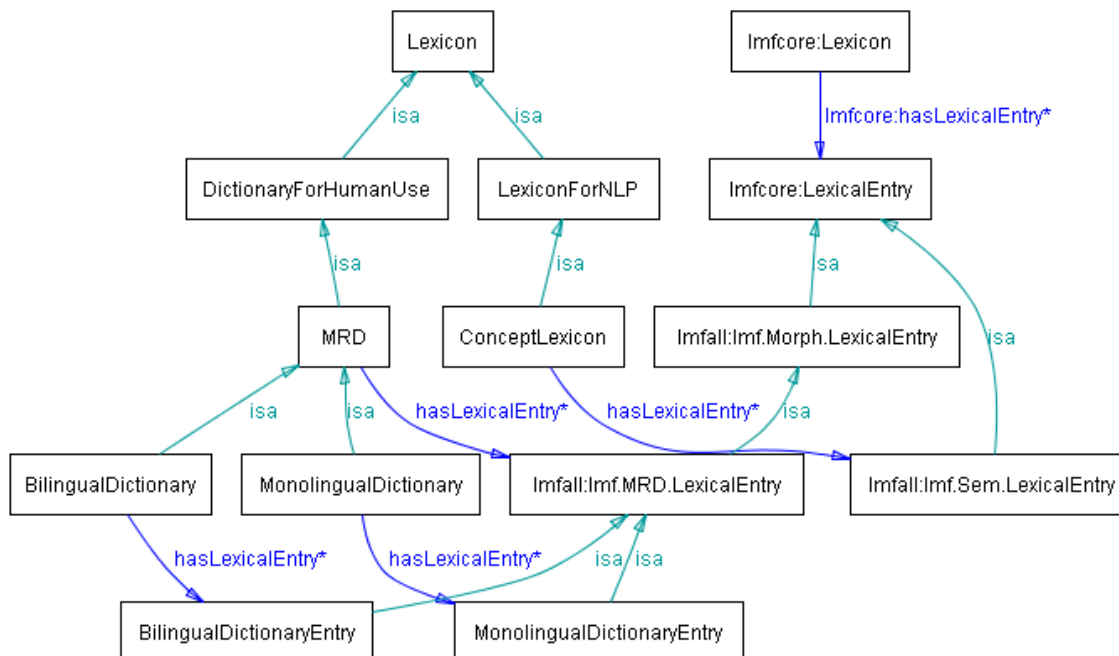


Figure 10: Grounding the service-oriented taxonomy to the ontologized LMF.

As seen in this figure, the configuration of the service-oriented lexicon taxonomy can be quite arbitrary, rather than linguistically or lexicographically motivated. However, once we have ontologized the necessary parts of the LMF, we can ground the service-oriented lexicon taxonomy to the ontologized LMF. Figure 10 depicts the basic notion of the grounding; it states that each of the classes in the service-oriented taxonomy is defined in terms of lexical entry type that they accommodate, and the lexical entry types are defined in the ontologized LMF. For example, `BilingualDictionary`, a sub-class of `MRD`, is defined by `hasLexicalEntry` property whose range is strictly restricted to `BilingualLexicalEntry`, which, in turn, is one of the descendant class of the `lmfcore:LexicalEntry` in the ontologized LMF.

When we are to represent and incorporate some new type of lexicon, we should first introduce a new lexical entry sub-class for the target lexicon in the service-oriented taxonomy, and then appropriately relate it to somewhere in the lexical entry taxonomy of the ontologized LMF.

## 6 Ontology for Language Processing Resources

### Taxonomy of language processing resources (Hayashi, 2007a; Hayashi, 2007b)

The top level of the language processing resource class mainly consists of following two subclasses as outlined in Figure 11, which take into account the input/output constraints of processing resources, as well as the language resources they utilize.

- **LR\_Accessor**: This class is introduced to describe language data resource access functionalities. It is first partitioned into `CorpusAccessor` and `LexiconAccessor`, depending on the type of language resource it accesses.
- **LinguisticProcessor**: The linguistic processor class is introduced to represent NLP tools/systems.

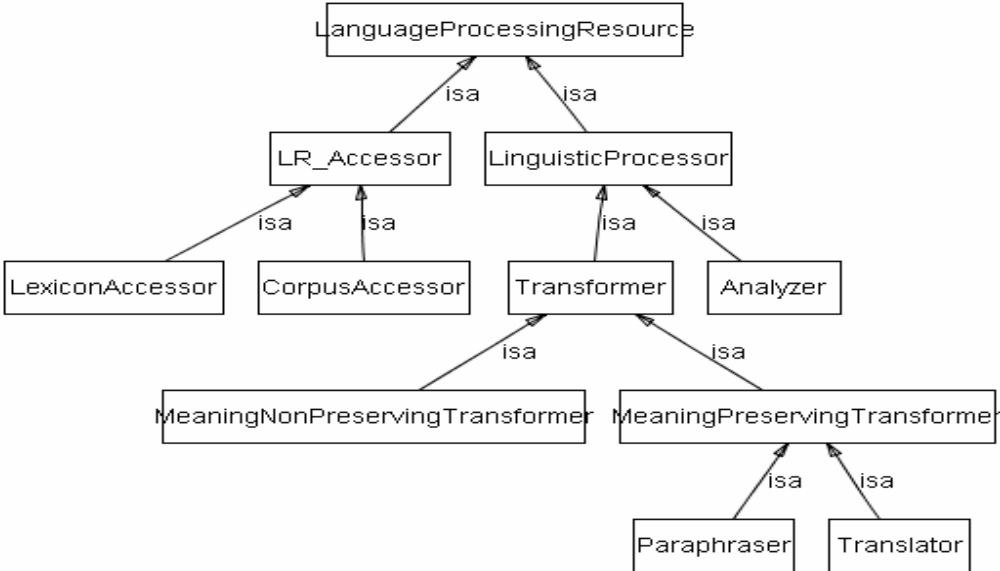


Figure 11: Taxonomy of the language processing resource.

Sub-ontologies for both `LinguisticProcessor` and `LexiconAccessor` are further discussed respectively.

### Linguistic processors

Currently and tentatively, the linguistic processor class is first partitioned into `Transformer` and `Analyzer`. `Transformer` class is introduced to represent `Paraphraser` and `Translator`: both rewrite the input linguistic expression into another expression while maintaining the original meaning. The only difference is the sameness of the input/output languages. We explicitly express the input/output language constraints in each class definition.

Figure 12 shows the working taxonomy of the `Analyzer`. While it is not depicted in the figure, the input/output constraints of a linguistic analyzer are specified by the `LinguisticExpression` class, while its precondition/effect parameters are defined by `NLProcessedStatus` class. Although the details are also not shown in this figure, these constraints are further restricted with respect to the taxonomy of the language processing resource<sup>6</sup>.

We also assume that any linguistic analyzer additively annotates some linguistic information to the input, as proposed by (Cunningham, 2002), (Klein and Potter, 2004). That is, an analyzer working at a certain linguistic level (or ‘depth’) adds the corresponding level of annotations to the input. In this sense, any natural language expression can have a layered/multiple linguistic annotation. To make this happen, a linguistic service ontology has to appropriately define a sub-ontology for the linguistic annotations as discussed in the previous sub-section.

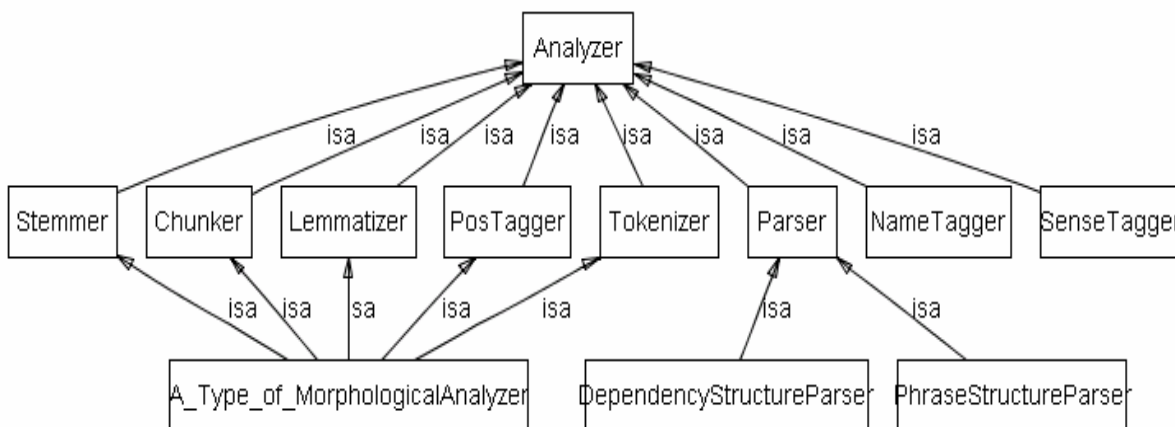


Figure 12: Taxonomy of linguistic analyzer.

Another thing that should be mentioned with the configuration of Fig 12 is that it exhibits multiply inherited hierarchy; for example, the `A_Type_of_MorphologicalAnalyzer` class inherits `Stemmer`, `Chunker`,

<sup>6</sup> Actually as shown in (Hayashi, 2007a), the taxonomy for the `NLProcessedStatus` is almost isomorphic to that of `LinguisticAnalyzer`.

**Lemmatizer**, **PosTagger** and **Tokenizer** classes, meaning that any instance of this class provides these basic NLP functionalities. This could be controversial, given such a configuration is not preferred in some senses. However, note here that, we introduced classes of basic NLP functionality in this configuration at the second-level in the hierarchy.

### Lexicon accessors (Hayashi et al. 2008b)

As already discussed, a range of lexicon access functions are grouped into **LexiconAccessor** class in the language service ontology. This class is derived from the **LR\_Accessor** class, which in turn is a sub-class of the **LanguageProcessingResource** class that can provide a language service. As a language processing resource is stipulated by the input/output data types and the language data resource which accesses to, the sub-ontology for lexicon access functions should naturally be configured with this principle.

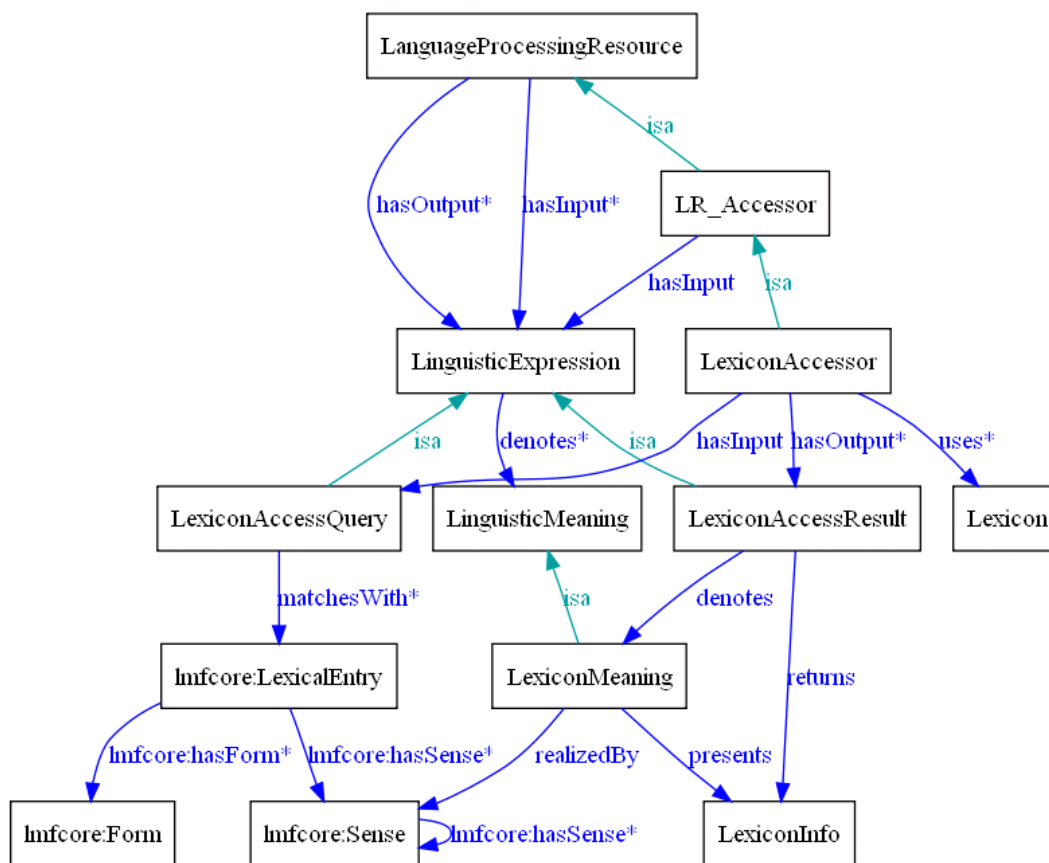


Figure 13: Configuration for lexicon accessors.

Figure 13 summarizes the ontological configuration around the lexicon accessor class; its input is restricted to **LexiconAccessQuery**, which is a sub-class of the **LinguisticExpression**; whereas the output is restricted to **LexiconAccessResult**, which is also a sub-class of the linguistic expression class.

A query to a lexicon access function, in general, consists of two types of information: matching specification and output specification. The core of the matching specification is, by necessity, a key string whose main part is a linguistic expression, typically a word, that could be further modified by some query syntax such as the regular expression. The allowed variety of key string patterns depends on the search methods that are provided by a lexicon access function. Therefore it is reasonable to sub-class the lexicon access query as a sub-class of the linguistic expression class which consists of the key string and the query conditions.

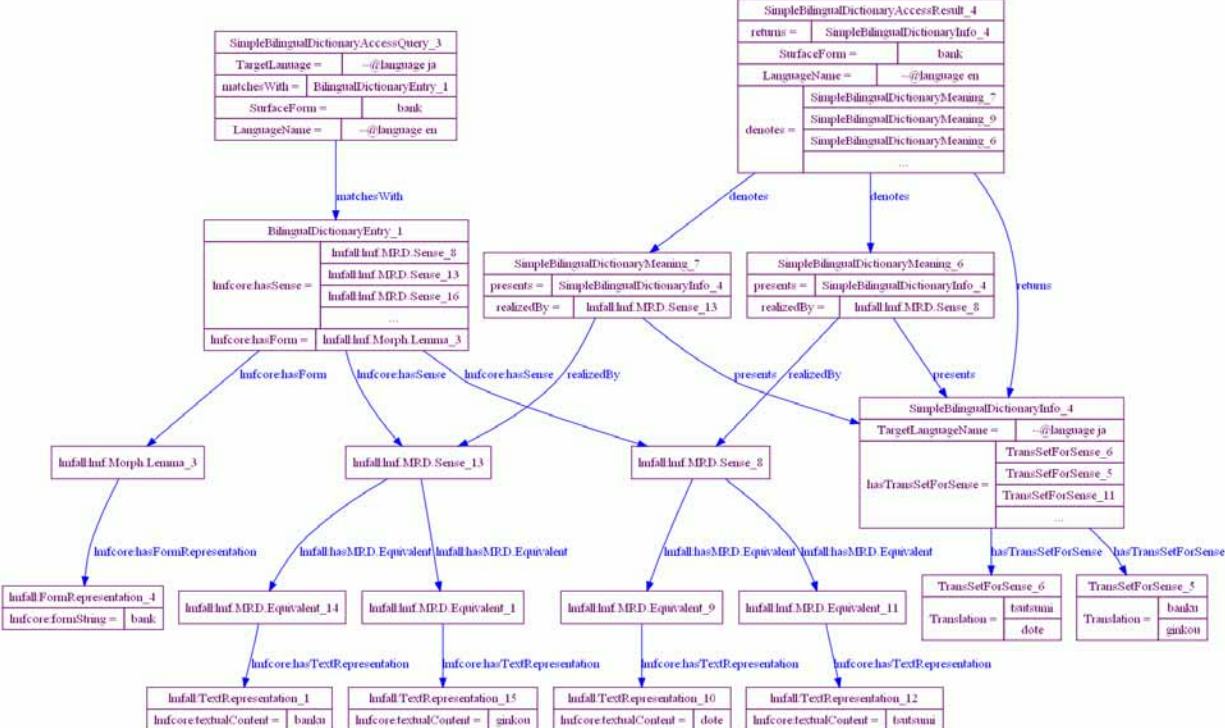


Figure 14: An instance-level example of bilingual dictionary access (partial).

The lexicon access result, on the other hand, denotes the essential information encoded in the matched lexicon entries. From the language service ontological viewpoint, the essential information encoded in a lexical entry is represented by the **LexiconMeaning** class, which is a sub-class of **LinguisticMeaning**. Notice that this account perfectly accords with LMF; in LMF, even in the MRD extension, the essential information, such as translation equivalents in a bilingual dictionary, is modeled as having a relation with the sense class, insisting that such information should capture some semantic aspects of the lexical entry. In Figure 13, notice that the linking between the lexicon meaning (peculiar in the service ontology world) and the sense part of a lexical entry (in LMF world) is adequately represented by the **realizedBy** property.

Figure 14 illustrates an instance level example of English-to-Japanese bilingual dictionary access. Here the query is an English word “bank” which has a number of translational equivalents depending on the senses. In the figure however, only two of

them are developed; financial bank sense gives Japanese equivalent 銀行(ginkou) and バンク(banku), while slope bank sense is represented by 土手 (dote) and 堤 (tsutsumi). The `matchesWith` property links an instance of lexicon access query class with an instance of lexical entry class, capturing the relationship held between the input to the lexicon access function and the corresponding lexical entries. The relationship however is only coarsely captured, and some deep constraints associated with the given query

```

.....
<wsdl:portType name="SimpleBilingualDictionary">
  <wsdl:operation name="serachSimpleBilingualDictionary"
    sawsdl:modelReference="http://langrid.nict.go.jp/Iso/Iso#SimpleBilingualDictionaryAccessor">
    <wsdl:input message="sbd:serachSimpleBilingualDictionaryRequest"/>
    <wsdl:output message="sbd:serachSimpleBilingualDictionaryResponse"/>
  </wsdl:operation>
.....
  <xsd:element name="serachSimpleBilingualDictionaryRequest"
    type="sbd:SearchQuery">
  </xsd:element>
  <xsd:element name="serachSimpleBilingualDictionaryResponse"
    type="sbd:SearchResults">
  </xsd:element>
.....
  <xsd:complexType name="SearchQuery">
    sawsdl:modelReference="http://langrid.nict.go.jp/Iso/Iso#SimpleBilingualLexiconAccessQuery">
    <xsd:sequence>
      <xsd:element name="SurfaceForm" type="xsd:string"/></xsd:element>
      <xsd:element name="LanguageName" type="xsd:language"/></xsd:element>
      <xsd:element name="TargetLanguage" type="xsd:language"/></xsd:element>
    </xsd:sequence>
  </xsd:complexType>
.....
  <xsd:complexType name="SearchResults">
    sawsdl:modelReference="http://langrid.nict.go.jp/Iso/Iso#SimpleBilingualDictionaryInfo">
    <xsd:sequence>
      <xsd:element name="Language" type="xsd:language"/></xsd:element>
      <xsd:element name="TransSetForSense" type="sbd:TransSetForSenseType"
        maxOccurs="unbounded" minOccurs="1"/></xsd:element>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="TransSetForSenseType">
    <xsd:sequence>
      <xsd:element name="Translation" type="xsd:string" maxOccurs="unbounded" minOccurs="1"/></xsd:element>
    </xsd:sequence>
  </xsd:complexType>
.....

```

specification is not explicitly encoded; it is beyond the scope of current ontologization.

Figure 15: Fragments of the SAWSDL document for SimpleBilingualDictionaryAccessor.

The ontologization of lexicon access functions depicted in Figure 13 is conceptual, and there exists a substantial gap between the conceptual picture and actual Web API specifications. To fill the gap, we try to apply a newly presented W3C recommendation SAWSDL (Semantic Annotations for WSDL) (Verma and Sheth, 2007; SAWSDL, 2007). Although some author (Yu, 2007) describes SAWSDL as a lightweight approach to Semantic Web Services, it may provide a simple yet reasonably powerful device. With the `sawsdl:modelReference` construct provided by SAWSDL, we can semantically annotate a WSDL document by making references to the concepts in the domain ontology. Figure 15 exemplifies the semantic annotations applied to a WSDL

document of the bilingual dictionary access ; whose inputs are source language, target language, key string, and the matching specification, whereas the return data is a set of translations slightly structured according to the senses.

With the current SAWSDL, we can anchor the input/output data types, as well as the service category only to some relevant concepts in the domain ontology; we cannot relate such an item to an ontological concept with some ontological path. This sometimes gives us a difficulty in appropriately specifying ontological anchors. For example, we can only specify `lmfall:TextRepresentation` class as the ontological reference to a translation equivalent; we, however in this case, need to specify that the instance of the text representation class is particularly the one linked from an instance of the `MRD.Equivalent` class. To remedy this problem, we have reluctantly introduced the `LexiconInfo` class to directly represent actual data structure returned by a lexicon access service. In Figure 9, we attached a reference to `SimpleBilingualDictionaryInfo` class for the output data whose data type is `SearchResults`.

## 7 Related Work

### Technical architecture for language infrastructures

GATE (General Architecture for Text Engineering) (Cunningham, et al., 2002) is a well-established software platform for natural language processing. It first proposed so-called processing pipeline architecture, on which several NLP components can be combined. It now has a history of more than ten years and has been used by a number of projects associated with information extraction and text mining. HoG (Heart of Gold) (Schäfer, 2004) is a similar software platform whose main concern is to integrate NLP components with different processing levels. Both of these infrastructures promote a web-based distributed working environment. However they both lack ontological foundation.

UIMA (Unstructured Information Management Architecture), first proposed by IBM, has recently attracted attentions from a wide range of audiences including NLP researchers, bioscience researchers, and so on (Kano et al., 2008), (Del Gratta et al., 2008). Similarly to GATE and HoG, UIMA provides a processing pipeline-based software platform. However a key difference of UIMA is CAS (Common Analysis System), which accommodates a container for NLP related objects such as layered annotations that are used as input/output in the processing pipeline architecture. Recently several attempts to make CAS more typed; (Bukyo and Hahn, 2008) recently proposed a sharable type system. As demonstrated by their work, UIMA now steps toward an ontology-driven language processing platform. Several attempts to integrate UIMA with above mentioned platforms have been accomplished.

### Ontology and metadata foundation for language infrastructures

Klein and Potter (2004) first sketched an ontology for NLP services with OWL-S specifications. This may be the only work that explicitly states necessity of ontological foundation for language infrastructure. Although the presented work naturally has been influenced by their proposal, it seems not active anymore. Drawbacks of their

work addressed by us are: (1) their proposal did not include ontologies for abstract linguistic objects such as linguistic annotations; (2) they also did not indicate any concern associated with related international standard for language resource interoperability.

Concerning systems for describing language resources and technologies, we mention only two of them here. LT World (Jörg and Uszkoreit, 2005) is a comprehensive knowledge portal for language technologies. One of the unique features of LT World is that it is based on a multi-dimensional ontology. For example, it classifies language technologies into such dimensions as: application, linguality, languages, technologies, linguistic area, and linguistic approach. This part of the ontology could be incorporated into our ontology especially for specifying the language processing resources. Another attempt recently made is SCHACHI (Tohyama et al. 2008) that proposes a metadata system for describing language data resources. They also operate a web site<sup>7</sup> which disseminates the metadata database whose entries have collected from around the world.

Several relevant frameworks around language data resources have been actively developed by ISO TC37/SC4. As noted in this article, we carefully observe the activities, and incorporate the results as much as possible into our language service ontology. Among these, future development of the DCR will be of importance. That is, by developing an ontology for linguistic categories on top of the basic DCR data categories, we will have an opportunity to explicitly define relations among the data categories in our language service ontology. In this regard, our approach to the ontology for linguistic categories is in some degree different from the one taken by GOLD (Farrar and Langendoen, 2003), where not only linguistic categories but complex relations among them are fundamentally defined within the central ontology.

## 8 Concluding Remarks

A *global language infrastructure* (GLI) is an open and web-based software platform on which tailored language services can be efficiently composed, disseminated and consumed. Given the increasingly realistic scenario in which language data resources and NLP tools/systems will be ubiquitous on the web, a comprehensive ontology (*language service ontology*) for describing these elements will be vital in addressing such issues in interoperability and reusability.

In this article, we have examined a triangular view of a language service, which consists of language processing, language data, and linguistic objects. Based on this definition, we have presented a top-level ontology configuration along with an essential set of sub-ontologies; these include ontologies for processing resources, language data resources, linguistic annotations, and lexicons. Among these, the ontologies for linguistic annotations and lexicons have been substantially detailed while referring to the ISO frameworks LAF, MAF, SynAF, DCR, and LMF. In doing so, we ontologized an essential part of these frameworks, and incorporated them into our comprehensive language service ontology.

---

<sup>7</sup> <http://facet.shachi.org/?ln=en>

We strongly believe that although the results presented in this article are still preliminary, the resulting language service ontology will be essential in defining an ontology-driven GLI. Obviously, we still have to provide further detail for the presented sub-ontologies by looking at concrete language data resources and NLP tools/systems for a range of human languages. In parallel, we will need to develop an approach for handling any differences in desired expressiveness inherent to the objective of a GLI; e.g., a language research infrastructure may require precise linguistic descriptions, while an infrastructure for NLP applications might demand more coarse-grained linguistic descriptions, while focusing rather on detailed communicative aspects.

To conclude, in reaching an ontology-driven GLI, we will need to establish a community of experts from a range of relevant research areas and human languages. We sincerely hope that this paper will contribute to the initiate such an initiative.

## References

- Paul Buitelaar, Thierry Declerck, Nicoletta Calzolari, and Alessandro Lenci. 2003. Language Resources and the Semantic Web. In: *Proc. of ELSNET/ENABLER workshop*.
- Ekaterina Buyuko and Udo Hahn. 2008. Fully Embedded Type System for the Semantic Annotation Layer. In: *Proc. of ICGL2008*, pp.26-33.
- Daan Broeder, et al. 2008. Foundation of a Component-based Flexible Registry for Language Resources and Technology. In: *Proc. of LREC2008*.
- Nicoletta Calzolari. (2008). Approaches towards a “Lexical Web”: the Role of Interoperability. In: *Proc. of ICGL2008*, pp.34-42.
- Lionel Clément, and Éric Villemonte de la Clergerie. 2005. MAF: a morphosyntactic annotation framework. In: *Proc. of LTC2005*.
- Hamish Cunningham, et al. 2002. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In: *Proc. of ACL 2002*, pp.168-176.
- Thierry Declerck. 2006. SynAF: Towards a Standard for Syntactic Annotation. In: *Proc. of LREC2006*, pp.229-233.
- Thierry Declerck, Nancy Ide, and Thorsten Trippel. 2008. Interoperable Language Resources. In: *Sprache und Datenverarbeitung (International Journal for Language Data Processing)*, Vol 31.1-2, pp.101-113.
- Riccardo Del Gratta, et al. 2008. UFRA: A UIMA-based Approach to Federated Language Resource Architecture. In: *Proc. of LREC2008*.
- Scott Farrar, and Terry Langendoen. 2003. A Linguistic Ontology for the Semantic Web. *Glott International*, Vol.7, pp.97-100.
- Gil Francopoulo, et al. 2006. Lexical Markup Framework (LMF). In: *Proc. of LREC2006*, pp.233-236.
- Yoshihiko Hayashi, and Toru Ishida. 2006. A Dictionary Model for Unifying Machine Readable Dictionaries and Computational Concept Lexicons. *Proc. of LREC2006*, pp.1-6.

- Yoshihiko Hayashi. 2007a. Conceptual Framework of an Upper Ontology for Describing Linguistic Services. In: Toru Ishida, Susan R. Fussell, Piek T. J. M. Vossen (Eds.): *Intercultural Collaboration*, LNCS 4568, Springer-Verlag, pp.31-45.
- Yoshihiko Hayashi. 2007b. A Linguistic Service Ontology for Language Infrastructures. In: *Proc. of ACL2007 (Demo and Poster Sessions)*, pp.145-148
- Yoshihiko Hayashi, Thierry Declerck, Paul Buitelaar, and Monica Monachini. 2008a. Ontologies for a Global Language Infrastructure. In: *Proc. of ICGL2008*, pp.105-112.
- Yoshihiko Hayashi, Chiharu Narawa, Monica Monachini, Claudia Soria, and Nicoletta Calzolari. 2008b. Ontologizing Lexicon Access Functions based on a LMF-based Lexicon Taxonomy. In: *Proc. of LREC2008*.
- Nancy Ide, and Laurent Romary. 2006. Representing Linguistic Corpora and Their Annotations. In: *Proc. of LREC2006*, pp.225-228.
- Toru Ishida. 2006. Language Grid: An Infrastructure for Intercultural Collaboration. In: *Proc. of SAINT2006*, pp.96-100.
- Toru Ishida, et al. 2008. A Non-Profit Operation Model for the Language Grid. In: *Proc. of ICGL2008*, pp.114-121.
- ISO DIS 24613:2008. 2008. Language resource management - Lexical markup framework (LMF), Rev.16.
- Brigitte Jörg, and Hans Uszkoreit. 2005. The Ontology-based Architecture of LT World, a Comprehensive Web Information System for a Science and Technology Discipline. In: *Leitbild Informationskompetenz: Positionen - Praxis - Perspektiven im europäischen Wissensmarkt*.
- Yoshinobu Kano, et al. 2008. Sharable Type System Design for Tool Interoperability and Combinatorial Comparison. In: *Proc. of ICGL2008*, pp.122-129.
- Evan Klein, and Stephen Potter. 2004. An Ontology for NLP Services. In: *Proc. of LREC Workshop on a Registry of Linguistic Data Categories within an Integrated Language Resource Repository Area*.
- Deborah L. McGuinness, and Frank van Harmelen. 2004. OWL Web Ontology Language Overview. <http://www.w3.org/TR/owl-features/>
- Yohei Murakami, et al. 2006. Infrastructure for Language Service Composition. In: *Proc. of Second International Conference on Semantics, Knowledge, Grid*.
- Ulrich Schäfer. 2004. Middleware for Creating and Combining Multi-dimensional NLP Markup. In: *Proc. of the EAACL2006 Workshop on Multi-dimensional Markup in NLP*.
- Hitomi Tohyama, et al. 2008. SHACHI: A Large Scale Metadata Database of Language Resources. In: *Proc. of ICGL2008*, pp.205-212.
- Sue Ellen Wright. 2004. A Global Data Category Registry for Interoperable Language Resources. In: *Proc. of LREC2004*, pp.123-126.
- SWSDL 2.0. 2007. Semantic Annotations for WSDL and XML Schema. <http://www.w3.org/TR/sawSDL/>
- Kunal Verma, Amit Sheth. (2007). Semantically Annotating a Web Service. *IEEE Internet Computing*, Vol.11, No.2, pp.83-85.

WSDL 2.0. 2007. Web Services Description Language (WSDL) Version 2.0.  
<http://www.w3.org/TR/wsdl20>

Liyang Yu. 2007. *Introduction to the Semantic Web and Semantic Web Services*. Chapman & Hall/CRC, 341 pages.