The Language Application Grid Web Service Exchange Vocabulary

Nancy Ide[1(✉)], Keith Suderman[1], Marc Verhagen[2], and James Pustejovsky[2]

[1] Vassar College, Poughkeepsie, NY, USA
ide@cs.vassar.edu, suderman@cs.vassar.edu
[2] Brandeis University, Waltham, MA, USA
marc@cs.brandeis.edu, jamesp@cs.brandeis.edu

Abstract. In the context of the Linguistic Applications (LAPPS) Grid project, we have undertaken the definition of a Web Service Exchange Vocabulary (WS-EV) specifying a terminology for a core of linguistic objects and properties exchanged among NLP tools that consume and produce linguistically annotated data. The goal is not to define a new set of terms, but rather to provide a single web location where terms relevant for exchange among NLP tools are defined and provide a "sameAs" link to all known web-based definitions that correspond to them. The WS-EV is intended to be used by a federation of six grids currently being formed but is usable by any web service platform.

Keywords: Linguistic standards · Interoperability · Web services · Service grids

1 Introduction

There is clearly a demand within the community for some sort of standard for exchanging annotated language data among tools.[1] This has become particularly urgent with the emergence of web services, which has enabled the availability of language processing tools that can and should interact with one another, in particular, by forming pipelines that can branch off in multiple directions to accomplish application-specific processing. While some progress has been made toward enabling *syntactic interoperability* via the development of standard representation formats (e.g., ISO LAF/GrAF [11,13], NLP Interchange Format (NIF) [7], UIMA[2] Common Analysis System (CAS)) which, if not identical, can be trivially mapped to one another, *semantic interoperability* among NLP tools remains problematic [8]. A few efforts to create repositories, type systems, and ontologies of linguistic terms (e.g., ISOCat[3], OLiA[4], various repositories for UIMA

[1] See, for example, proceedings of the recent LREC workshop on "Language Technology Service Platforms: Synergies, Standards, Sharing" (http://www.ilc.cnr.it/ltsp2014/).
[2] https://uima.apache.org/.
[3] http://www.isocat.org.
[4] http://nachhalt.sfb632.uni-potsdam.de/owl/.

Y. Murakami and D. Lin (Eds.): WLSI 2015, LNAI 9442, pp. 18–32, 2016.
DOI: 10.1007/978-3-319-31468-6_2

type systems[5], GOLD[6], NIF Core Ontology[7]) have been undertaken to enable (or provide) a mapping among linguistic terms, but none has yet proven to include all requisite terms and relations or be easy to use and reference. General repositories such as Dublin Core[8], schema.org, and the Friend of a Friend project[9] include some relevant terms, but they are obviously not designed to cover all the kinds of information found in linguistically annotated data. There have been recent efforts to address semantic interoperability among NLP web services [15,16], but the solutions deal only with meta-data and high-level elements (e.g., text); more precise specification of information types are deliberately left underspecified and left to the service provider to determine.

In the context of the Linguistic Applications (LAPPS) Grid project [10], we have undertaken the definition of a Web Service Exchange Vocabulary (WS-EV) specifying a terminology for a core of linguistic objects and properties exchanged among NLP web services that consume and produce linguistically annotated data. The work is being done in collaboration with ISO TC37 SC4 WG1 in order to ensure full community engagement and input. The goal is not to define a new set of terms, but rather to provide a single web location where terms relevant for exchange among NLP tools are defined and provide a "sameAs" link to all known web-based definitions that correspond to them. A second goal is to define relations among the terms that can be used when linguistic data are exchanged. The WS-EV is intended to be used by a newly-formed federation of grids, including the Kyoto Language Grid[10], the Language Grid Jakarta Operation Center[11], the Xinjiang Language Grid, the Language Grid Bangkok Operation Center[12], LinguaGrid[13], MetaNet/MetaShare[14], and LAPPS, but is usable by any web service platform.

This paper describes the LAPPS WS-EV, which is currently under construction. We first describe the LAPPS project and then overview the motivations and principles for developing the WS-EV. We then describe its use in the JSON-LD LAPPS Interchange Format (LIF) to enable semantic interoperability among web services in the LAPPS Grid.

2 The Language Application Grid Project

The Language Application (LAPPS) Grid project establishing a framework that enables language service discovery, composition, and reuse, in order to promote

[5] E.g., http://www.julielab.de/Resources/Software/UIMA+type+system-p-91.html.
[6] http://linguistics-ontology.org.
[7] http://persistence.uni-leipzig.org/nlp2rdf/ontologies/nif-core/nif-core.
[8] http://dublincore.org.
[9] http://www.foaf-project.org.
[10] http://langrid.org.
[11] http://langrid.portal.cs.ui.ac.id/langrid/.
[12] http://langrid.servicegrid-bangkok.org.
[13] http://www.linguagrid.org/.
[14] http://www.meta-share.eu.

sustainability, manageability, usability, and interoperability of natural language Processing (NLP) components. It is based on the service-oriented architecture (SOA), a more recent, web-oriented version of the pipeline architecture that has long been used in NLP for sequencing loosely-coupled linguistic analyses. The LAPPS Grid provides a critical missing layer of functionality for NLP: although existing frameworks such as UIMA and GATE provide the capability to wrap, integrate, and deploy language services, they do not provide general support for service discovery, composition, and reuse.

The LAPPS Grid is a collaborative effort among US partners Brandeis University, Vassar College, Carnegie-Mellon University, and the Linguistic Data Consortium at the University of Pennsylvania, and is funded by the US National Science Foundation (NSF). The project builds on the foundation laid in the NSF-funded project SILT [9], which established a set of needs for interoperability and developed standards and best practice guidelines to implement them. LAPPS is similar in its scope and goals to ongoing projects such as The Language Grid [12], PANACEA[15], LinguaGrid[16], and CLARIN[17], which also provide web service access to basic NLP processing tools and resources and enable pipelining these tools to create custom NLP applications and composite services such as question answering and machine translation, as well as access to language resources such as mono- and multi-lingual corpora and lexicons that support NLP. The transformative aspect of the LAPPS Grid is therefore not the provision of a suite of web services, but rather that it orchestrates access to and deployment of language resources and processing functions available from servers around the globe, and enables users to easily add their own language resources, services, and even service grids to satisfy their particular needs. The specific goals of the LAPPS project are to: (1) design, develop, and promote a Language Application Grid (LAPPS Grid) based on Service Grid Software to support the development and deployment of integrated natural language applications and enable federation of grids and services throughout the world; (2) provide an open advancement (OA) framework (Ferrucci et al., 2009a) for component- and application-based evaluation; (3) provide access to language resources for members of the NLP community as well as researchers in a wide range of social science and humanities disciplines; (4) enable easy navigation through licensing issues; and (5) actively promote adoption, use, and community involvement with the LAPPS Grid.

One of the most unique innovations in the LAPPS Grid is the provision of an open advancement (OA) framework (Ferrucci et al., 2009a) for component- and application-based evaluation of NLP tools and pipelines. The availability of this type of evaluation service will provide an unprecedented tool for NLP development that could, in itself, take the field to a new level of productivity. OA involves evaluating *multiple possible solutions* to a problem, consisting of different configurations of component tools, resources, and evaluation data, to find the

[15] http://panacea-lr.eu/.
[16] http://www.linguagrid.org/.
[17] http://www.clarin.eu/.

optimal solution among them, and enabling rapid identification of frequent error categories, together with an indication of which module(s) and error type(s) have the greatest impact on overall performance. On this basis, enhancements and/or modifications can be introduced with an eye toward achieving the largest possible reduction in error rate [2,20]. OA was used in the development of IBM's Watson to achieve steady performance gains over the four years of its development [3]; more recently, the open-source OAQA project has released software frameworks which provide general support for open advancement [5,20], which has been used to rapidly develop information retrieval and question answering systems for bioinformatics [14,20].

The fundamental system architecture of the LAPPS Grid is based on the Open Service Grid Initiative's Service Grid Server Software[18] developed by the National Institute of Information and Communications Technology (NICT) in Japan and used to implement Kyoto University's Language Grid, a service grid that supports multilingual communication and collaboration. Like the Language Grid, the LAPPS Grid provides three main functions: language service registration and deployment, language service search, and language service composition and execution. The LAPPS Grid project has adopted Galaxy [6] as a workflow engine, which provides a graphical interface where data inputs and computational steps are selected from dynamic menus, and results are displayed in plots and summaries that encourage interactive workflows and the exploration of hypotheses. The LAPPS Grid produces relevant component-level measures for standard metrics, given gold-standard test data, for each component in a pipeline, which facilitates error analysis. In addition, the Grid automatically generates metrics measurements plus variance and statistical significance calculations for each possible pipeline, using a service-oriented version of the Configuration Space Exploration (CSE) algorithm [20]. The LAPPS Grid also implements a dynamic licensing system for handling license agreements on the fly[19], provides the option to run services locally with high-security technology to protect sensitive information where required, and enables access to grids other than those based on the Service Grid technology.

We have adopted the JSON-based serialization for Linked Data (JSON-LD) to represent linguistically annotated data for the purposes of web service exchange. The JavaScript Object Notation (JSON) is a lightweight, text-based, language-independent data interchange format that defines a small set of formatting rules for the portable representation of structured data. Because it is based on the W3C Resource Definition Framework (RDF), JSON-LD is trivially mappable to and from other graph-based formats such as ISO LAF/GrAF and UIMA CAS, as well as a growing number of formats implementing the same data model. Most importantly, JSON-LD enables services to reference categories and definitions in web-based repositories and ontologies or any suitably defined concept at a given URI. JSON-LD provides *syntactic interoperability* among services

[18] http://servicegrid.net.
[19] See [1] for a description of how licensing issues are handled in the LAPPS Grid.

in the LAPPS Grid, while *semantic interoperability* is provided by the LAPPS Web Service Exchange Vocabulary, described in the next section.

3 LAPPS Web Service Exchange Vocabulary

3.1 Motivation

The WS-EV addresses a relatively small but critical piece of the overall LAPPS architecture: it allows web services to communicate about the content they deliver, such that the *meaning*–i.e., exactly what to do with and/or how to process the data–is understood by the receiver. As such it performs the same function as a UIMA type system performs for tools in a UIMA pipeline that utilize that type system, or the common annotation labels (e.g.,"Token","Sentence", etc.) required for communication among pipelined tools in GATE. These mechanisms provide semantic interoperability among tools as long as one remains in either the UIMA or GATE world. To pipeline a tool whose output follows GATE conventions with a tool that expects input that complies with a given UIMA type system, some mapping of terms and structures is likely to be required.[20] This is what the WS-EV is intended to enable; effectively, it is a *meta-type-system* for mapping labels assigned to linguistically annotated data so that they are understood and treated consistently by tools that exchange them in the course of executing a pipeline or workflow. Since web services included in LAPPS and federated grids may use any i/o semantic conventions, the WS-EV allows for communication among any of them–including, for example, between GATE and UIMA services[21].

The ability to pipeline components from diverse sources is critical to the implementation of the OA development approach described in the previous section, it must be possible for the developer to "plug and play" individual tools, modules, and resources in order to rapidly re-configure and evaluate new pipelines. These components may exist on any server across the globe, consist of modules developed within frameworks such as UIMA and GATE, and/or be user-defined services existing on a local machine.

3.2 WS-EV Design

The WS-EV was built around the following design principles, which were compiled based on input from the community:

1. The WS-EV will not reinvent the wheel. Objects and properties defined in the WS-EV will be linked to definitions in existing repositories and ontologies wherever possible.

[20] Within UIMA, the output of tools conforming to different type systems may themselves require conversion in order to be used together.

[21] Figure 6 shows a pipeline in which both GATE and UIMA services are called; GATE-to-GATE and UIMA-to-UIMA communication does not use the WS-EV, but it is used for communication between GATE and UIMA services, as well as other services.

2. The WS-EV will be designed so as to allow for easy, one-to-one mapping from terms designating linguistic objects and properties commonly produced and consumed by NLP tools that are wrapped as web services. It is not necessary for the mapping to be object-to-object or property-to-property[22]
3. The WS-EV will provide a *core* set of objects and properties, on the principle that "simpler is better", and provide for (principled) definition of additional objects and properties beyond the core to represent more specialized tool input and output.
4. The WS-EV is not LAPPS-specific; it will not be governed by the processing requirements or preferences of particular tools, systems, or frameworks.
5. The WS-EV is intended to be used *only* for interchange among web services performing NLP tasks. As such it can serve as a "pivot" format to which user and tool-specific formats can be mapped.
6. The web service provider is responsible for providing wrappers that perform the mapping from internally-used formats to and/or from the WS-EV.
7. The WS-EV format should be compact to facilitate the transfer of large datasets.
8. The WS-EV format will be chosen to take advantage, to the extent possible, of existing technological infrastructures and standards.

As noted in the first principle, where possible the objects and properties in the WS-EV are drawn from existing repositories such as ISOCat and the NIF Core Ontology and linked to them using the taxonomy of relation types defined in RELcat [19], which accommodates multiple vocabularies for relation predicates, including those from the Web Ontology Language (OWL) [17] and the Simple Knowledge Organization System (SKOS) [18], as shown in Fig. 1.

However, many repositories do not include some categories and objects relevant for web service exchange (e.g., "token" and other segment descriptors), do include multiple (often very similar) definitions for the same concept, and/or do not specify relations among terms. We therefore attempted to identify a set of (more or less) "universal" concepts by surveying existing type systems and schemas–for example, the Julie Lab and DARPA GALE UIMA type systems and the GATE schemas for linguistic phenomena–together with the I/O requirements of commonly used NLP software (e.g., the Stanford NLP tools, OpenNLP, etc.). Results of the survey for token and sentence identification and part-of-speech labeling[23] showed that even for these basic categories, there exists no"standard" set of categories and relations.

[22] We follow the terminology used in RDF/OWL and JSON-LD: the term "objects" (in RDF, nodes in the Semantic Web graph) refers to common linguistic labels or types, and "properties" denote what are often referred to as "features" or "attributes" of an object (in RDF, these are labels of edges between object nodes). We emphasize that our assignment of linguistic labels as objects and properties, while principled to the extent possible, is otherwise arbitrary and may therefore differ from existing type systems and schemas. This does not, however, impede mapping to object and properties in the WS-EV.

[23] Available at http://www.anc.org/LAPPS/EP/Meeting-2013-09-26-Pisa/ep-draft.pdf.

```
1. Related (rel:related)
1.1. Sameas(rel:sameAs)
1.2. Almost same as (rel:almostSameAs)
1.3. Broader than(rel:broaderThan)
1.3.1. Superclass of (rel:superClassOf)
1.3.2. Has part (rel:hasPart)
1.3.2.1. Has direct part (rel:hasDirectPart)
1.4 Narrower than (del:narrowerThan)
1.4.1. Sub class of (rel:subClassOf)
1.4.2. Part of (rel:partOf)
1.4.2.1. Direct part of (rel:directPartOf)
```

Fig. 1. Relation types in RELCat

Perhaps more problematically, sources that do specify relations among concepts, such as the various UIMA type systems and GATE's schemas, vary widely in their choices of what is an object and what is a property; for example, some treat "token" as an object (label) and "lemma" and "pos" as associated properties (features), while others regard "lemma" and/or "pos" as objects in their own right. Decisions concerning what is an object and what is a property are for the most part arbitrary; no one scheme is right or wrong, but a consistent organization is required for effective web service interchange. The WS-EV therefore defines an organization of objects and properties solely for the purposes of communication among web services in the LAPPS Grid. It is irrelevant if a given scheme treats, say, "pos" as an object or type in its own right, as long as it is mapped to the correspondingly defined WS-EV object or property for the purposes of web service exchange.

In addition, the WS-EV is intended to provide a *core* set of terms, augmented as needed when services are added to the LAPPS Grid, but it is by no means intended to be comprehensive. The WS-EV includes *sameAs* and *similarTo* mappings that link to like concepts in other repositories where possible, thus serving primarily to group the terms and impose a structure of relations required for web service exchange in one web-based location.

In addition to the principles above, the WS-EV is built on the principle of orthogonal design, such that there is one and only one definition for each concept. It is also designed to be very lightweight and easy to find and reference on the web. To that end we have established a straightforward web site (the Web Service Exchange Vocabulary Repository[24]), similar to *schema.org*, in order to provide web-addressable terms and definitions for reference from annotations exchanged among web services. Our approach is bottom-up: we have adopted a minimalist strategy of adding objects and properties to the repository only as they are needed as services are added to the LAPPS Grid. Terms are organized in a shallow hierarchy, with inheritance of properties, as shown in Fig. 2.

[24] http://vocab.lappsgrid.org.

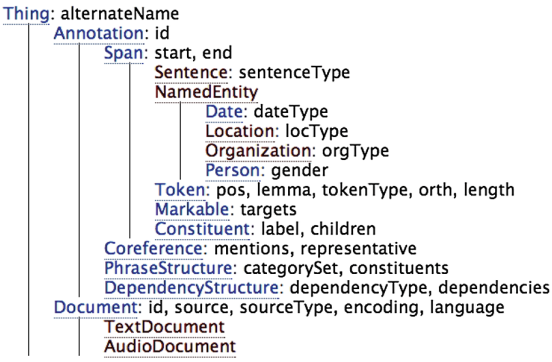

Fig. 2. Fragment of the WS-EV type hierarchy (associated properties in gray)

Note that the WS-EV does not provide a repository of specific categories for part-of-speech or syntactic and semantic roles; rather, a specific label may be referenced in the JSON-LD representation using a URI for one of the several locations where such information resides on the web. Alternatively, a string providing the information may be used (see, for example, the JSON-LD sample in Sect. 4.1). Metadata specifying the tags and/or software that produced a given labeling can be checked to ensure that the labels required by a consumer service conform to those generated by the provider.

4 WS-EV and JSON-LD

We have defined the *LAPPS Interchange Format (LIF)*[25] using JSON-LD for interchange among LAPPS Grid web services. References in LIF point to URIs providing definitions for specific linguistic categories in the WS-EV. They may also reference documentation for processing software and rules for processes such as tokenization, entity recognition, etc. used to produce a set of annotations, which are often left unspecified in annotated resources thus inhibiting reproducibility of results (see for example [4]). While not required for web service exchange in the LAPPS Grid, the inclusion of such references can contribute to the better replication and evaluation of results in the field. Figure 4 shows the information for *Token*, which defines the concept, identifies application types that produce objects of this type, cross-references a similar concept in ISOCat, and provides the URI for use in the LIF representation. It also specifies the common properties that can be specified for a set of Token objects, and the individual properties that can be associated with a Token object. There is no requirement to use any or all of the properties in the LIF representation, and we foresee that many web services will require definition of objects and properties not included in the WS-EV or elsewhere. We therefore provide mechanisms for (principled)

[25] For a full description of LIF, see Verhagen, *et al.*, "The LAPPS Interchange Format", elsewhere in this volume.

definition of objects and properties beyond the WS-EV. Two options exist: users can provide a URI where a new term or other documentation is defined, or users may add a definition to the WS-EV. In the latter case, service providers use the name space automatically assigned to them at the time of registration, thereby avoiding name clashes and providing a distinction between general categories used across services and more idiosyncratic categories.

```
"@context" : "http://vocab.lappsgrid.org/",
"metadata" : { },
"text" : {
    "@value" : "Some of the strongest critics of our welfare system..."
 }
"views" : [ {
    "metadata" : {
          "contains" : {
               "Token" : {
                    "producer" : "org.anc.lapps.stanford.SATokenizer:1.4.0",
                    "type" : "tokenization:stanford"
                }
           }
      }
 } ],
 "annotations" : [ {
     "@type" : "Token",
     "id" : "tok0",
     "start" : 18,
     "end" : 22
 } ],
    . . .
```

Fig. 3. JSON-LD fragment referencing the LAPPS Grid WS-EV

Figure 3 shows a fragment of the LIF representation that references terms in the WS-EV. The *context* statement at the top identifies the URI that is to be prefixed to any unknown name in order to identify the location of its definition. For the purposes of the example, the text to be processed is given inline. Our current implementation includes results from each step in a pipeline, where applicable, together with metadata describing the service applied in each step (here, org.anc.lapps.stanford.SATokenizer:1.4.0) and identified by an internally-defined type (tokenization:stanford). The annotations include references to the objects defined in the WS-EV, in this example, *Token* (defined at http://vocab.lappsgrid.org/Token), with (inherited) properties *id, start*, and *end* defined at http://vocab.lappsgrid.org/Token#id, http://vocab.lappsgrid.org/Token#start and http://vocab.lappsgrid.org/Token#end. The web page defining these terms is shown in Fig. 4.

Thing > Annotation > Span > Token

Definition	A string of one or more characters that serves as an indivisible unit for the purposes of morpho-syntactic labeling (part of speech tagging).
Similar to	http://www.isocat.org/datcat/DC-1403
URI	http//vocab.lappsgrid.org/Token

Metadata

Metadata from Annotation

Properties	Type	Description
producer	List of URI	The software that produced the annotations.
rules	List of URI	The documentation (if any) for the rules that were used to identify the annotations.

Properties

Properties	Type	Description
pos	String or URI	Part-of-speech tag associated with the token.
lemma	String or URI	The root (base) form associated with the token. URI may point to a lexicon entry.
tokenType	String or URI	Sub-type such as word, punctuation, abbreviation, number, symbol, etc. Ideally a URI referencing a pre-defined descriptor.
orth	String or URI	Orthographic properties of the token such as LowerCase, UpperCase, UpperInitial, etc. Ideally a URI referencing a pre-defined descriptor.
length	Integer	The length of the token

Properties from Span

Properties	Type	Description
start	Integer	The starting offset (0-based) in the primary data.
end	Integer	The ending offset (0-based) in the primary data.

Properties from Annotation

Properties	Type	Description
id	String	A unique identifier associated with the annotation.

Properties from Thing

Properties	Type	Description
alternateName	String	An alias for the item.

Fig. 4. Token definition

4.1 Mapping to JSON-LD

As noted above in Sect. 1, existing schemes and systems for organizing linguistic information exchanged by NLP tools vary considerably. Figure 5 shows some variants for a few commonly used NLP tools, which differ in terminology, structure, and physical format. To be used in the LAPPS Grid, tools such as those in the list are wrapped so that their output is in JSON-LD format, which provides syntactic interoperability, terms are mapped to corresponding objects in the WS-EV, and the object-feature relations reflect those defined in the WS-EV. Correspondingly, wrappers transduce the LIF representation to the format used internally by the tool on input. This way, the tools use their internal format as usual and map to LIF for exchange only.

For example, the Stanford POS tagger XML output format produces output like this:

```
<word id="0" pos="VB">Let</word>
```

Name	Input	Form	Output	Form	Example
Stanford tagger	pt	n/a	word_pos	opl	box_NN1
	XML	n/a	XML	inline	<word id="0" pos="VB">Let</word>
NaCTeM tagger	pt	n/a	word/pos	inline	box/NN1
CLAWS (1)	pt	n/a	word_pos	inline	box_NN1
CLAWS (2)	pt	n/a	XML	inline	<w id="2" pos="NN1">Type</w>
CST Copenhagen	pt	n/a	word/pos	inline	box/NN1
TreeTagger	pt?	n/a	word pos lem	opl	The DT the
TnT	token	opl	word pos	opl	der ART
			word (pos pr)+	opl	Falkenstein NE 8.00 NN 1.99
Twitter NLP	pt	opl	word pos conf	opl	smh G 0.9406
NLTK	pt	s, bls	[('word', 'pos')]	inline	[('At', 'IN'), ('eight', 'CD'),]
OpenNLP splitter	pt	n/a	sentences	ospl	I can't tell you if he's here.
OpenNLP tokenizer	sent	ospl	tokens	wss, ospl	I can 't tell you if he 's here .
OpenNLP tagger	token	wss, ospl	word_pos	ospl	At_IN eight_CD o'clock_JJ on_IN

pt = plain text
opl = one per line
wss = white space separated
ospl = one sentence per line
bps = blank line separated

Fig. 5. I/O variants for common splitters, tokenizers, and POS taggers

This maps to the following LIF representation:

```
{
    "@type" : "Token",
     "id" : "0",
     "start" : 18,
     "end" : 21,
     "features" : {
         "pos"    : "VB"
     }
}
```

The Stanford representation uses the term "word" as an XML element name, gives an id and pos as attribute-value pairs, and includes the string being annotated as element content. For conversion to JSON-LD/WS-EV, "word" is mapped to "Token", and the attributes *id* and *pos* map to properties of Token with the same names. Because the LIF representation uses standoff annotation, the properties *start* and *end* are included in order to provide the offset location of the string in the primary data.

Services that share a format other than JSON-LD need not map into and out of LIF when pipelined in the LAPPS Grid. For example, two GATE services would exchange GATE XML documents, and two UIMA services would exchange UIMA CAS, as usual. This avoids unnecessary conversion and at the same time allows including services consisting of individual tools or entire composite workflows from other frameworks. Figure 6 gives an example of the logical flow in the LAPPS Grid, showing conversions into and out of LIF where needed.

Each service in the LAPPS Grid is required to provide metadata that specifies what kind of input is required and what kind of output is produced. For example, any service as depicted in the flow diagram in Fig. 6 can require input with specific content (tokens, sentences, etc.), reduced according to certain specifications (stanford-style tokenization, penn pos tags, etc.), and in a particular format (gate-document, uima-cas, LIF).

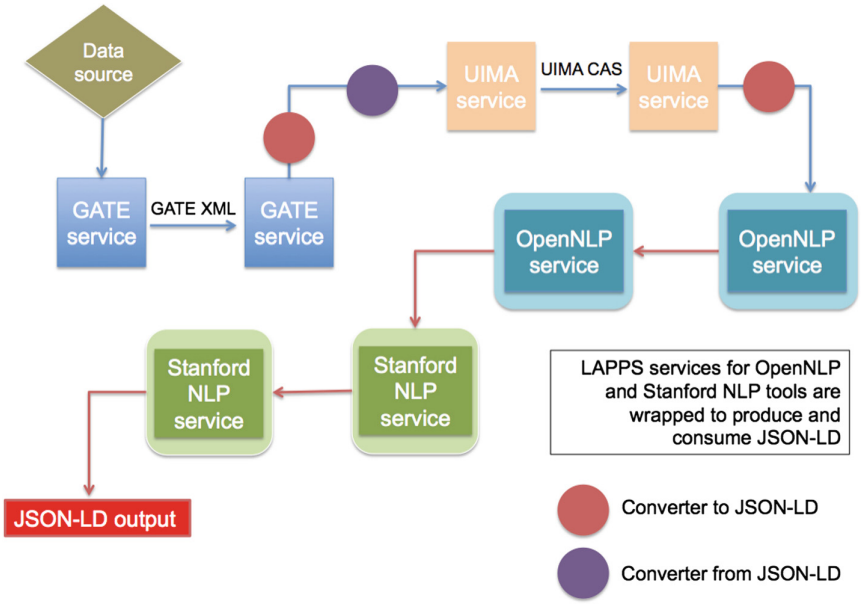

Fig. 6. Logical flow through the LAPPS Grid (client-server communication not represented) (Color figure online)

We have created the WS-EV to provide a basic, common terminology that can handle the basic types that are exchanged among LAPPS Grid services, regardless of the internal representations they use, with the intention that where possible, commonly used linguistic types (whatever their names, and whether they are objects or properties in the original scheme) are mapped to terms in the WS-EV. However, services may provide their own definitions for any object or property, or use names other than those in the WS-EV. This is achieved by using the optional JSON-LD @CONTEXT key to reference a set of user-defined context elements or redefine the names that refer to WS-EV terms. For example, in the fragment below, a service provides an alternative definition for *Token* by associating it with a different URI (where, presumably, an alternative definition is provided). It also renames the properties "start" and "end" to "startOffset" and "endOffset", by associating these names with the URIs for the former names in the WS-EV:

```
{
 "@context": {
         "Token":"http:/www/example.com/MyToken",
          "startOffset":"http://vocab.lappsgrid.org/Token#start",
          "endOffset":"http://vocab.lappsgrid.org/Token#end",
           },
 "annotations": [
    {"@type":"Token","id":"t0","startOffset": 0,"endOffset": 5}
  ]
}
```

The @CONTEXT key can also be used to provide alternative definitions for linguistic objects and properties when mappings are not one-to-one.

Note that the *producer* field in the LIF representation provides the name of the process or program that produced the object (see Fig. 3). So, for example, the producer associated with a set of *Token* objects (e.g., the Stanford Tokenizer) can be checked by the consuming service to ensure they are produced according to specific tokenization rules.

Properties associated with objects are not required (with the exception of properties such as "id", "start", and "end"). So, for example, "pos" (part-of-speech) is specified as a property of *Token*, but would be omitted if no part-of-speech tag is associated with a token.

5 Conclusion

In this paper, we have given a brief overview of the LAPPS Web Service Exchange Vocabulary (WS-EV), which provides a terminology for a core of linguistic objects and properties exchanged among web services that consume and produce linguistically annotated data. The goal is to enable semantic interoperability among NLP data, tools, and services within the LAPPS Grid. While we recognize the inherent problems of defining a type system for linguistic objects, the LAPPS Grid cannot operate without semantic interoperability among services, and the WS-EV is our means to fulfill that requirement. Our approach is therefore notably bottom-up (adding objects and properties as needed) and treads a fine line between over- and under-specification. Ideally, the WS-EV will be useful to others, either web service providers or users of systems like UIMA, as a point of departure for defining type systems etc., and potentially provide a base upon which others can usefully build.

Acknowledgements. This work was supported by National Science Foundation grants NSF-ACI 1147944 and NSF-ACI 1147912.

References

1. Cieri, C., DiPersio, D., Wright, J.: Intellectual property rights management with web services. In: Proceedings of the Workshop on Open Infrastructures and Analysis Frameworks for HLT, Dublin, Ireland, August 2014
2. Ferrucci, D., Nyberg, E., Allan, J., Barker, K., Brown, E., Chu-Carroll, J., Ciccolo, A., Duboue, P., Fan, J., Gondek, D., Hovy, E., Katz, B., Lally, A., McCord, M., Morarescu, P., Murdock, B., Porter, B., Prager, J., Strzalkowski, T., Welty, C., Zadrozny, W.: Towards the open advancement of question answering systems. Technical report, IBM Research, Armonk, New York (2009)
3. Ferrucci, D.A., Brown, E.W., Chu-Carroll, J., Fan, J., Gondek, D., Kalyanpur, A., Lally, A., Murdock, J.W., Nyberg, E., Prager, J.M., Schlaefer, N., Welty, C.A.: Building Watson: an overview of the DeepQA project. AI Mag. **31**(3), 59–79 (2010)

4. Fokkens, A., van Erp, M., Postma, M., Pedersen, T., Vossen, P., Freire, N.: Offspring from reproduction problems: what replication failure teaches us. In: Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, vol. 1, pp. 1691–1701. Association for Computational Linguistics, Sofia, Bulgaria, August 2013. http://www.aclweb.org/anthology/P13-1166
5. Garduno, E., Yang, Z., Maiberg, A., McCormack, C., Fang, Y., Nyberg, E.: CSE framework: a UIMA-based distributed system for configuration space exploration unstructured information management architecture. In: Klgl, P., de Castilho, R.E., Tomanek, K. (ed.) UIMA@GSCL, CEUR Workshop Proceedings, pp. 14–17. CEUR-WS.org (2013)
6. Giardine, B., Riemer, C., Hardison, R.C., Burhans, R., Elnitski, L., Shah, P., Zhang, Y., Blankenberg, D., Albert, I., Taylor, J., Miller, W., Kent, W.J., Nekrutenko, A.: Galaxy: a platform for interactive large-scale genome analysis. Genome Res. **15**(10), 1451–1455 (2005)
7. Hellmann, S., Lehmann, J., Auer, S., Brümmer, M.: Integrating NLP using linked data. In: Alani, H., Kagal, L., Fokoue, A., Groth, P., Biemann, C., Parreira, J.X., Aroyo, L., Noy, N., Welty, C., Janowicz, K. (eds.) ISWC 2013, Part II. LNCS, vol. 8219, pp. 98–113. Springer, Heidelberg (2013). http://svn.aksw.org/papers/2013/ISWC_NIF/public.pdf
8. Ide, N., Pustejovsky, J.: What does interoperability mean, anyway? toward an operational definition of interoperability. In: Proceedings of the Second International Conference on Global Interoperability for Language Resources, ICGL (2010). http://www.cs.vassar.edu/~ide/papers/ICGL10.pdf
9. Ide, N., Pustejovsky, J., Calzolari, N., Soria, C.: The SILT and FlaReNet international collaboration for interoperability. In: Proceedings of the Third Linguistic Annotation Workshop, ACL-IJCNLP, August 2009
10. Ide, N., Pustejovsky, J., Cieri, C., Nyberg, E., Wang, D., Suderman, K., Verhagen, M., Wright, J.: The language application grid. In: Proceedings of the Ninth International Conference on Language Resources and Evaluation LREC 2014. European Language Resources Association (ELRA), Reykjavik, Iceland, May 2014
11. Ide, N., Suderman, K.: The linguistic annotation framework: a standard for annotation interchange and merging. Lang. Resour. Eval. **48**(3), 395–418 (2014)
12. Ishida, T. (ed.): The Language Grid: Service-Oriented Collective Intelligence for Language Resource Interoperability. Springer, Heidelberg (2011)
13. ISO-24612: Language Resource Management - Linguistic Annotation Framework. ISO 24612 (2012)
14. Patel, A., Yang, Z., Nyberg, E., Mitamura, T.: Building an optimal QA system automatically using configuration space exploration for QA4MRE 2013 tasks. In: Proceedings of CLEF 2013 (2013)
15. Poch, M., Bel, N.: Interoperability and technology for a language resources factory. In: Proceedings of the Workshop on Language Resources. Technology and Services in the Sharing Paradigm, Asian Federation of Natural Language Processing, Chiang Mai, Thailand, pp. 32–40, November 2011
16. Villegas, M., Bel, N., Bel, S., Rodrguez, V.: A case study on interoperability for language resources and applications. In: Proceedings of the Seventh International Conference on Language Resources and Evaluation, LREC 2010, Valletta, Malta, May 2010
17. W3C OWL Working Group: OWL 2 Web Ontology Language: Document Overview. W3C Recommendation (2012)
18. W3C SKOS Working Group: SKOS Simple Knowledge Organization System Reference. W3C Recommendation (2009)

19. Windhouwer, M.: RELcat: a Relation Registry for ISOcat data categories. In: Calzolari, N., Choukri, K., Declerck, T., Dogan, M.U., Maegaard, B., Mariani, J., Odijk, J., Piperidis, S. (eds.) LREC. European Language Resources Association (ELRA), pp. 3661–3664 (2012)
20. Yang, Z., Garduno, E., Fang, Y., Maiberg, A., McCormack, C., Nyberg, E.: Building optimal information systems automatically: configuration space exploration for biomedical information systems. In: Proceedings of the CIKM 2013 (2013)

http://www.springer.com/978-3-319-31467-9

Worldwide Language Service Infrastructure
Second International Workshop, WLSI 2015, Kyoto, Japan, January 22-23, 2015. Revised Selected Papers
Murakami, Y.; Lin, D. (Eds.)
2016, X, 201 p. 71 illus. in color., Softcover
ISBN: 978-3-319-31467-9