You'll find here the **RCxn structure & architecture**, a documentation of the RDF implementation of the database on which the research constructicon will be built. The documentation relative to the **RCxn design** can be found here: Design RCxn

Note: orange boxes describe the current implementation, while plain text contributes to the discussion around choices made for the implementation.

Ontologies/Modules

Ontologies should be independent of each other (we have more chances that some people will use them?). This is not completely the case for now, since some classes (like Constructions) are used by different modules.

- Constructicon-specific module
    - metadata in a broad sense: Title, annotator, etc.
    - the URIs that have to do with constructions elements and slots (including Gesture, IS)
    - Meaning of the construction (metaphorical meaning, image-shema recruited, etc.)
    - Reference and Literature
    - Metadata
- Research (Project, Member, Research Question)
- Evidence (Corpus/Experiments results, Visualizations)
- Links
- Language
- Examples (Translation, Transliteration, Transcription, Glosses, Source of the example)
- Comparative concepts

Question

Are Findings part of "Research and researchers" or "Evidence"?

Are frames their own ontology or part of comparative concepts? (the former seems more appropriate)

MODULE Constructicon (rcxn)

This module borrows from Dublin Core (dc; description), RDF Schema (rdfs; label), Schema Datatype (xsd; date), Olia (olia; DiscourseFeature & oliatop; SemanticRole) and the research module (rsrch; Project).

Labels for properties can be used directly in the online interface as a label for the corresponding field.

I plan to implement the Abox as follows:
- Constructions stored under cx.ttl
- Syntactic functions stored under synfunc.ttl (not done yet; for the moment syntactic functions are literal objects)

Class rcxn#Construction	
description	Construction; linguistic sign with a form component and a meaning component. A network of constructions builds a Constructicon.

Title

Property rcxn#hasTitle	
description	Property that relates a construction to its name.
domain (subject)	rcxn#Construction
range (object)	Literal
label	Title (en)

If we decide that all constructions should follow the pattern “X Y construction” where X is the language and Y the name of the construction, we should make sure that X and “construction” are added automatically.

For the moment, the IRI of the construction is based on the title, but we could run into duplicates. I think that the IRI for the construction should be a random name (e.g., random number). This would help to create IRI for slots, examples, etc. that are kept relatively short.

Semantic function of the construction

Class rcxn#ConstructionMeaning	
description	A construction has both form and meaning. ConstructionMeaning refers to the meaning of a construction.

The URI for construction meaning is created by adding "_Meaning" to the URI of the construction.

Property rcxn#hasConstructionMeaning	
description	Property that relates a construction to its meaning component (an object of type ConstructionMeaning).
domain (subject)	rcxn#Construction
range (object)	rcxn#ConstructionMeaning

Property rcxn#hasMeaning	
description	Property that relates the meaning component of a construction to the corresponding plaintext description of this meaning.
domain (subject)	rcxn#ConstructionMeaning
range (object)	Literal
label	Meaning of the construction (en)

Lemon has a property lemon:sense, that links a lexical entry to a lexical sense. We could define lemon:sense as a subproperty of rcxn#hasConstructionMeaning and lemon:LexicalEntry as a subclass of rcxn#Construction. We could even use the lemon terminology in the specific case of lexical entries, but I doubt it is worth the trouble.

It would make sense to have a mechanism that links elements of the semantic function of the construction to the semantic role of individual slots in the construction.

Example

Cx ADJ-Agreement
Semantic function: Ascribing ATTRIBUTE to ATTRIBUTEE
Semantic role of slots: (1) ATTRIBUTE, (2) ATTRIBUTEE

However, I can’t think of a way to do it that is user-friendly. This is probably not possible if the object of rcxn#ConstructionMeaning is a Literal object.

Property rcxn#usesImageSchema	
description	Relates the meaning component of a construction to the cognitive pattern that it is based on (i.e., containment or path) and that helps convey meaning through spatial or experiential metaphors.
domain (subject)	rcxn#ConstructionMeaning
range (object)	Literal
label	Image-Schema

So far, Image-Schema is just plain text. It would be nice to create a collection of image-schema, which then would be objects of the class Image-schema (and no Literal objects).

Slots

Class rcxn#Slot	
description	Construction elements in the construction. They encompass both slot elements, which are positions where specific lexical items or phrases can be inserted, and lexically defined elements, which are fixed words or phrases that contribute to the construction's specific form and meaning. In constructicography, construction elements are linked together by "sequential" links.

Class rcxn#SlotMandatory	
description	Non-optional construction elements that need to be realized.
subclass of	rcxn#Slot
disjoint with	rcxn#SlotOptional

Class rcxn#SlotOptional	
description	Optional construction elements; the meaning of the construction is not dependent on its realization.
subclass of	rcxn#Slot
disjoint with	rcxn#SlotMandatory

The URI for elements is based on the URI of the construction, followed by a _ and the number of the sot (1, 2, 3 etc.). We still need to integrate embedded elements into the system (i.e., element 1 consisting of element 1.1 and 1.2).

Property rcxn#hasSlots	
description	Relates a construction to the sequence of construction elements that constitutes it.
domain (subject)	rcxn#Construction
range (object)	Sequence

RDF proposes different types of groups:
- bags: unordered
- sequence: ordered
- collection/list: ordered and exhaustive (no element can later be included), based on a first/rest recursive structure.

We are using sequences because the elements should be presented in a fixed order on the website, even if the linearization is free in the construction. Linearization (word order) is considered separately.

The URI for the sequence is the URI of the construction followed by “_slots”.

Class rcxn#SlotForm	
description	Formal features of a construction element.

The URI for the form of the construction element is the URI of the element followed by “_Form”. Such a URI is only created in the online form if the user specifies the phonology, root, stem, surface form, translation, transliteration and/or morphosyntactic form (see following properties).

Property rcxn#hasSlotForm	
description	Relates a construction element to its formal features.
domain (subject)	rcxn#Slot
range (object)	rcxn#SlotForm

Property rcxn#hasPhonology	
description	Phonological form of the construction element.
domain (subject)	rcxn#SlotForm
range (object)	Literal
label	Phonology

Property rcxn#hasRoot	
description	Describes the form of a construction element as based on a certain root.
domain (subject)	rcxn#SlotForm
range (object)	Literal

label	Root (en)

Property rcxn#hasStem	
description	Describes the form of a construction element as based on a certain stem or lemma.
domain (subject)	rcxn#SlotForm
range (object)	rcxn#Construction
label	Stem / Lemma (en)

Property rcxn#hasSurfaceForm	
description	Describes the form of a construction element as having a specific surface form.
domain (subject)	rcxn#SlotForm
range (object)	Literal
label	Surface form (en)

Property rcxn#hasSyntacticForm	
description	Identifies the form of the element with that of a construction: typically, it would be a morphosyntactic category (a type of morpheme, a part of speech, a phrase type, a clause type).
domain (subject)	rcxn#SlotForm
range (object)	rcxn#Construction
label	Syntactic form (en)

As a corollary to hasSyntacticForm, I found necessary to add a link between construction that relates a construction back to all construction that use it (for example, the NP construction will potentially be used by many constructions):

Property rcxn#elementOf	
description	Relates a construction to the constructions that it is a part of. If the form of one construction element of A is identified as B (property hasSyntacticForm), then B is an element of B.
domain (subject)	rcxn#Construction
range (object)	rcxn#Construction
label	Construction Element of (en)

Property rcxn#hasSyntacticFunction	
description	Relates the construction element to its function. Typical functions are “subject”, “complement”, “modifier” etc.
domain (subject)	rcxn#Slot
range (object)	Literal
label	Syntactic function (en)

Property rcxn#hasSemanticContribution	
description	Describes the meaning (e.g., semantic role) of the construction element in the construction.
domain (subject)	rcxn#Slot

Property rcxn#hasSemanticRole

description	Describes the semantic role of the construction element in the construction. The semantic roles are defined by the OLiA ontology.
subproperty of	rcxn#hasSemanticContribution
range (object)	oliatop:SemanticRole
label	Semantic role (en)

Property rcxn#hasOtherSemanticContribution	
description	Describes the meaning of the construction element in the construction that cannot be captured by the semantic roles of the OLiA ontology.
subproperty of	rcxn#hasSemanticContribution
range (object)	Literal
label	Semantic contribution (en)

Property rcxn#hasSemanticProperty	
description	Describes any loosely defined semantic category that the construction element (typically a word) should belong to. E.g. “the noun has a generic meaning” or “the verb denotes a destructive event”
domain (subject)	rcxn#Slot
range (object)	Literal
label	Semantic property (en)

For information structure, we decided to diverge from OLiA. However, the following implementation is very simple, OLiA is actually much more detailed and accurate. We may want to adopt their model in the future.

Class rcxn#informationStructure	
description	

Class rcxn#Focus	
description	Focus indicates the presence of alternatives that are relevant for the interpretation of linguistic expressions. (Féry et al. 2007) That part of an expression which provides the most relevant information in a particular context as opposed to the (not so relevant) rest of information making up the background of the utterance. Typically, focus on a subexpression indicates that it is selected from possible alternatives that are either implicit or given explicitly, whereas the background can be derived from the context of the utterance. (SFB632 Guidelines) Foci are classified here according to their discourse function (independently from their structural realization).
is defined by	olia#Focus
equivalent to	olia#Focus
subclass of	rcxn#informationStructure
label	Focus

Class rcxn#Topic	
description	The topic constituent identifies the entity or set of entities under which the information expressed in the comment

	constituent should be stored in the CG content. The notion of topic is best understood as a kind of address or file card which specifies the individual or set about which the remainder of the sentence makes a comment (see Reinhart 1981 for such a concept of topicality). It has no truthconditional effect except that it presupposes the existence of that individual. In this sense, the complement of ‘topic’ is ‘comment,’ which can itself be partitioned into a focused and a backgrounded part. Sentences usually have only one topic, but can also have none, or more than one. Following Jacobs (2001), topics can be aboutness or frame-setting topics, and the means to express a topic in the grammar can be pinpointed rather precisely in terms of which syntactic and intonational preferences the topic displays, at least in an intonation language. However, according to Féry’s theses, none of these properties are definitional for topic. Rather they express preferences as to how a ‘good’ topic has to be realized (see also Jacobs 2001 for a similar view). (Féry et al. 2007)
is defined by	olia#Topic
equivalent to	olia#Topic
subclass of	rcxn#informationStructure
label	Topic

Class rcxn#Background	
description	The Background is in complementary distribution with the Focus. Typically, the background is discourse-given and can be derived from the context of the utterance. It is the not so relevant information of the utterance.
subclass of	rcxn#informationStructure
label	Background

Class rcxn#ISComment

description	The Comment is in complementary distribution with the Topic. It is what is said about the topic, the main predication of the utterance.
subclass of	rcxn#informationStructure
label	Comment

Property rcxn#hasIS	
description	The construction element is part of the Focus or Background / Topic or Comment of the full utterance.
domain (subject)	rcxn#Slot
range (object)	rcxn#informationStructure
label	Information Structure

Metadata

Class rcxn#Metadata	
description	Non-linguistic information on the construction entry, such as the persons involved in the creation of the entry or the sources.

The URI for the metadata of the construction is based on the URI of the construction followed by “_MD”.

Property rcxn#hasMetadata	
description	Relates a construction to its metadata.

domain (subject)	rcxn#Construction
range (object)	rcxn#Metadata

Property rcxn#annotator	
description	Relates a construction entry’s metadata to the person who created it.
domain (subject)	rcxn#Metadata
range (object)	rsrch#Member
label	Annotator (en)

Property rcxn#creationDate	
description	Indicates the date where the construction entry was created and links it to the construction’s metadata.
domain (subject)	rcxn#Metadata
range (object)	xsd#date
label	Creation date (en)

References & Literature

Dublin Core (db) is the ontology typically used for RDF implementations of publications.

References (i.e., the publication that the construction is based on)

I contacted Susanne and Emilia, who need to gather the bibliography of the RTG, to join forces. For the RTG-website, the university wants them to use CRIS. More about it later…

Literature (i.e., relevant literature on the topic)

Building a Zotero bibliography could be a way to gather these references. Export in db is possible (maybe even automatizable?).

Similar cxs in other Cxns [o]

So far, it is stored under RDFS.seeAlso. Ontolex-Lemon has a property called lemon:reference. It is used as the object of a definition, i.e. to link a lexical entry to a BDpedia entry.

MODULE Casa (casa)

Because the Research Constructicon project is tightly linked to the CASA Project (https://constructicon.de/), and because the entries of CASA are imported in the Research Constructicon, we needed to adopt some of their terminology.

Property casa#hasCasaSyntacticFunction	
description	Relates the construction element to its function, using CASA's methodology. Typical functions are "potential subject", "Obj" (object), "Attr" (attribute) etc.
domain (subject)	rcxn#Slot
range (object)	Literal
label	Syntactic function (CASA terminology) (en)

MODULE Research (rsrch)

This module borrows from Dublin Core (dc; description), RDF Schema (rdfs; label) and Friend-of-a-friend (foaf; Person, Project).

I implemented the Abox as follows:
- researchers, as well as their projects, research questions and findings are in membr.ttl (prefix membr)

Class rsrch#Member	
description	Member of the research group and/or contributors to the Constructicon.
subclass of	foaf:Person

Class rsrch#Researcher	
description	PhD student researcher working in the research group.
subclass of	rcxn#Member
disjoint with	PI, PostDoc

Class rsrch#PostDoc	
description	Post-doctoral researcher working in the research group.
subclass of	rcxn#Member
disjoint with	Researcher

Class rsrch#PI	
description	Researchers supervising or otherwise involved in a research project.
subclass of	rcxn#Member
disjoint with	Researcher

The IRI for the researcher is based on the student’s last name. I left out diacritics (e.g., Führer is membr#Fuhrer, and not Fuehrer) and changed “ß” to “ss” (i.e., membr#Kessler for Florian Keßler).

Research Question

Class rsrch#Project	
description	A research question, i.e., a concise inquiry that defines the academic focus of a researcher. Within the terminology of the Research Training Group, the main research question of a PhD candidate for their thesis is called a “project”.
subclass of	foaf:Project

For the IRI of the student’s projects, instead of “Project 1”, “Project 2” etc., I chose to name the project after the student’s last name (e.g., prjct#Project_Rastegar). The project is then labeled with the title of the project (property is rdf:label; object is a rdf:Literal; language code is included). This has some advantages: the label can be translated into different languages, and extra-RTG students can be easily included, as well as any researcher in the FAU. The researcher and its project are linked via foaf:currentProject (e.g., rcxn#Rastegar foaf:currentProject rcxn#Project_Rastegar).

Property rsrch#hasResearchQuestion	
description	Property that breaks down a project or research question and relates it to smaller research questions.
domain (subject)	rsrch#Project
range (object)	rsrch#Project
Characteristics	Transitive, Asymmetric

Property rsrch#projectName

description	Property that relates the URI of a project to its exact label.
domain (subject)	rsrch#Project
range (object)	Literal

Findings

Research questions lead to findings. Findings are based on research data on the one hand and constructions on the other hand.

Class rsrch#Finding	
description	Research on a research question leads to one or several finding(s). They are statements, answers to the research question.

Property rsrch#hasFindings	
description	Property that links a research question to the finding(s) answering this research question.
domain (subject)	rsrch#Project
range (object)	rsrch#Finding
Characteristics	Asymmetric

Property rsrch#basedOn	
description	Property that relates a finding to the evidence for the finding: either research data or construction(s).

domain (subject)	rsrch#Finding
range (object)	cx#Construction
Characteristics	Asymmetric

MODULE Language

We make here a distinction between “Macrolanguages” and “Language Variety”.

Macrolanguages are defined by the norm ISO 639-3 (https://iso639-3.sil.org/sites/iso639-3/files/downloads/iso-639-3_Name_Index.tab). Some of the languages defined by the norm ISO 639-3 are mapped to a macrolanguage (https://iso639-3.sil.org/sites/iso639-3/files/downloads/iso-639-3-macrolanguages.tab), in which case there are considered a language variety in our terminology. For example, “Literary Chinese” (code lzh) is a variety of “Chinese” (code zho). Languages defined by the norm ISO 639-3 that have no Macro-languages are considered in our terminology as being a macrolanguage.

Contributors of the RCxn can create language varieties as fine grained as they wish, but need to range them under a language defined by the norm ISO 639-3. By doing so, they can specify any of the following:
- (macro-)language
- time
- dialect area
- modality (written/spoken)
- genre (e.g., legal, informal, etc.)

Class lg#lg	
label	Language
description	Language constructional space that the construction is a part of.

Class lg#macrolanguage

label	Macrolanguage
defined by	ISO 639-3
subclass of	lg:lg

Class lg#variety	
label	Language Variety
description	Language constructional space that the construction is a part of. It contains indications about the language name, but also register, modality, dialectal area and temporality when relevant for the description of the construction.
subclass of	lg:lg

Property lg#isVarietyOf	
description	Property that relates a more specific language variety to a more generic language (language variety or macrolanguage).
domain (subject)	lg#variety
range (object)	lg#lg

Property lg#partOfLanguage	
description	Property that relates a construction to its language.
domain (subject)	rcxn#Construction
range (object)	lg#lg

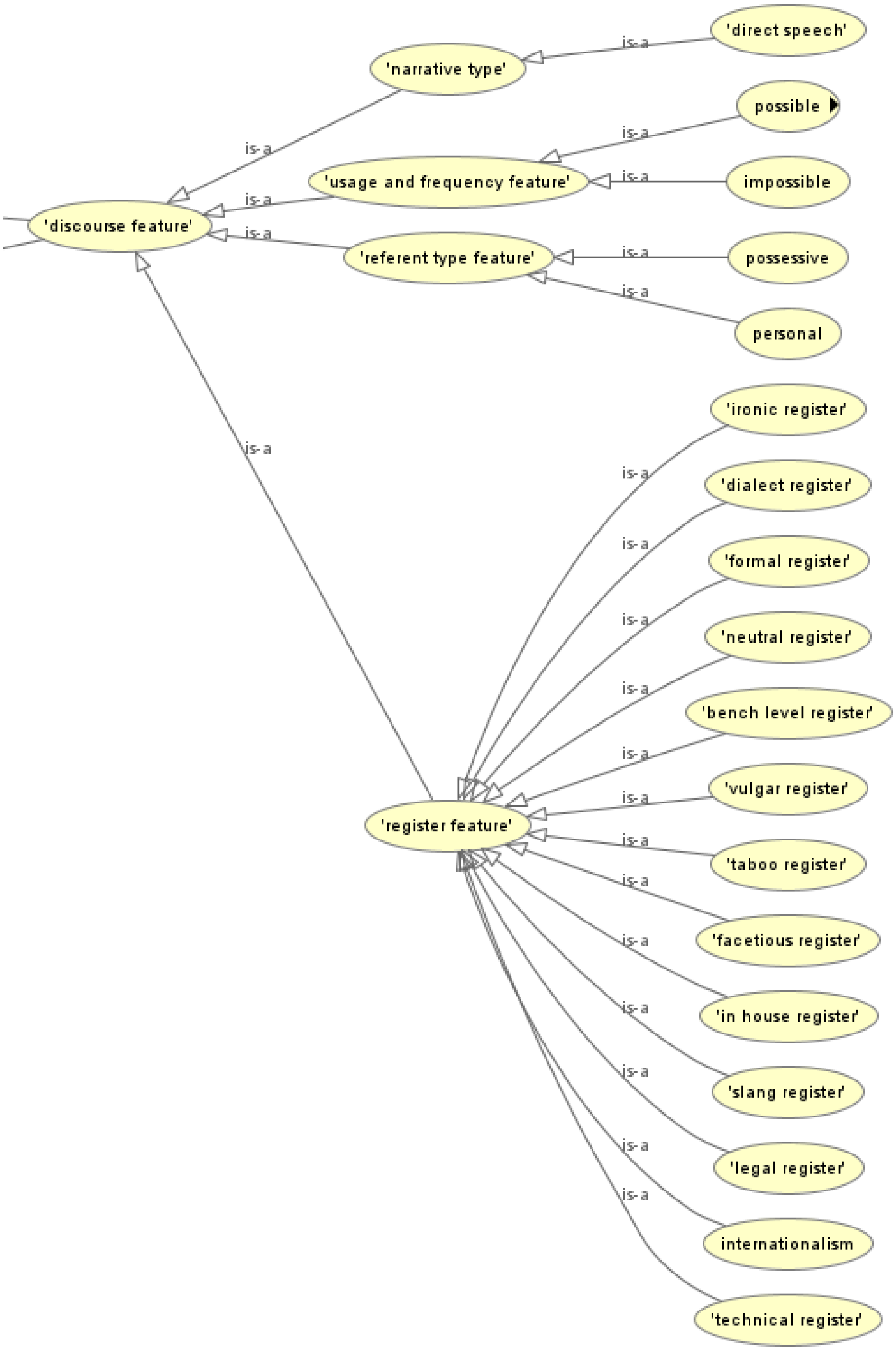
Our hierarchy of languages is implemented as individuals directly in the ontology. The IRI is the ISO code when available. For example, German is lg#deu:

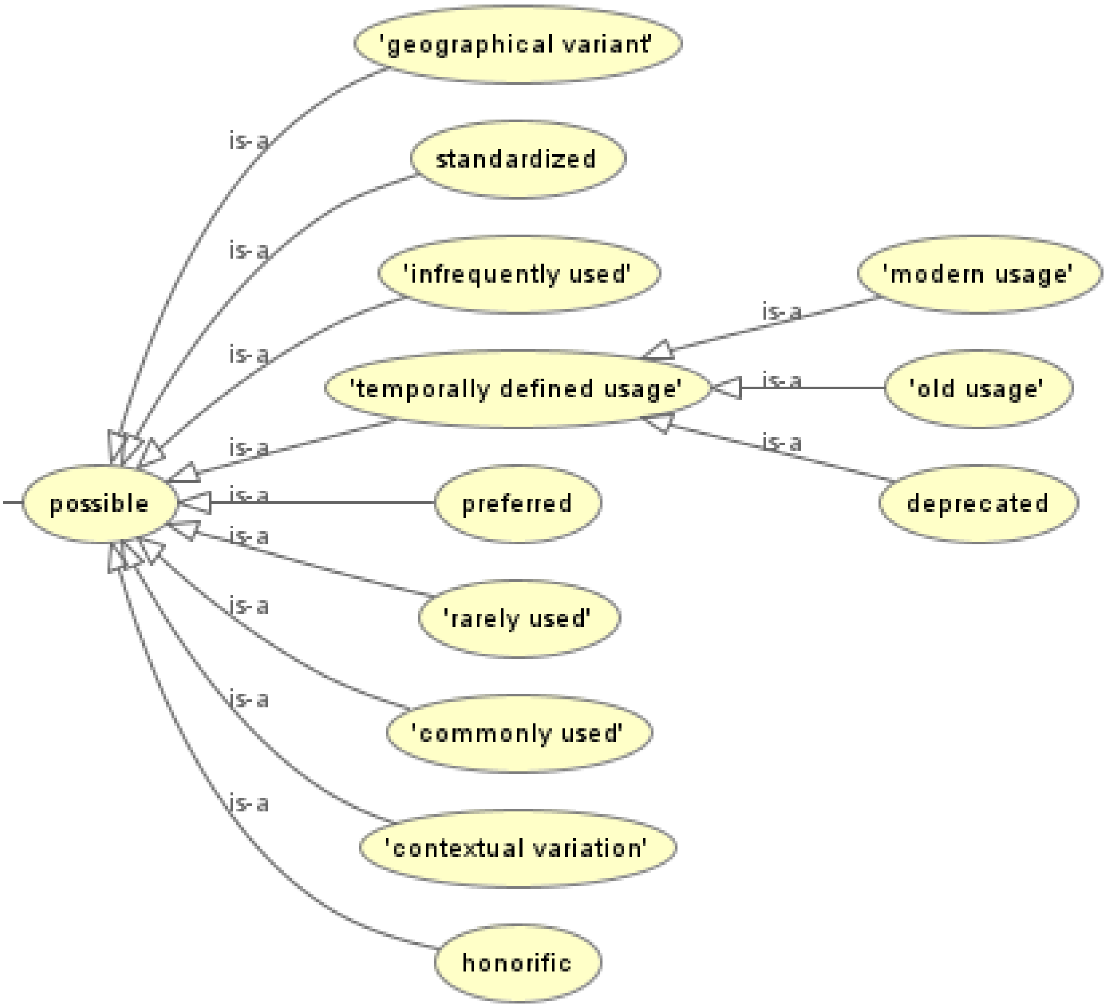
```
<!-- https://bdlweb.phil.uni-erlangen.de/RCxn/ontologies/lg#deu -->

    <owl:NamedIndividual
rdf:about="https://bdlweb.phil.uni-erlangen.de/RCxn/ontologies/lg#deu">
        <rdf:type
rdf:resource="https://bdlweb.phil.uni-erlangen.de/RCxn/ontologies/lg#macrolangua
ge"/>
        <rdfs:label xml:lang="en">German</rdfs:label>
    </owl:NamedIndividual>
```

It would be desirable to link the languages in our hierarchy to other existing ontologies. OLIA has a register hierarchy. For time frames, Dublin Core refers to the DCMI Period Encoding Scheme (https://www.dublincore.org/specifications/dublin-core/dcmi-period/).

Olia’s hierarchy:


'discourse feature'
'narrative type'
'direct speech'
'usage and frequency feature'
possible
impossible
'referent type feature'
possessive
personal
'register feature'
'ironic register'
'dialect register'
'formal register'
'neutral register'
'bench level register'
'vulgar register'
'taboo register'
'facetious register'
'in house register'
'slang register'
'legal register'
internationalism
'technical register'
is-a



MODULE Examples

~~Property rexn#hasTranslation~~	
~~description~~	~~Gives a translation of the form of a construction element. It is used mostly to translate the root, stem, lemma or surface form in languages that differ from the language of the interface.~~
~~domain (subject)~~	~~rexn#SlotForm~~
~~range (object)~~	~~Literal~~
~~label~~	~~Translation (en)~~

- property hasTransliteration, domain is [slot form & examples] and range is a literal

MODULE Links

The ontology for links is presented in the following publication:
Winckel, Elodie. 2025. Defining relationships in the constructional network: A Semantic Web ontology for Construction Grammar. *Lexicographica* 41(1). 299–317. https://doi.org/10.1515/lex-2025-0012.

Since this first version 1.0, I added this property (which is therefore part of the 1.1 version):

Property links#isMetaphoricalExtensionOf	
description	Links back the metaphorical extension to the construction with literal meaning (see metaphorical link).
subproperty of	links#LanguageInternalLink
label	is a metaphorical extension of

MODULE Evidence

I implemented a prototype. The ontology does not exist yet.

The prototype includes classes:

- Study
- Study type
  - corpus
  - behavioral
  - brain imaging
  - LLM
- Study target

    - adults
    - children
    - DLD
    - written
    - spoken
    - multimodal

The prototype includes properties:

- hasTitle
- basedOnStudy ; revelantFor
- Visualization ; belongsTo
- dataRepository
- Summary
- target
- studyType