User's Handbook for the Icelandic Wordweb

Hjalti Daníelsson
The Árni Magnússon Institute for Icelandic Studies

1. Introduction

This is a user's handbook for the new version of the Icelandic Wordweb ("Íslenskt Orðanet").

The original Wordweb was a database of words, categorizations and word relations, presented through a web site tailored for specific ways in which to use its data. The new version consists of a single, large RDF file that hosts this content and is encoded with a standardized vocabulary. While the new format opens up a vast array of new options for exploring the Wordweb, its open-ended nature also poses a nontrivial barrier to entry, particularly for those unfamiliar with the original's data and design.

This handbook thus has the aim of making the Wordweb more accessible for its users. It describes the original Wordweb's architecture, explains how those features have been implemented in the new format, and demonstrates some of the ways it may be used and explored. Some familiarity with OntoLex, SKOS, SPARQL, and RDF triplets might be required, although we do our best to explain the most important facets of each one.

Please note that this handbook is not meant as an exhaustive list of the ways in which the Wordweb may be traversed and used. Rather, it is intended to clarify the primary aspects of the Wordweb's design and possible use, and to point out most significant ways in which it maintains the features of the original. In addition, it should be kept in mind that the models in which the new Wordweb was implemented are immensely flexible. In our design of the Wordweb, we've tried to hew close to how these models were intended to be used, and not break convention any more than necessary. While we do on occasion mention how certain limitations guided that design, we are not necessarily implying that these limitations couldn't have been overcome or circumvented with different solutions, but simply that the particular design path we chose seemed sensible in context.

In chapter 2 we provide the necessary background on the original Wordweb, and on the two models, OntoLex and SKOS, in which the new version was implemented. This is also the last chapter where these entities are discussed separately. Each of the ensuing chapters examines a particular Wordweb feature, demonstrating its design in the original Wordweb and the corresponding implementation and use in OntoLex and SKOS: Chapter 3 explores the Wordweb's basic elements, chapter 4 explains the relationships between these elements, and chapter 5 looks into some of the more specialized functionality.

2. The models

In this chapter we look at the architecture of the original Wordweb, and discuss certain basic features of the models in which the new version was implemented. This is by no means an exhaustive list: The goal here is simply to ensure sufficient understanding of these models so that later chapters will make sense to the reader.

2.1 The original Wordweb

The Wordweb is a semi-categorized collection of lemmas and their relations. To better distinguish the two core Wordweb entities, we'll capitalize them from here on: Lemmas ("Fletta" singular / "Flettur" plural) and Concepts ("Hugtak"/"Hugtök").

Lemmas come in many varieties: Monolexical (single-word) and polylexical (multi-word), unordered and ordered, sourced both from reference works and primary sources, and sometimes accompanied with various types of explanatory and morphosyntactic metadata. Lemmas generally do not have definitions except in cases where it's necessary to differentiate word forms, rather, their meanings are considered to be implicit in the relations they have to other Lemmas.

The relations between these lemmas are likewise a mix of different types. There is a heavy slant toward semantic relations, meaning that lemmas tend to be connected if their respective meanings have some common factor, but there is also a syntactic aspect.

The categorization schema is composed of Concepts. While it is quite extensive, it is not intended to be all-encompassing. Each Lemma may belong to one, none, or multiple Concepts. Their schema has a flat structure: All Concepts have equal priority, there are no Concept hierarchies, and from a semantic perspective their subjects may overlap. Although Concepts exist as separate entities in the Wordweb (not just as properties of Lemmas), there are no direct Concept-to-Concept relations; they are considered to be related only through the Lemmas that belong to them.

One of the Wordweb's greatest strengths is its magnitude. It is an extensive and growing collection, numbering well over 200,000 Lemmas and innumerable relations of various types. It contains a wealth of semantic relations that may seem obvious to the reader but cannot be found in more heavily curated and structured reference works.

But with this magnitude comes a degree of complexity, particularly in terms of encoding all that information in machine-readable fashion. This is compounded by the fact that the Wordweb has been built on and expanded over a development period of many years, through the work of several people employing multiple sources & databases. As a result, there is no single, exact type of Lemma in the original Wordweb's database tables. Some of the Lemmas may be written or encoded differently from others, some have more metadata, and in certain cases the metadata itself may also be encoded differently between lemmas.

In short, a direct conversion to an established format is simply not possible. The only way of standardizing the Wordweb while simultaneously maintaining its unique functionality has been to design and implement it in the new format almost from scratch. In addition, we've incorporated as much information as possible - including data that was unavailable in the original - while also trying to establish a more standardized storage.

2.2 OntoLex and SKOS

While there is no single standardized model capable of containing the entirety of the Wordweb's data, each of its two core systems - Lemmas and Concepts - lends itself quite well to a specific model: OntoLex and SKOS, respectively.

OntoLex (previously known as "lemon", short for "The Lexicon Model for Ontologies") is intended primarily to codify grammatical and syntactical information in linked data collections. This includes varying morphosyntactic forms and metadata, of which there is a great wealth in the Icelandic language. An important aspect of OntoLex is that while it is very suitable for storing this kind of information, it is generally neither applicable nor intended for other purposes such as categorization. We use OntoLex to store Lemmas, most of their metadata, and some - but importantly, not all - of their relations.

SKOS, on the other hand, is intended for classification and categorization. It does not store grammatical information, and generally is better suited for handling groups of entities rather than representing each one individually. We mainly use it to store the Wordweb's Concepts, and to represent certain complex relations. (The basic SKOS unit is also called a "concept". To avoid confusion, we'll use "Concept" for the Wordweb entity, and "SKOS concept" for the SKOS unit). There are two particular properties of SKOS that have proven very useful when implementing the new Wordweb. Firstly, since it is intended for a more general purpose than OntoLex is, it is also quite a bit more flexible. Secondly, its focus is on the relations between its entities rather than information on the entities themselves. Certain core Wordweb features straddle the gap between semantics and syntax, which made them a difficult fit for OntoLex. In those cases, we were able to leverage SKOS's flexibility and extensive relations vocabulary instead.

We'll now look at each aspect of the Wordweb, go into more detail about its original design, and explain how it was implemented - and should be handled - with the new models.

3. Elements

3.1 Lemmas

Lemmas are the core entities of the Wordweb, and all of its information - including Concepts - derives from them in some way.

Each Lemma always consists of at least three parts: A series of one or more words, with a specific collective meaning, and certain grammatical information. In OntoLex, these are called the Lexical Entry, Lexical Sense, and Form, respectively.

Lexical Entries

Although we will generally use the terms "Lemma" and "Lexical Entry" interchangeably, Lexical Entries are in fact the word or words that make up each Lemma, shorn of meaning and grammar. Lexical Entries are the absolute core of OntoLex; most of its functionality and data is meaningless unless attached to one of those.

It should be noted - and we will discuss this later in the handbook - that although Lexical Entries don't need to contain relations with other Lexical Entries (nor really any data other than their actual words), OntoLex effectively requires them to be connected to a corresponding ontology entity, via Lexical Senses. A Lexical Entry without this kind of connection is, by definition, meaningless. In our case, these ontology entities are the Concepts.

One of the few OntoLex properties that can be attached directly to Lexical Entries, and which we employ in the Wordweb, is its multiplicity: All Lemmas are identified either as monolexical (single-word), polylexical (multi-word) or affixes. Monolexical Lemmas may generally be considered lexemes, with each Lemma presented in its canonical form. Keep in mind, however, that as with many other languages, Icelandic has its share of words whose standard version may not be in the default form (such as "skæri", which shares the plurale tantum of its English "scissors" counterpart). As a result, monolexical Lemmas can not be assumed to always have the same syntactic properties, even in their dictionary form, and need to be accompanied by grammatical marks.

Polylexical (multi-word) Lemmas, meanwhile, are considered phrases, since they stand together as a conceptual unit, and may sometimes be idioms as well. Lemmas do not need to be fully

formed components or subclauses. For example, the Lemma "detta af hesti" translates to "fall off a horse", not "to fall off a horse" nor "he fell off a horse".

Lexical Sense

A Lexical Sense is not a "definition" in the traditional sense, inasmuch that it does not textually define its Lexical Entry. Instead, a Lexical Sense identifies a particular usage of a single Lexical Entry, by linking it with a specific ontological element (i.e. a Concept). In most cases, a Lexical Entry will only have one corresponding Lexical Sense. If the Lexical Entry represents a concept that has multiple definitions or meanings, then each of those meanings will have its own Lexical Sense.

The point about Lexical Senses deriving their meaning from their links with ontological elements is important. Recall that in the Wordweb, not all Lemmas belong to a single Concept. Some belong to many Concepts, others to no Concept at all. But the function that a Lexical Sense represents, from a Lexical Entry on one end to an ontological element on the other, has to be strictly one-to-one. Because of this issue, along with a separate one that has to do with how the Wordweb models its semantic relations, we created a special "encapsulation layer" between the OntoLex and SKOS implementations that serves as a connecting point for all Lexical Senses. We detail its design in chapter 3.2.

Forms

A Form is a specific written representation of a Lexical Entry, and may contain descriptive metadata to help the user identify its grammatical properties. In the Wordweb, this metadata is mostly morphosyntactic, although the precise contents vary between monolexical and polylexical Lemmas - the former tend to have far more detailed information - and between different grammatical categories. Most of this data is derived from The Database of Icelandic Morphology (DIM), and was stored separately in the original Wordweb's database tables.

For monolexical Lemmas, the new Wordweb distinguishes between the inflectional forms of nouns, adjectives, verbs, adverbs, and pronouns. Noun properties are gender, declension, number, and the definite article. Adjective properties are degree, gender, declension, and number. Verb properties are voice, mood, tense, person, number, gender, declension, and declension strength. Adverb properties are only degree. Pronoun properties are gender, declension, and number. For both mono- and polylexical Lemmas the Wordweb also specifically recognizes exclamations, conjunctions, prepositions, numerals and proper nouns, and specifically for polylexical Lemmas, it designates set phrases ("orðastæða"), phraseological units ("orðsamband") and idioms ("orðtak") (In reality, the first two types are very similar. They have their own dedicated marks in the Wordweb - a special "x" mark for the former, and a separate "OSAMB" designation in a dedicated database table for the latter - but are effectively almost the same thing. In the Wordweb, phraseological units tend to be complete phrases or

idioms, with a meaning that is clear, definite, and well removed from the meaning of their individual words. Set phrases, meanwhile, generally don't have all the words that would be needed for a fully formed unit, and their meaning tends to be more obscure and less clearly defined.)

Note that for each word in a polylexical Lemma, generally only the grammatical category is specified. There are several reasons for this, but in brief, single-word data for the Wordweb's polylexical entries is quite challenging to establish, implement, and represent. The easiest and most efficient way - particularly in a single-file implementation, where we want to avoid needless repetition of metadata - is to create a link between the individual words in a polylexical Lemma, and their monolexical Lemma counterparts. We've done this wherever possible, and also pointed to the monolexical Lemma's specific morphosyntactic form when appropriate.

An important factor in our design and implementation of Forms is space. As can be seen by the list of properties for monolexical Lemmas, Icelandic words tend to have a high number of morphosyntactic variations, each of which has its own written representation. To prevent the Wordweb's file container from absolutely blowing up, we've created standardized Forms for every possible combination of every grammatical category: One Form for all "noun-masculine-nominative-singular-no definite article" words, another Form for all "noun-masculine-nominative-plural-no definite article", and so on. Every morphosyntactic variation of every monolexical Lemma is thus assigned only two properties: Its written representation, and a link to one of these standardized Forms.

In the RDF file, our naming scheme for Lemmas is "[dictionary form]_[grammatical category]_fl". Recall that while the same Lemma may have different meanings, these are distinguished through the use of Lexical Senses. Thus, the only real identifier we need for each Lemma is its grammatical information, not its meaning. In most cases the grammatical category is sufficient, although for a few rare cases some additional information has been encoded to ensure each Lemma's identity remains clear. The "fl", meanwhile, is shorthand for "Fletta", Icelandic for the Wordweb's Lemmas, and is intended to help distinguish them from the Concepts and LaCs described in chapter 3.2.

3.2 Concepts, LaCs, and the SKOS encapsulation layer

The Wordweb's Concepts are implemented as SKOS concepts. They represent the ontology that OntoLex needs in order for its Lexical Entries and Lexical Senses to work.

Concepts are a much less complex entity than Lemmas. Their only properties are their individual names, and the only formal relations they have in the Wordweb are with the Lemmas that belong to them. The entire Concept structure was constructed manually, rather than from

direct sources, and functions as an addendum of convenience rather than an all-encompassing system.

Thanks to the malleability both of the Wordweb Concept system and the SKOS model in which we've encoded it, we were able to extend its functionality to cover two OntoLex prerequisites that would otherwise have proven rather problematic. One - the necessity of one and exactly one Lexical Sense connecting each Lemma/Concept pair - has already been described in chapter 3.1. The other will be detailed in chapter 4, but in brief, we had to implement a specific type of relation between certain Wordweb entities, where the entities had to be encoded in OntoLex but the relation couldn't be represented in OntoLex.

In both cases, what we really needed was the ability to connect every OntoLex Lexical Entry to exactly one SKOS concept, no more nor less. We therefore created the aforementioned encapsulation layer around the entire Concept system and populated it with what we termed "Lemmas-as-Concepts" or LaCs. Each LaC is effectively a combination Lexical Entry and Lexical Sense, recreated as a SKOS concept. Note that in the actual RDF file, we can then comfortably employ the SKOS "broader" and "narrower" properties to create a two-way link between each LaC and Concept pair.

It may help to think of LaCs as adaptor plugs that ensure all Lemma-type entities can be connected to the Concept system. Thus, instead of the connection between the two systems being represented as follows, where no Lexical Sense can be established for Lemma_B, and thus OntoLex cannot properly represent Lemma_B itself:

Lexical Entry		Lemma_A		Lemma_B		Lemma_C
		↓		↓		↓
Lexical Sense		✓		X		✓
		↓				↓
SKOS concept		Concept_A		[No concept]		Concept_C

We now have fully-fledged connections for all Lemmas, as follows, where all Lemmas are plugged into Concept objects and thus are considered valid entities by OntoLEx:

Lexical Entry		Lemma_A		Lemma_B		Lemma_C
		↓		↓		↓
Lexical Sense		✓		✓		✓
		↓		↓		↓
Lemma-as-Concept		LaC_A		LaC_B		LaC_C
		↓				↓
SKOS concept		Concept_A		[No concept]		Concept_C

We can then use the SKOS "senseOf" keyword to relate one LaC to another without implying precisely what type of relation this is (as opposed to the OntoLex "isSenseOf", for example, which requires the relation to express either synonymy or hierarchical structure). This implementation also has the advantage of confining this rather nonstandard use to a single location in our model, which is much preferable to, say, weaving nonstandard keywords (or worse, standard keywords in a nonstandard fashion) into the general fabric of our OntoLex and SKOS data.

In terms of raw OntoLex and SKOS encoding, this means that we need to distinguish between Lemmas, LaCs and Concepts in our RDF file. To do this, we add a shorthand type descriptor to each entry. As we've mentioned, Lemmas have "fl". Concepts have "ht" (short for "hugtak", the Wordweb's Icelandic term for "concept"), while LaCs have "FsH", Icelandic for "Fletta-sem-Hugtak" ("Lemma-as-Concept").

3.3 Element grouping

Before we move on to describing these elements in detail, one final note about their organization. Both OntoLex and SKOS allow the grouping of individual entities. This is an immensely powerful feature for one simple reason: It effectively allows us to assign a common property to any group of elements without having to encode that property with OntoLex or SKOS keywords. As a result, this kind of grouping may be used to extend the models' functionality in a myriad of nonstandard ways.

We have, however, intentionally avoided this kind of grouping, choosing to assign it only to Lemmas, Concepts and Lacs, which respectively belong to the OntoLex-Lexicon type "limeLexicon" and the SKOS-ConceptSet type "conceptScheme_ht" and "conceptScheme_fsh" collections. The reason we've avoided it is simply that these collections can very quickly become rather unwieldy. There is no way to encode any documentation for them in the Wordweb file, and calling on their elements through the SPARQL query language is somewhat more complex than calling on normal element properties. Future Wordweb versions might implement them in greater detail (one obvious option would be to group the word suggestions found inside [square brackets] into discrete sets, where each set might easily be assigned a corresponding Lexicon), but for now, we constrain ourselves to the three groupings listed above.

4. Relation types and SPARQL queries

4.1 Pairs

Aside from the "Lexical Entry - Lexical Sense - ontological element" tuples, the original Wordweb contains one fundamental semantic relation and three ancillary ones. It also contains a wealth of derived relations that build on these. In the original Wordweb, these derived relations are hardcoded as part of the Wordweb's web presentation. As a result, search functionality for these relations, based on each Lemma, is simple and straightforward - but is also absolutely fixed, which means the search parameters cannot be altered, and the search results cannot be exported for further examination.

In the new Wordweb, we have encoded only the fundamental semantic relations into the system's data. All other relations may be produced through queries written in the SPARQL query language. We will now explain how these semantic relations work, and after that explain the basis of SPARQL and provide queries for all the other types of relations.

Those fundamental relations are called parallel constructions, which we'll refer to hereafter with the shorthand "Pairs" (the Icelandic term employed by the Wordweb is "pör"). In the Wordweb, a Pair consists of two unordered Lemmas that have appeared somewhere in the source texts with the conjunction "and" (Icelandic "og") between them. For example, if we have the preexisting Lemmas "dog" ("hundur") and "cat" ("köttur"), and if somewhere in one of our sources we encounter the words "dog and cat" ("hundur og köttur", or "hundar og kettir" plural), then we add a Pair relation to the two Lemmas: "Dog" now forms a Pair with "cat", and vice versa.

(Without diving too deep into the semantics of synonymy and word relations, we should note that this kind of relation isn't of a single clear type. The Lemmas' pairing indicates they may be used in the same context but that they are not necessarily interchangeable: Their definitions may thus be related, and occasionally even shared, but may not be identical. There is also the implication of syntax, since a Pair relation indicates they have appeared together in a sentence. There is even a slight indication of shared categorization: The two Lemmas may not have the exact same meaning or syntactical purpose, but they clearly *go together* in a fashion. The nebulous nature of these relations allows us to look further than simple synonymy, and - as we'll see below - to build even more extensive and interesting relations on top of the Pairs.)

In terms of implementing Pairs in OntoLex and SKOS, recall from chapter 3.2 that while Lemmas are OntoLex entities, the Pair relation itself is difficult to represent solely with OntoLex

functionality, seeing as how a prerequisite for that functionality is that it represent grammar and syntax - not semantics (at least not unless it can be avoided), and absolutely not categorization.

(To avoid any misunderstanding, it should be clear that OntoLex does support a number of Sense-to-Sense relations, and that these relations do cover a fair number of different types of semantic and terminological relations, including synonym and antonymy, printed variants, different emphasis on a specific subproperty of the Entry in question, etc. Doubtlessly these are very useful when required - but unfortunately they don't cover the semantic/syntactic/categorizational mix inherent in Pair relations.)

That's where SKOS comes in. We employ its "senseOf" keyword to encode Pair relations, and since that means we need entities that are SKOS concepts, rather than OntoLex Lexical Entries, we use the encapsulation layer to represent these relations. Let's look again at the encapsulation layer table, but this time indicate where the Pair relations would be encoded, if Lemma A and Lemma B were Pairs (see the sole entry in the table's middle column):

Lexical Entry		Lemma_A		Lemma_B		Lemma_C
		↓		↓		↓
Lexical Sense		✓		✓		✓
		↓		↓		↓
Lemma-as-Concept		LaC_A	↔ Pair	LaC_B		LaC_C
		↓				↓
SKOS concept		Concept_A		[No concept]		Concept_C

4.2 Other encoded relations

Let's now examine the rest of the Wordweb's relations that are encoded into the system itself.

Unlike Pairs, these are not used as building blocks for other relations. They are synonyms, antonyms, "links" (Icelandic term is "stiklur"), and compound words ("samsetningar"). In the original Wordweb, the first three are grouped together under the moniker "assessed relations" ("metin vensl").

The compound words associated with a particular (monolexical) Lemma are instances where the Lemma corresponds either to one of the two stems that form a compound word, or to the compound word itself. Note that these are not exhaustive, and only involve words divided into exactly two stems.

There is always at least one other Lemma present in this relation. For example, if we imagine that "sun" is a Lemma, and that one of the compound words related to it is "sunflower", either the Lemmas "sunflower" or "flower" (or both) would also be part of this relation. Likewise, if "sunflower" were the original Lemma, and if it contained a compound word relation, either the Lemmas "sun" or "flower" (or, again, both) would also be part of this relation.

The assessed relations have usually been derived from older thesaurii and other sources of that type. They form a relatively minor part of the Wordweb. Links are generally considered near-synonymy relations. Since synonyms and antonyms are one of the few relation types explicitly defined in OntoLex, we've opted to encode them using the OntoLex "Sense Relation" functionality rather than implementing them in the encapsulation layer.

Links, however, required custom relations. We've employed the SKOS-XL "labelRelation" functionality for this purpose, and created "stikla" as a special "subProperty". We then apply these to the appropriate LaCs.

4.3 SPARQL and derived relations

We will now go over each of the original Wordweb's derived relations - the ones based on Pairs - and for each, describe its nature, its scope, and its corresponding SPARQL query.

Before we move on to the relations, however, let's briefly look at the query language itself. SPARQL is an RDF query language that, thanks to our choice of models, is directly applicable to the data encoded in the Wordweb RDF file. Its queries are usually in the form:

```
SELECT * WHERE {
   ?s ?p ?o .
}
```

The letters "s", "p" and "o" stand for variables that represent subject-predicate-object relationships, often in a kind of "entity s has the relationship of type p with entity o" pattern. (You don't always have to use all three.) Keep in mind that some queries depend on the language in which the data is encoded. This means that you may have to affix "@is" to certain queries. A simple search for the LaC "mús" (Icelandic for "mouse") would thus be:

```
SELECT * WHERE {
   $concept rdfs:label $rdfsLabel .
   FILTER (str($rdfsLabel) = "mús")
}
```

Whereas a search for the actual Lemma "mús" (note the "@is" affix) would be:

```
SELECT * WHERE {
   ?concept ontolex:canonicalForm ?lexlabel.
   ?lexlabel ontolex:writtenRep "mús"@is.
}
```

Recall that we use the "skos:broader" property to establish a link from LaCs to Concepts. (This is a one-way link. The other way, from Concept to LaC, is represented by "skos:narrower.) If we're given a particular Lemma, and we want to check if that Lemma belongs to any Concepts, we would first query for the Lemma's corresponding LaC, and then run a query with the general pattern of:

```
SELECT * WHERE {
    ?s skos:broader ?o .
}
```

Where "?s" would be the LaC in question, and ?o would return all the Concepts it is connected to.

Let's move on to the derived relations. When we describe these, note that in most cases they rely on the user already having searched for and selected a particular Lemma. While the original Lemma's existence may not be explicitly mentioned, it's usually implied (for example, in the relation called "Lemmas sharing the same Concept", "the same" obviously implies "as the Lemma the user has already selected). When we diagram these relations, we will use the term "Lemma" to indicate the original entity whose relations are being inspected, enumerate other terms as required to distinguish their hierarchies, and underline those entities that would appear to the user. (It might help to think of these entities as a list of results from a search query. Note that the user would not be presented with the full hierarchy of entities; only the underlined ones.)

Concept-based relations: Lemmas sharing a Concept ("Flettur undir hugtakinu")

The original Wordweb offers three types of relations that specifically involve Concepts. This effectively means that unlike most other relations, the user must not only select a particular Lemma, but also a particular Concept to which that Lemma belongs. Lemmas that do not belong to any Concepts are not considered for these types of relations.

The first is Lemmas that share the same Concept. This relation might be modelled as:

- Lemma
    - Concept
        - Lemma_1
        - Lemma_2
        - ...

Once a Concept has been selected, we can use the following pattern for a SPARQL query to list its LaCs:

```
SELECT * WHERE {
   [CONCEPT] skos:narrower ?o .
}
```

As an example, if we've chosen the concept "MÚS" (for "MOUSE"), our query would be:

```
SELECT * WHERE {
   <http://orda.net/MÚS_ht> skos:narrower ?o .
}
```

(We will omit, from this and most future examples, the code required to specifically call forth Lemmas from LaCs. Once a LaC has been found, the Lemma has also effectively been found. For sample code that shows how Lemmas may be procured, see the chapter on Pairs.)

Concept-based relations: Pair-related Lemmas ("Venslaðar flettur í gegnum pör")

These are similar to the previous relation, except instead of looking for Lemmas that directly belong to the same Concept, we're looking only for Lemmas that form Pairs with those Lemmas:

- Lemma
    - Concept
        - Lemma_A
            - "and" LemmaA_1
            - "and" LemmaA_2
            - ...
        - LemmaB
            - "and" LemmaB_1
            - "and" LemmaB_2
            - ...
        - ...

The SPARQL query would be in two parts. Once the user has decided on a particular Concept, let's say "KÖTTUR" ("cat"), they would then call on all LaCs belonging to that Concept, and then for each one list every LaC (here ?svar) that it's Paired with:

```
SELECT ?svar WHERE {
   <http://orda.net/KÖTTUR_ht> skos:narrower ?FsH .
   ?FsH skos:related ?svar.
}
```

Concept-based relations: Related Concepts ("Tengd Hugtök")

This is the last of the three relations directly involving Concepts. By this point, the relation may start to look a little labyrinthine, but it has the same basic structure as the last relation.

Recall that we selected a Lemma, then its Concept, got a list of all the Lemmas under that Concept, and for each of those Lemmas inspected all its Paired Lemmas.

Now, for each one of those Paired Lemmas, we return not the Lemma itself, but all the Concepts (if any) that it belongs to:

- Lemma
    - Concept
        - Lemma_A
            - "and" LemmaA_1
                - ConceptA1_a
                - ConceptA1_b
                - ...

        - "and" LemmaA_2
            - ConceptA2_a
        - ...
    - LemmaB
        - "and" LemmaB_1
            - ConceptB1_a
        - "and" LemmaB_2
            - ConceptB2_a
            - ConceptB2_b
        - ...
    - ...

The relations that link the original Concept to the eventual Concept can thus be laid out as:

Concept ↔ LaC ↔ "skos:related" ↔ LaC ↔ Concept

If we again select "KÖTTUR" as the original Concept, the SPARQL query could be:

```
SELECT DISTINCT ?Concept WHERE {
   <http://orda.net/KÖTTUR_ht> skos:narrower ?Ht .
   ?Ht skos:narrower ?LaC1 .
   ?LaC1 skos:related ?LaC2 .
   ?LaC2 skos:broader ?Concept .
   FILTER (?Hugtak != <http://orda.net/KÖTTUR_ht>) .
}
```

Pairs (''Parasambönd'' or ''Pör'')

As previously mentioned, this relation is the cornerstone of the Wordweb. Luckily, it is easy to represent and query. The model would simply be:

- Lemma
    - "and" Lemma1
    - "and" Lemma2
    - ...

The corresponding SPARQL query then becomes:

```
SELECT DISTINCT ?AnswerLemma WHERE {
   <http://orda.net/köttur_fl> ontolex:denotes ?ChosenLaCs .
```

```
    ?ChosenLaCs skos:related ?PairedLaCs .
    ?PairedLaCs ontolex:isDenotedBy ?AnswerLemma .
}
```

A few aspects of this query bear mention. Firstly, we use "DISTINCT" to ensure we don't have a long list of duplicate entries. Secondly, this query fully implements Lemma results, rather than just LaCs; its code may be re-used in other query examples as needed. Lastly, note that we don't use any SPARQL "FILTER" options, which means that "A to A" relations are possible. This reflects the Wordweb's design, where reflexive relations are allowed. If the user wants to avoid reflexivity, additional query restrictions should be used.

Relationals ("Skyldheiti")

Two Lemmas are considered Relationals if both are Pairs with the same, third Lemma.

In the original Wordweb, Relationals may be ordered in two ways: By the raw count of how many third Lemmas they're Paired with, or the ratio of how strongly those third Lemmas are Paired with them rather than other, unrelated Lemmas. Representing these two orderings as a bulletpoint model would likely be rather confusing, so we'll employ set notation instead:

The raw count of common partners ("fjöldi sameiginlegra félaga") is likely the easier of the two to represent and explain. If A is the Lemma we originally selected, C is a Lemma with which A has a Relational relationship, and the set Bx = {B1, B2, B3, ...} is the set of all Lemmas that have a Pair relationship with both A and C, then the raw count of common partners is the number of Lemmas in B, or |B|

Let's now look at the ratio of relational pairs ("hlutfall af parasamböndum skyldheitis"). As before, A is the Lemma we originally selected, C is a Lemma with which A has a Relational relationship, and the set Bx = {B1, B2, B3, ...} is the set of all Lemmas that have a Pair relationship with both A and C. Let's now define the set Dx = {D1, D2, D3, ...} as the set of all Lemmas with which C is a Pair. (Note that this implies Bx is a subset of Dx.) The ratio of the relational pairs that C has with A is then |B| / |D|

Let's show how we could produce these results through SPARQL. We'll do so in three queries. (As with other query languages, it is of course perfectly possible to merge multiple queries into a single one, but at the cost of readability.)

Let's first find the set of Bx. We'll assume the original Lemma was "hundur" ("dog"):

```
SELECT ?label WHERE {
    ?a rdfs:label "hundur"@en.
```

```
    ?a skos:related ?b.
    ?b skos:related ?c.
    ?c rdfs:label ?label.
    FILTER (?c != ?a)
}
```

This is sufficient if we're only looking for the raw count of common partners. Moving on to the ratio of relational pairs, let's say that these results - the entire set Bx - turned out to simply be {mús}. For every element in Bx, we'll need to run the following, which will give us its Dx:

```
SELECT ?x WHERE {
    ?s rdfs:label "mús"@en.
    ?s skos:related ?o.
    ?o rdfs:label ?x.
}
```

Let's say that for this particular Bx, the query returned the Dx {"köttur", "rotta"} (note that we could use "COUNT" to return numerical results if we're not interested in the Lemmas themselves but only the ratios). If we now want to determine which elements in Dx relate back to A ("hundur") through Pair relations, we can run the query:

```
SELECT ?x WHERE {
    ?s rdfs:label "mús"@en.
    ?s skos:related ?o.
    ?hundur rdfs:label "hundur"@en.
    ?o skos:related ?hundur.
    ?o rdfs:label ?x.
}
```

Neighbors ("Grannheiti")

Neighboring Lemmas also come in two types, though this time around there is a more explicit distinction between them.

The first type is a Neighboring Pair. Note that this is more akin to a simplified Relational than a regular Pair. If the original Lemma is A, then the Lemma C is a Neighboring Pair so long as there exists at least one Lemma B such that A and B are a Pair, and B and C are a Pair.

The second type is a Neighboring Phrase, and involves a phraseme (in the original Wordweb these are usually identified as "orðasamband" or "orðtak"). Say the original Lemma is A. The

Lemma C is a Neighboring Phrase if there exist the phrasemes B1 and B2 such that B1 contains A, B2 contains C, but B1 and B2 are otherwise identical. As a simple example, if A was "man", then a Neighboring Phrase C could be "woman" if there exist in the Wordweb the phrasemes "grown man" (B1) and "grown woman" (B2).

The SPARQL query for Neighboring Pairs (for "hundur") is shown below:

```
SELECT ?label WHERE {
   ?a rdfs:label "hundur"@en.
   ?a skos:related ?b.
   ?b skos:related ?c.
   ?c rdfs:label ?label.
   FILTER (?c != ?a)
}
```

We will forego showing queries for Neighboring Phrases, as these could be highly complex and depend somewhat on the parameters of each particular query. For the interested reader, we suggest the use of "FILTER" and "CONTAINS" for simpler queries, while more complicated ones may require either "BIND", or the use of regular expressions.

5. Lemma composition and special characters

The last major feature concerns the structure of polylexical Lemmas, or more specifically, how and why they are split into subparts.

OntoLex supports numerous ways for dividing polylexical entries into segments. Generally speaking, the chosen method for this kind of division depends on two factors: The type of entries a system contains, and the purpose of the division.

The first factor is quite straightforward. A system's basic elements can only be parsed and divided in a certain specific way if the elements include the necessary data, and if the data is formatted in a manner that's amenable to being divided with the chosen methods. Parsing lines of text to find, say, each line's subject-verb-object, would become rather complicated if the lines weren't fully formed sentences, and likewise if data formatting made it difficult to differentiate between particular words. With the Wordweb, we've focused on the kinds of segmentation methods that clearly apply to the data - and the intent behind that data.

The second factor ties in to that intent. The Wordnet is intended to display semantic relations between lemmas. As a result, one of the most obvious applications for segmentation is to help elicit even more of these relations; while another clearly beneficial application would be to clarify even further the shades of meaning between lemmas. These applications tie in with the Wordweb's original purpose. It might very well be possible to implement a number of other types of subdivisions, mostly based on grammar and syntax, but while these might be useful for specific kinds of research, they don't have the same obvious connection to the system's original purpose as the two aforementioned applications.

In chapter 5.1 we look at the main way with which we subdivide Lemmas. In subsequent chapters, we look at the two types of special characters, angle brackets and square brackets, and explain how they were encoded in the new Wordnet.

5.1 Components

The first of those two applications, subdivision to help elicit more relations, is a new and powerful feature of the new Wordweb. Simply put, polylexical Lemma are now divided into OntoLex "components".

Components may have many uses, depending on the system in question, and can theoretically hold as many or as few sub-elements as needed. In our case, we use them on polylexical Lemmas to represent every individual word that also exists, somewhere else in the Wordweb, as its own separate monolexical Lemma. The component links to that monolexical entry, and also stores information on that particular word's morphosyntactic form in the polylexical one.

This new functionality allows for a much greater and more thorough level of interconnectedness in the Wordweb. In the original system, single words from polylexical Lemmas could be found only through a separate search for that particular word. In our implementation, the Lemmas can be traversed back and forth like a web.

(As per our earlier discussion of OntoLex's capabilities for element subdivision, it should be noted that OntoLex components may also be used to represent syntactic roles, both for single and multiple words. This kind of functionality was not on offer in the original Wordweb and was thus not given priority during the development process. However, the new format would absolutely support this kind of information, for example if it were added through a language parser.)

5.2 Special characters

There are two types of special characters from the original Wordweb that needed to be designed and implemented in the new system: Angle brackets (< >) and square brackets ([ ]).

In the original, both types of brackets were used as a wildcard of sorts, standing in for a subset of other words that share certain characteristics. With odd brackets it was usually a grammatical category and certain morphosyntactic properties, most often pronouns: "fljúga á <hann, hana>" ("attack <him, her>"). With square brackets it was more often related to meaning, and tends to play (very informally) the role of an intensional definition: "dökkrauður [hestur, kýr]" ("dark-red [horse, cow]").

These kinds of stipulations may be helpful to certain readers, but they are not simple nor easy to implement in machine-readable format. They rely strongly on implied meaning and appropriate use, which is nontrivial to codify, and they're being used in a system whose design is in good part based on representing various degrees of meaning without explicitly defining them or providing examples. However, they can unquestionably be helpful, and we thought it important not to lose them entirely during the conversion.

5.2.1 Angle brackets (< >) and Arguments

Angle brackets have been encoded as OntoLex "arguments". In our implementation, an OntoLex argument may be thought of as a kind of OntoLex component that has a special property - the wildcard described earlier - and that may contain one or more words.

There is no ideal OntoLex feature for this sort of thing, nor is there a fully apt linguistic term that explicitly defines it. OntoLex arguments are usually applied as specific argument subtypes (adjuncts, clausal, etc.) when describing OntoLex "syntactic frames". After some discussion, we decided that they'd be a close enough approximation to the Wordweb's angle bracket-entities.

If a Lemma in the original Wordweb contained angle brackets, the brackets themselves have now been removed from its written representation, and the words inside the brackets are grouped together as a single entity under the "argument" banner. The single words inside that argument are then, in turn, divided into individual Lemma-linked components like usual; as are the other words in the polylexical Lemma that were not inside angle brackets.

5.2.2 Square brackets ([ ]) and Usage

As described earlier, square brackets serve a slightly different role than that of angle brackets. They're more akin to suggestions pertaining to how the Lemmas' meanings should be interpreted.

As such, words enclosed in square brackets don't syntactically belong to the Lemma in question; they might be better thought of as footnotes. This is reflected even in the original Wordweb, where words in square brackets tend to be ignored when it comes to semantic relations. A Lemma of the form "x [y]", and a separate bracket-less Lemma of the form "x", will will often be filed under the same Concepts. The "x [y]" Lemma will rarely, if ever, belong to a Pair; and in general will have an absolute minimum of semantic relations other than pure synonymy and antonymy. Moreover, in many cases, some of the words inside square brackets are morphosyntactically at odds with the rest of the Lemma's contents. An adjective outside the bracket might be in the male gender, while a noun inside the bracket - which should modify the adjective - may be female or neutral, making the Lemma in its entirety grammatically incorrect (and adding a layer of complexity if we were to simply remove the brackets and let the words stand on their own, as we did with the angle brackets). All in all, it is clear that the bracket contents primarily serve to add a new meaning variant, and not as a syntactically meaningful part of the Lemma.

Thankfully, meaning variants are well-established in OntoLex as Lexical Senses. Since OntoLex offers us a pretty good implementation for the kind of feature that square brackets represents,

and since the words inside those brackets effectively act as meaning-related and syntactically disposable properties, we opted to remove the brackets and their contents from the Lemma's texts, and recreate the bracket contents instead as a brand new Lexical Sense. To that sense we add a special OntoLex feature called "usage".

Usage is a directive to the user on how that sense should be applied, and is generally applied when there is a difference in meaning that may not be covered by the connection between a Lexical Sense and its ontological counterpart. In fact, OntoLex intentionally refrains from being too explicit with how it should be employed, leaving it up to the designer instead.

What we've done, therefore, is simply add the bracket contents into a "usage" label on a newly created Lexical Sense. We then look at the remainder of the original Lemma, shorn of the bracket contents. If that remainder already exists as a Lemma of its own - which it almost always does - we simply attach the new Lexical Sense to that Lemma. If such a Lemma does not exist already, we create it from scratch.

(Here, again, we see the benefit of LaCs. Without them, we would have no ontological counterpart for this new Lexical Sense - since the Sense wasn't part of our original system - and thus no way of actually creating the Sense, since by design it must be connected to *something* in our ontology. As it stands, we simply create a new LaC that corresponds to this new Lexical Sense.)