Subject: LIME proposal for Metadata Module inside OntoLex

Here would provide:

1. a quick recap of the old LIME
2. a list of desiderata/requirements for the updated version to be considered for OntoLex
3. a rationale behind the adoption of a dedicated Lexicalization element for the metadata
4. a concrete proposal
5. discussion about ratio vs integer issue
6. discussion about which elements should be counted to represent lexicalizations

Legenda

Ontology: here always used in a very broad sense. Ontology may be a dataset, an owl ontology, a skos thesaurus and so on. It is meant, in the onto-lex dualism, as the conceptual part of a domain model, as opposed to its lexicalization.

Lexicalization: there is a minor ambiguity here. Later we will introduce “lexicalizations” as the links between entries in a Lexicon and concepts in an ontology. We also use the term “Lexicalization” to address the dataset containing a collection of lexicalizations for an ontology by using entries from a given lexicon. We will make consistent use of lowercase/uppercase for the initial letter to clarify the intended meaning.

1. A one page recap about LIME

LIME (and before that, Linguistic Watermark) was thought before OntoLex, with simple lexicalizations in mind, usually part of the ontology itself. Therefore, LIME was more about letting an ontology/thesaurus self-describe its linguistic characteristics (this was part of a LIME OntoLinguistic module, OL in short). There was another module dedicated to represent lexical resources (a part mostly cut from OntoLex). We will focus here on the first one.

The OL had two main classes of objects describing how an ontology/thesaurus is lexically covered:

- `lime:LanguageCoverage` contains things such as percentage of covered resources, and average number of entries per resource. It was thought to describe how much a language is covering the ontology (or partitions of it, by classes/properties, etc.),
- `lime:LexicalResourceCoverage` represents the coverage of the same elements, considering not a natural language, but what we could call *LexicalConcepts* in OntoLex (e.g., synsets in WordNet). The intent was clear: if two ontologies were described with links to (e.g.) synsets, alignment processors could use this semantically richer interlingua for a mediation activity.

Here follows an example of a `lime:languageCoverage` (`lime:languageCoverage` is the property, while the *bnode* pointed here is an element of class `lime:LanguageCoverage`, uppercase initial char). In the example, we describe a dataset (`:dat`) in which 75% of `skos:Concepts` have English lexicalizations with an average number of such lexicalizations per concept equal to 3.5.

```
:dat lime:languageCoverage [
  lime:lang "en";
  lime:resourceCoverage [
    lime:class skos:Concept;
    lime:percentage 0.75;
    lime:avgNumOfEntries 3.5
 ]
].
```

Another example illustrates the use of `lime:lexicalResourceCoverage` (a property connecting a `void:Dataset` to an instance of `lime:LexicalResourceCoverage`, upper case initial char) to describe a dataset (`:dat`) in which 75% of `skos:Concepts` have links to synsets of WordNet 3.0 with an average number of such links per concept equal to 3.5

```
:dat lime:lexicalResourceCoverage [
  lime: lexresource <http://purl.org/vocabularies/princeton/wn30/>;
  lime:resourceCoverage [
    lime:class skos:Concept;
    lime:percentage 0.75;
    lime:avgNumOfEntries 3.5
 ]
].
```

In bringing this to OntoLex, we were initially going to at least:

- just rename `LanguageCoverage` to `LexicalCoverage`
- consider if `lime:LexicalResourceCoverage` could be embodied in `LexicalCoverage` somehow, with some discriminating element

2. Desiderata/Requirements for OntoLex

In the attempt to bring LIME into Ontolex, and updating it where necessary, we collected the following desiderata:

1) we would retain the possibility for LIME to describe lexicalizations expressed through traditional lexical facilities (rdfs:label, skos labeling properties, skosxl reified labels, or even take into account human readable names used for the local names of the ontology URIs)

2) acknowledge the specific case of a distributed scenario of publication as envisaged by Ontolex.

3) Define best practices for the reuse of existing vocabularies

With respect to the original LIME model, we should move away from the original assumption that lexicalizations are embedded in a ontology, in order to accommodate the flexible and distributed publication model envisaged by the OntoLex community. In fact, lexica can be published as separate datasets, and may not relate to any specific ontology, such as general-purpose lexical resources (e.g. WordNet), which may be later be bound to a specific conceptualization.

Therefore, we propose to distinguish three kinds of entities at the metadata level:

1. the *Ontology*;
2. the *Lexicon*;
3. the *Lexicalization*

These classes (from now on: OLL classes) are specializations of the more general concept of `void:Dataset`, that is a set of triples published under a single administrative authority. As it already holds for void:Dataset, such categories are not required to exist at the level of plain data (i.e. data should not necessarily mention them) and be only recognized at the level of metadata.

Entities of the aforementioned classes may be published as distinct data sources (e.g., each one with its own SPARQL endpoint), or combined into a single source. By allowing this freedom, we intend to support the following scenarios:

1. a lexicon is published as a stand-alone resource (e.g. WordNet), independently of any specific ontology. Then the following two cases may happen
    a. an ontology contains already a Lexicalization adopting entries of the lexicon (thus ontology+lexicalization as a single data source)
    b. an ontology exist independently of the lexicon, and a third party publishes a Lexicalization of the ontology by adopting the above lexicon (thus all the three datasets are separate entities)

2. a lexicon is created for a specific ontology (lexicon and lexicalizations published together);

3. an ontology exists with an embedded lexicon (ontology, lexicon and lexicalization published together).

Obviously, since ontologies may be lexicalized for more languages, and as a general-purpose lexicon may be reused across different ontologies, multiple combinations of the above cases may happen for a single ontology/lexicon when being interfaced with, respectively, other lexicons/ontologies.

With respect to any of the categories, we should be able to represent provenance (e.g., creator and last modification date) and structural metadata (e.g., various metrics). There are several compelling reasons for these metadata, including preservation in archives and quality assessment.

2.1 Some Scenarios

Possible scenarios requiring (or at least, benefiting from) metadata include Ontology Alignment (OA), especially if carried on the fly, and Information Extraction and Triplification.

In OA, the input is a couple of ontologies <O1, O2>. In order to be able to carry on the alignment task, agents should be able to:

1) retrieve Lexicalizations of the involved ontologies (and metadata before concretely access them)
2) evaluate the best lexical compatibility between O1 and O2, given the search space of retrieved Lexicalizations

In IE&T, the input is a pair <O, T> (ontology and text) where, to our purpose, this ultimately translates into a <O, L> that is an ontology, and its available lexicalizations for language L.

Beyond general-purpose metadata (author of Lexicons, publication date etc..), we must introduce metadata that are specifically tailored to each of the OLL classes. While some of the proposed metadata could be derived systematically from data, an explicit vocabulary of metadata and precompiled values might be useful in many circumstances, as:

1. metadata are costly to compute,
2. we may be looking-up a resource in a directory (thus the resource is not immediately available)
3. we are reasoning on resources to assess their compatibility or usefulness for a given task, therefore we need a terminology to express succinctly facts that might otherwise depend on complex queries (possibly depending, in turn, on the given contour conditions).

3. Rationale for Lexicalizations

Considering the metadata we already brought from LIME, such as the avgNumOfEntries (and even more evidently in the xxxCoverage!) it is clear the we are interested in the number of attachments (that is, of lexicalizations), and not in the number of lexical entries.

There is a series of very good reasons for that:

1) Consider Scenario 1 of section 2 (independent lexicon): the number of lexical entries in the lexicon is useless for our counts if not all of them are involved in the lexicalization (which will not happen to be
    a. As a consequence, the number of lexical entries for the lexicon may still be considered as a useful metadata per se (so, we do not have to make a choice and we can keep both), but again, it has to be local to the lexicon, and is not relevant for the onto-lexical metadata

2) Even with a 100% participation of lexical entries to a lexicalization, a lexical entry could participate in lexicalizing two concepts (polysemy), and we would really prefer to tell that two concepts benefited from that lexical content

3) In the specific case of the xxxcoverage properties, the real target is the amount of concepts being lexicalized, so it is in no way related to the amount of lexical entries. If we had 100 skos:Concepts and 1000 lexical entries, and only one concept covered by those 1000 lexical entries which happen to be synonyms, then the coverage for class skos:Concept is sadly 1%.

See in this sense, the distinction we already made in the LIME paper [1] (which actually dates back to the precursor of Lime, the Linguistic Watermark [2, 3]) about “lexical metadata” and “onto-lexical metadata”

[1] http://aclweb.org/anthology//W/W13/W13-5504.pdf
[2] http://art.uniroma2.it/publications/docs/2008_OntoLex08_Enriching%20Ontologies%20with%20Linguistic%20Content%20an%20Evaluation%20Framework.pdf
[3] http://iospress.metapress.com/content/x043167268663268/

4. Proposal

The proposed metadata module is an extension of the VoID vocabulary, which provides specific vocabulary for characterizing lexica and how RDF dataset are linguistically grounded. Meanwhile, LIME inherits from VoID the publishing conventions, and other recommendations concerning the use of property from popular vocabularies, e.g. Dublin Core Terms.

```
Prefix: lime: <…>
Prefix: rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
Prefix: dc:   <http://purl.org/dc/terms/>
Prefix: void: <http://rdfs.org/ns/void#>
Prefix: xsd:  <http://www.w3.org/2001/XMLSchema#>

Class: lime:Lexicon
   SubClassOf: void:Dataset .
```

A lexicon modelled according to the OntoLex core model.

```
Class: lime:Lexicalization
   SubClassOf: void:Dataset .
```

A lexicalization of an RDF dataset is in fact an association between an ontology and its lexicon, i.e. the natural language realization of resources found in the dataset. This proposal aims at supporting a variety of scenarios, ranging from simple RDFS labels, through reified SKOS-XL labels, to OntoLex lexica.

The pattern for a lexicalization should be something like that:

```
lime:Lexicalization
  --lime:lexicalizedDataset--> rdfs:resource  (thought to hold the lexicalized ontology)
  --lime:lexicalModel--> rdfs:resource  (thought to hold rdfs:, skos:, skosxl:, ontolex: )
  --lime:resourceCoverage-->  (the presented stat info, see discussions below)
```

The one above has the advantage of using the same proxy of the elements we are describing (the bindings) as subject in the triples of its own void file.
An example of its usage:

```
/** inside the void file of a lexicalized ontology (if the Lexicalization is known to exist by the ontology)
:dat a void:Dataset;
     lime:lexicalization myItLex:myItalianLexicalizationOfDat

/** inside the void file of the Lexicalization
myItLex:myItalianLexicalizationOfDat
  a   lime:Lexicalization;
  lime:lang "it";   // important to be here, this is the focus of search by agents!!! Not the lexicon!
  lime:lexicalizedDataset :dat ;
  lime:lexicalModel ontolex: ;
  lime:lexicon :italianWordnet;
  lime:resourceCoverage [   // see discussion later in sections 5
    lime:class owl:Class;
    lime:percentage 0.75;
    lime:avgNumOfEntries 3.5
  ].
```

A single Lexicalization may expose more resourceCoverage entries for different classes (e.g. telling the coverage related to classes, properties, or for skos:Concepts).

The above example would be very easy to remap in the simpler case of a dataset holding its own labels (as for simple rdfs skos or in most of the cases even skosxl labels). In this case, we can simply declare that:

```
:dat lime:lexicalization :myItalianLexicalizationOfDat
```

And add the following to myItalianLexicalizationOfDat

```
:myItalianLexicalizationOfDat void:subset :dat
```

In the case above, both of them would be on the same void file.

Obviously, each dataset (unless a subset of another one), brings with it all the usual metadata specified by void (SPARQL endpoint, distribution, etc…) which is not reported here for sake of conciseness.

P.S: note for future clarification: in the case of RDFS / SKOS / SKOS-XL, it should be made clear which information should be reported for the lime:lexicon.

5. Discussion about ratio vs integer issue

Inside a lexicalCoverage – which represents the level of coverage for a certain type T of elements of the ontology, for a given language L – the following two properties are defined:

1) *avgNumberOfEntries:* could be defined as #lexicalizations / #concepts

2) *percentage*: the percentage of elements of type T, covered by at least an entry in the language L

The use of ratios and percentages has been criticized by some of the ontolex members (mainly John, sympathy from Philipp) for being redundant, as the denominator could be provided by metadata about the ontology, so it would suffice to provide the more explicit integer about the number of lexicalizations.

There are some good reasons which motivate us to insist for ratios/percentages:

1. In a completely distributed scenario, the denominator may not be available (e.g. an owl ontology has no metadata at all), or at least, of no immediate retrieval (must get to the other dataset metadata)

2. The criticism was born when lexical entries were thought to be used for the ratio, which has already been proven not correct: lexicalizations are to be used instead. So, Lexical Entries count will be reported as an integer value for the Lexicon metadata. However, in the context of using lexicalizations, there is little value added in reporting their count.

3. The percentage is not redundant (so this in any case requires a dedicated number). Due to the already mentioned phenomena of polysemy and synonymy, this percentage is not obtainable through any kind of ratio on the available counts and must be calculated instead by determining the number of concepts that are covered by *at least* a lexicalization.

4. The contra based on the fact that the target ontology may vary (and thus the integer is more robust) is not valid. The target ontology “must be” the correct version (OWL 2 provides version IRI, datasets may have publication date or other metadata), otherwise even integer counts may refer to lexicalizations of concepts which does not exist anymore.

A note about the average number of entries: we could decide that the average is actually calculated on the total number of considered ontology elements, or on the sole elements which have at least a lexicalization (which would be even more easily obtainable, and thus local, to the lexicalization, and also more resilient to changes in the target ontology).

6. Lexicalization core triples: senses or what?

Senses act as reifications of the relationships between LexicalEntries and Conceptual Entities (be them LexicalConcepts or entities of the lexicalized ontology). In effect, a single sense is always 1-1 (it links a single Lexical Entry with a single Conceptual Entity)
The ontolex model has a shortcut for the relationship (mediated by senses) between LexicalEntries and LexicalConcept: ontolex:denotes.

We would propose to formally consider the number of denotes triples (triples with predicate == ontolex:denotes) to obtain the count. Obviously, this information may not always be available (not inferred), though the detail of how to obtain this are just technicalities.
To support our claim, please note the following case:

1. a lexicon exists (independently of an ontology), with one lexical entry having two very close senses (two smooth variations of a broad meaning)
2. the lexicon is used to lexicalize an ontology
3. the authors of the Lexicalization decide to collapse the two senses into the same ontology concept
4. the two triples connecting the two similar senses to the same ontology concept entail the same ontolex:denotes triple
5. to the purpose of counting the lexicalizations of that lexical concept, the single triple count on ontolex:denotes is more appropriate than counting the two senses of a same LexicalEntry linked to the same concept.