Tutorial: Generating text from ontologies in multiple languages

C. Maria Keet[1] and Zola Mahlaza[2]

[1] University of Cape Town, South Africa, mkeet@cs.uct.ac.za
[2] University of Pretoria, South Africa, z.mahlaza@up.ac.za

Joint Ontology Workshops 2022, 15-19 August 2022

Outline

Outline

Ontology verbalisation: The Idea

- Render the axioms from the ontology as a natural language sentence

Ontology verbalisation: The Idea

- Render the axioms from the ontology as a natural language sentence
  - Maybe sometimes a *pseudo*-natural language sentence (i.e., not 100% grammatically correct, but good enough)

Ontology verbalisation: The Idea

- Render the axioms from the ontology as a natural language sentence
  - Maybe sometimes a *pseudo*-natural language sentence (i.e., not 100% grammatically correct, but good enough)
- For some purpose; e.g.:
  - Make axioms accessible to non-logicians for validation
  - Generate answerable questions in some edTech tool
  - Input to an ontology-driven human-robot interaction module
- The kind of sentence. Mainly: statements or questions

Examples for knowledge-to-text

- Electronic health records and patient discharge notes generation
- Getting the relevant business logic into your app
- Querying the data with conceptual queries in OBDA
- And many other areas; e.g., question generation, intelligent textbooks, automation of language learning exercises

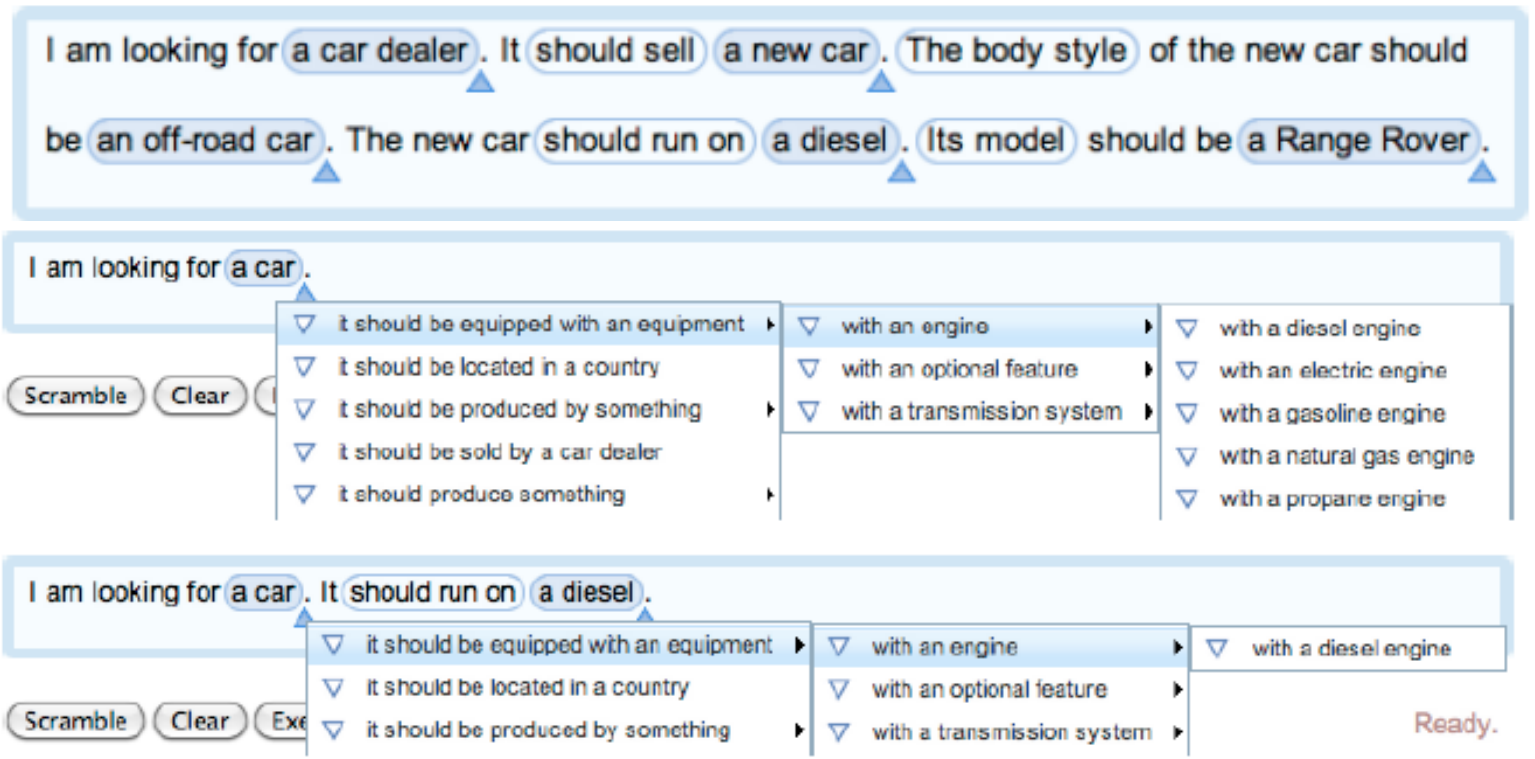
Examples for knowledge-to-text

- Electronic health records and patient discharge notes generation
  - e.g., SNOMED CT, OpenMRS localisation
  - "The patient has as symptom fever and dizziness"
  - "The patient must drink water when taking the pills"
    "If the patient takes the pills, then he must drink water"
- Getting the relevant business logic into your app
  - Requirements engineering, competency questions
  - "Which animals eat impalas?"
- Querying the data with conceptual queries in OBDA
  - "Show me all employees who are not working on a project"
- And many other areas; e.g., question generation, intelligent textbooks, automation of language learning exercises

Structured sentences – examples for knowledge-to-text

- Electronic health records and patient discharge notes generation
  - e.g., SNOMED CT, OpenMRS localisation
  - "The patient has as symptom fever **and** dizziness"
  - "The patient must drink water **when** taking the pills"
    "**If** the patient takes the pills, **then** he must drink water"
- Getting the relevant business logic into your app
  - Requirements engineering, competency questions
  - "**Which** animals eat impalas?"
- Querying the data with conceptual queries in OBDA
  - "Show me **all** employees who are **not** working on a project"
- And many other areas; e.g., question generation, intelligent textbooks, automation of language learning exercises

Example: Query formulation with Quelo [7]



This Tutorial

- Basic idea of CNLs and NLG with ontologies
  - The 'ingredients'
  - Variation in the components, with architectures
  - Example systems
- Hands-on exercises: build your own verbaliser (or part thereof)
- Advanced topics
  - Modelling effects
  - 'Localisation' and multilingualism
  - Research challenges

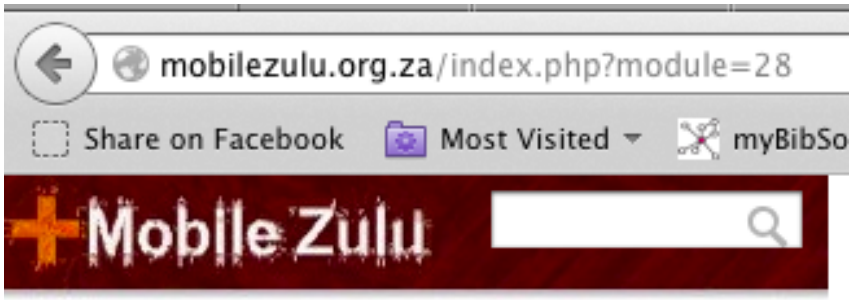
Outline

Outline

Short answer

- **C**controlled **N**atural **L**anguage: constrain the grammar or vocabulary (or both) of a natural language
- **N**atural **L**anguage **G**eneration: generate natural language text from structured data, information, or knowledge

Ex: S. Moolla's mobile healthcare app with **canned text**



Modules Translate About

Home » History » Cardiovascular History

Chest Pain

Have you had any recent pain in your chest? - Uke waba nobuhlungu esifubeni maduzane?

Does the pain radiate to your jaw, neck or arm? - Engabe ubuhlungu bakho bujikeleza emihlathini, emqaleni noma nasezingalweni?

Does anything precipitate or relieve the pain? - Ingabe ikhona into eyenza ubuhlungu buqhubeke noma eyehlisa ubuhlungu?

Dyspnoea

Are you breathless at any time? - Uke uphelelwe umoya

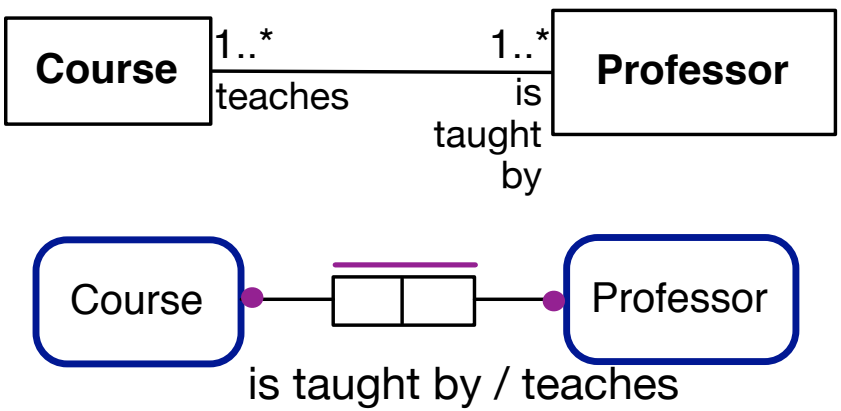
Ex: Avalanche bulletins with **canned segments** [23]

Segment 1	Segment 2	Segment 3	Segment 4	Segment 5
die Lawinen	**können**			gross werden.
nasse Lawinen		auch	oft	weit vorstossen.
diese		**{on_steep} Sonnenhängen**	weiterhin	bis in die aperen Täler vorstossen.
		in diesen Gebieten		**bis in tiefe Lagen vorstossen.**

Segment 3a	Segment 1	Segment 2	Segment 3b	Segment 4	Segment 5
	the avalanches	**can**			reach large size.
	wet avalanches		also	in many cases	reach a long way.
{on_steep} sunny slopes	they			as before	reach the bare valleys.
in these regions					**reach low altitudes.**

Fig. 2. Schema of a phrase in the source language German (above). {on_steep} mark a sub-segment with several further options. In this example, [blank] is one of the options in the third and fourth segment. In English, the order of the segments is different and segment 3 is split.

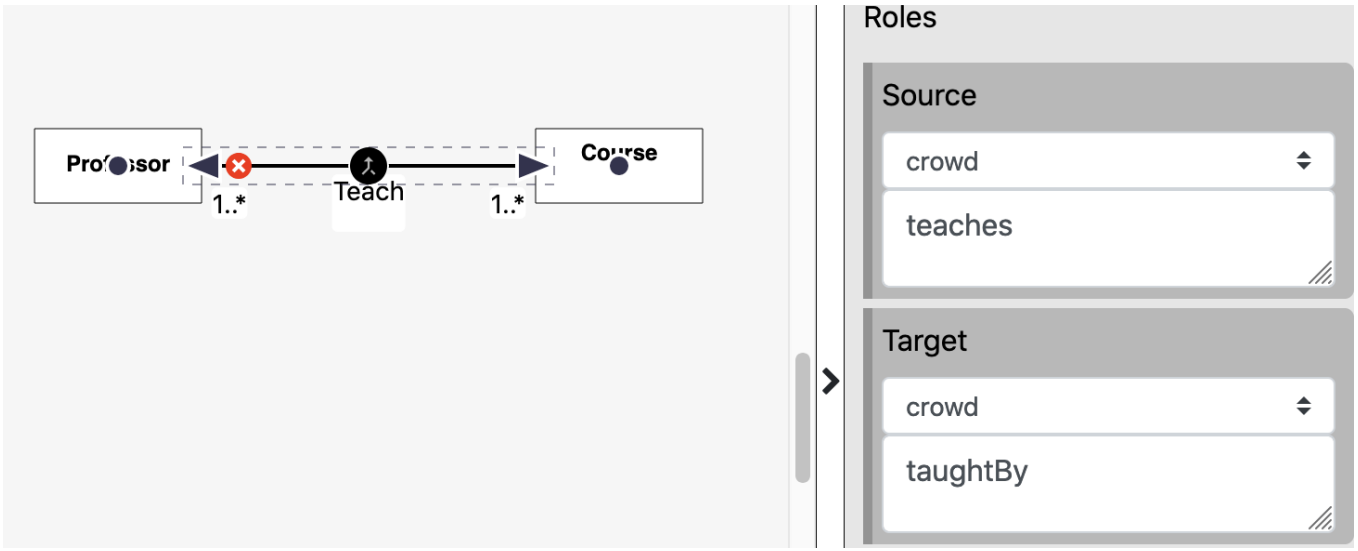
Ex: Business rules and conceptual data models with *static* templates



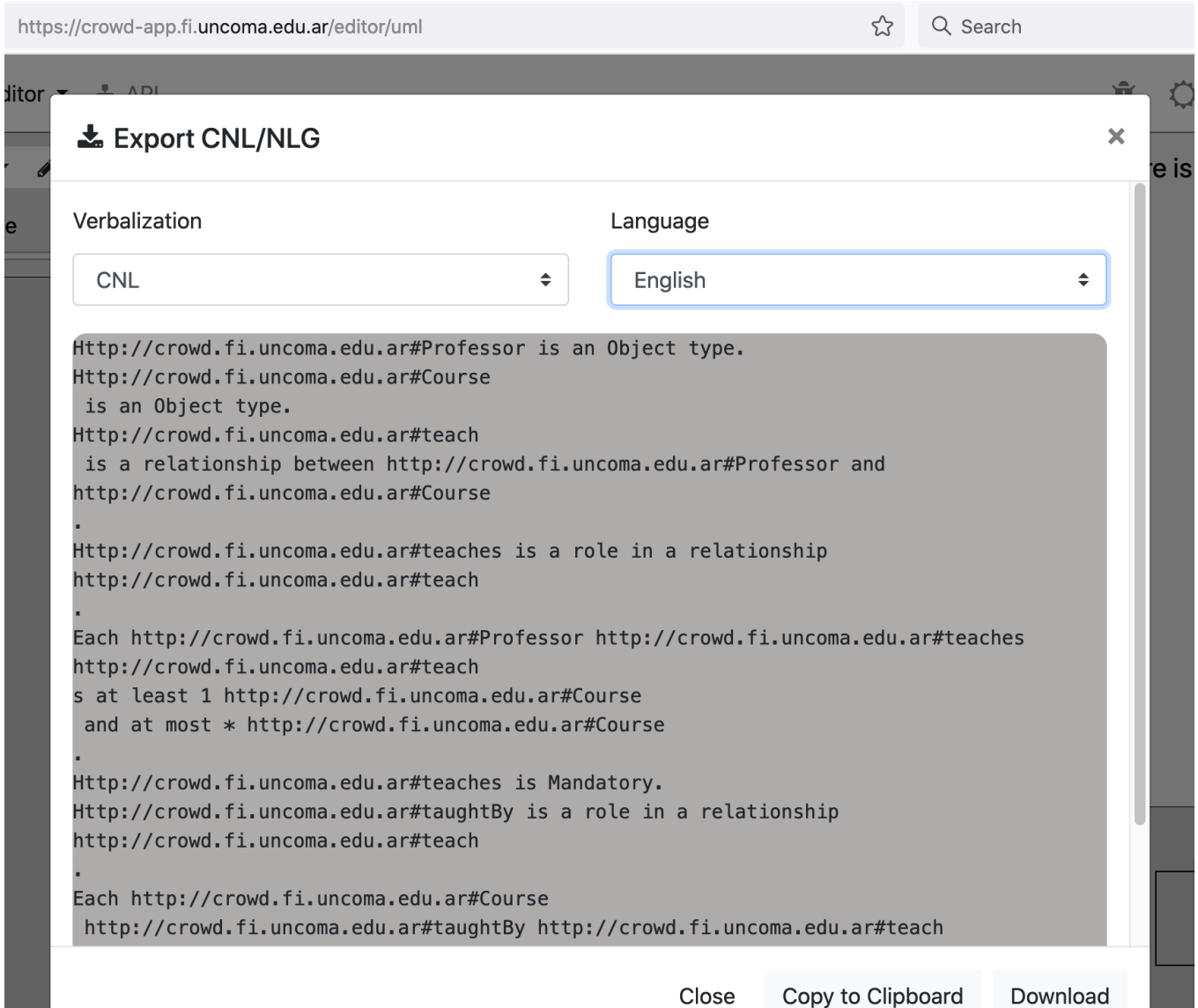
Each Course is taught by **at least one** Professor

Each Professor teaches **at least one** Course

Ex: crowd2.0 modelling tool (CDM & ontologies) [3]



Ex: crowd2.0 modelling tool (CDM & ontologies) [3]



Ex.: Mixing grammar with templates

- Idea: store the words in their base form with POS tag, specify in the 'template' what needs to be done with it, use a realisation engine to finalise the sentence
- e.g., yes/no pronomial or gender as variables to set

Ex.: Mixing grammar with templates

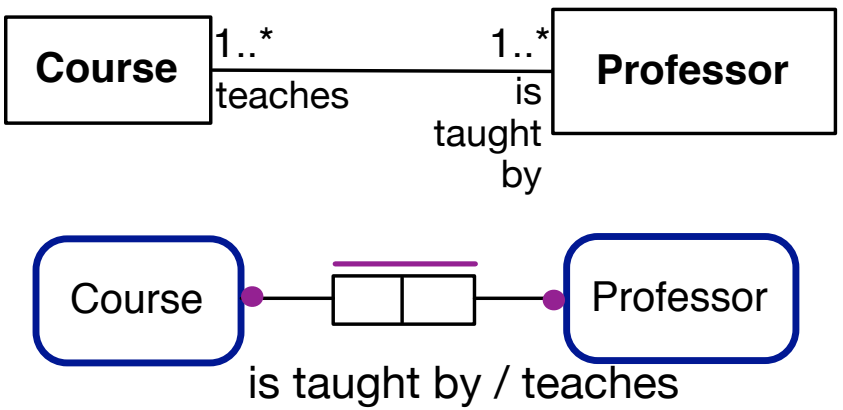
- Idea: store the words in their base form with POS tag, specify in the 'template' what needs to be done with it, use a realisation engine to finalise the sentence
- e.g., yes/no pronomial or gender as variables to set
- Same stems or words and core structure of the grammar-infused template, generate different sentences; e.g.:
  John eats an apple
  He eats an apple
  He eats it
  John eats it
  ...

NLG, principal approaches to generate the text

- ~~Canned text, with complete sentences (CNLs only)~~
- Canned segments to make a sentence (CNL mostly, not NLG)
- Templates (different types)
  - Mainly for English but also other languages
  - Hand-crafted ('old' approach) or ML/neural-based ('new')
- Grammar engines, such as [17], Grammatical Framework (`http://www.grammaticalframework.org/`), SimpleNLG [8]

See also: [18]

Business rules/conceptual data models and logic reconstruction



BR: **Each** Course is taught by **at least one** Professor

FOL: $\forall x\ (\text{Course}(x) \rightarrow \exists y\ (\text{is\_taught\_by}(x, y) \wedge \text{Professor}(y)))$

DL: $\text{Course} \sqsubseteq \exists\ \text{is\_taught\_by}.\text{Professor}$

- (i.e., a mandatory constraint / existential quantification)

Content stored in, e.g., some XML

```
...
<Predicate>
<Object_Role ID='ExEN:249' Object='Professor' Role='teaches'/>
<Object_Role ID='ExEN:250' Object='Course' Role='taught'/>
</Predicate>
...
<Constraint xsi:type='Mandatory'>
<Object_Role>ExEN:249</Object_Role>
</Constraint>
...
```

Example of static templates

Simple existential quantification ('mandatory constraints' in CDM parlance)

```
<Constraint xsi:type="Mandatory">
 <Text> -[Mandatory] Cada</Text>
 <Object index="0"/>
 <Text>debe</Text>
 <Role index="0"/>
 <Text>al menos un(a)</Text>
 <Object index="1"/>
</Constraint>
```

```
<Constraint xsi:type="Mandatory">
 <Text> -[Mandatory]  Each</Text>
 <Object index="0"/>
 <Text>must</Text>
 <Role index="0"/>
 <Text>at least one</Text>
 <Object index="1"/>
</Constraint>
```

that and more for 11 languages (ORM terminology): [11]

John eats apples – fancier templates

```
((template clause)
   (act 'eat')
   (agent ((template noun-phrase)
     (np-type PROPER)
     (head 'John')
     (gender MASCULINE)
     (pronominal NO)))
   (object ((template noun-phrase)
     (head 'apple')
     (pronominal YES))))
```

John eats it

```
((template clause)
   (act 'eat')
   (agent ((template noun-phrase)
     (np-type PROPER)
     (head 'John')
     (gender FEMININE)
     (pronominal YES)))
   (object ((template noun-phrase)
     (head 'apple')
     (pronominal NO))))
```

She eats an apple

NL Grammars, illustration (1/2)

$$
\begin{aligned}
\textit{Sentence} &\longrightarrow \textit{NounPhrase} \mid \textit{VerbPhrase} \\
\textit{NounPhrase} &\longrightarrow \textit{Adjective} \mid \textit{NounPhrase} \\
\textit{NounPhrase} &\longrightarrow \textit{Noun} \\
&\ldots \\
\\
\textit{Noun} &\longrightarrow \textit{car} \mid \textit{train} \\
\textit{Adjective} &\longrightarrow \textit{big} \mid \textit{broken} \\
&\ldots
\end{aligned}
$$

(and complexity of the grammar)

+ rules for verb tenses, pluralisation etc.

SimpleNLG tool [8] (2/2)

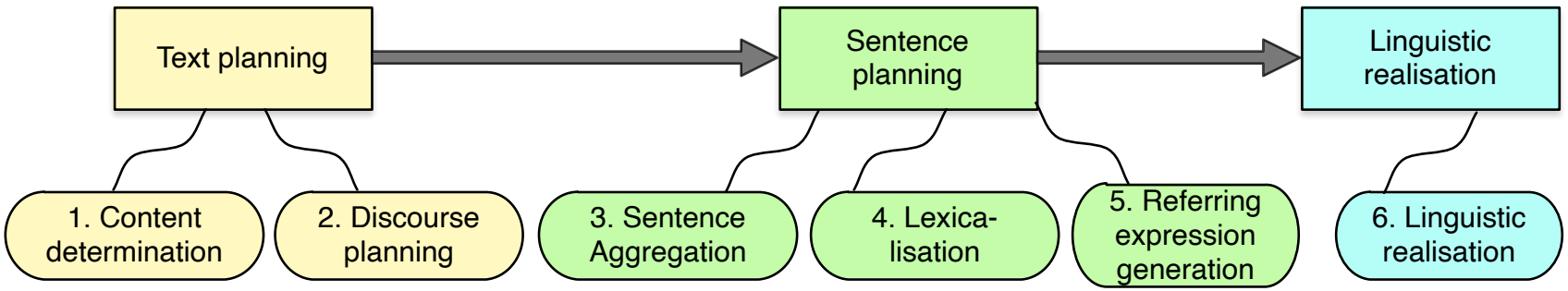
with grammars for EN, FR, ES, PT, NL, DE, and Galician

```
<Document>
  <child xsi:type="SPhraseSpec">
    <subj xsi:type="VPPhraseSpec" FORM="PRESENT_PARTICIPLE">
      <head cat="VERB">
        <base>refactor</base>
      </head>
    </subj>
    <vp xsi:type="VPPhraseSpec" TENSE="PRESENT" >
      <head cat="VERB">
        <base>be</base>
      </head>
      <compl xsi:type="VPPhraseSpec" FORM="PAST_PARTICIPLE">
        <head cat="VERB">
          <base>need</base>
        </head>
      </compl>
    </vp>
  </child>
</Document>
```

Generates: "Refactoring is needed"

https://github.com/simplenlg/simplenlg

The 'NLG pipeline'

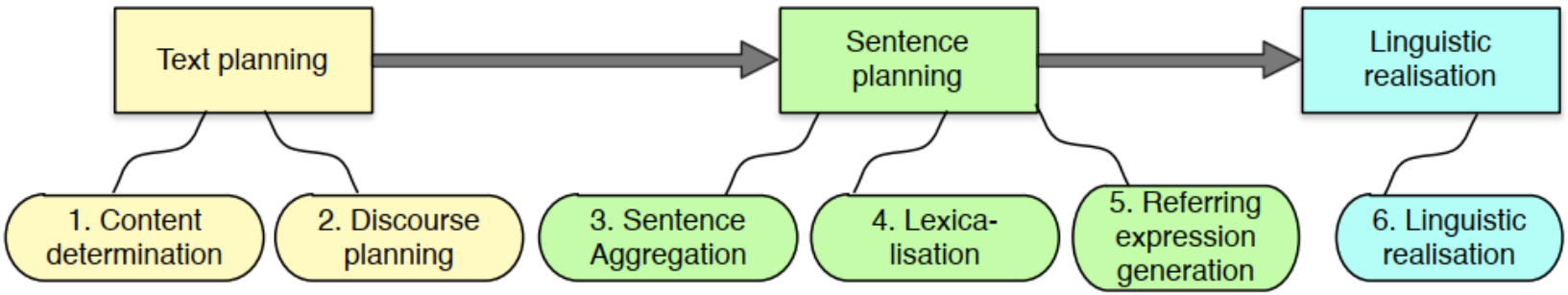


1. What structured data/info/ knowledge do you want to put into NL sentences?
2. In what order should it be presented?

3. Which messages to put together into a sentence?
4. Which words and phrases will it use for each domain concept and relation?
5. Which words or phrases to select to identify domain entities?

6. Use grammar rules to produce syntac-tically, morphological-ly, and orthographical-ly correct (and is also meaningful)

The 'NLG pipeline' of ontology verbalisation



1. What structured data/info/ knowledge do you want to put into NL sentences?
2. In what order should it be presented?

3. Which messages to put together into a sentence?
4. Which words and phrases will it use for each domain concept and relation?
5. Which words or phrases to select to identify domain entities?

6. Use grammar rules to produce syntac-tically, morphological-ly, and orthographical-ly correct (and is also meaningful)

The NLG 'pipeline'

Ontology verbalisation

1. The (OWL) ontology
2. Your choice (e.g., first all classes and class expressions in the TBox, then the object properties, etc.)

3. Aim: sentence for each axiom
4. Use vocabulary of the ontology; Select term for each constructor in the language (Each/All, and, some/at least one)
5. Combine related small axiom, or to relate the sentences generated for a large axiom

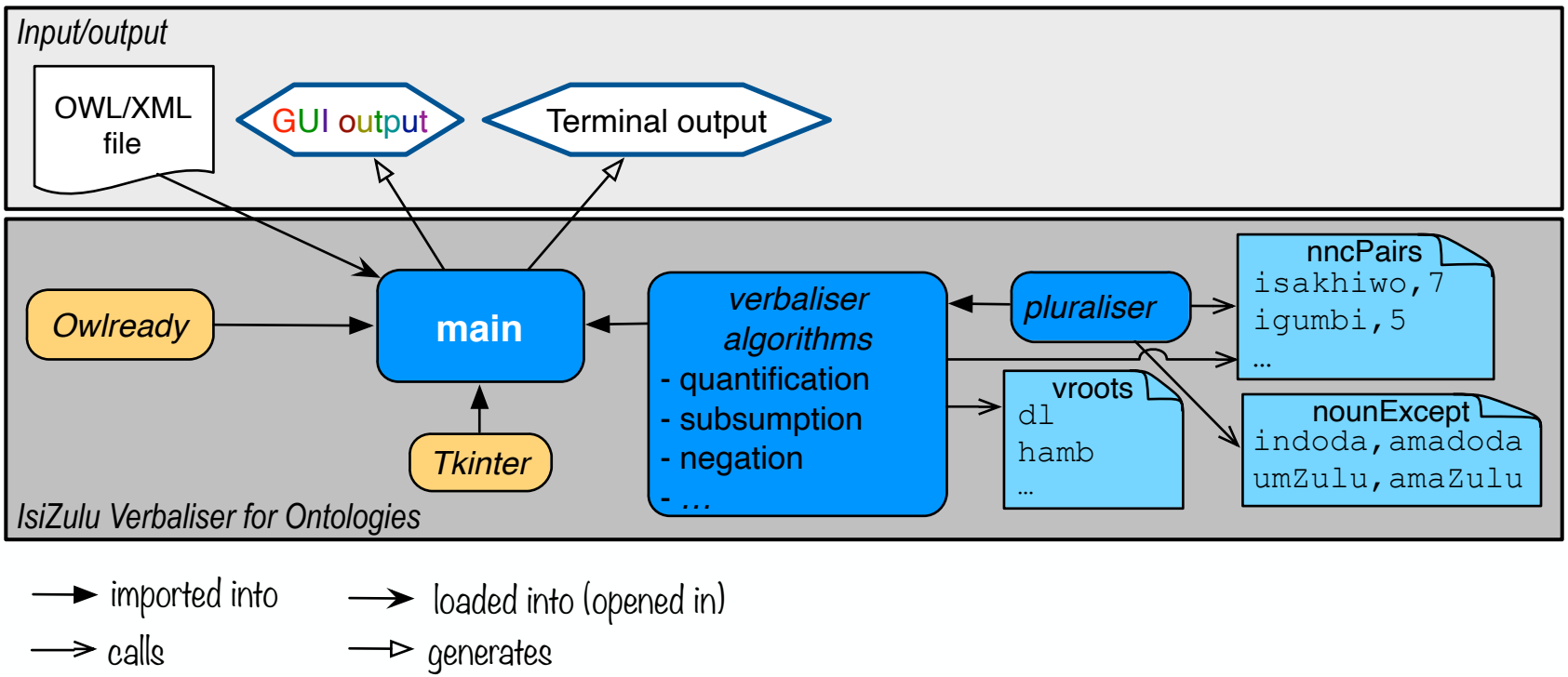
6. Language-specific issues (e.g., singular/plural of the class in agreement with conjugation of the verb, 'a' and 'an' vs 'a(n)', etc.)

Outline

Example tools

- Ontology → NL, NL → ontology, or both
- Subject domain independent ones: e.g., SWAT4NL, Rabbit, ACE, GF-based ones
- Tailored to the ontology: e.g., SNOMED CT verbaliser
- Longer list of tools with references: `https://people.cs.uct.ac.za/~mkeet/OEbook/verbalisersSorted.htm`
- Review on ontology verbalisation: [22]

Example: the isiZulu verbaliser for ontologies (1/3)



Implementation (2/3)

```
450         <SubClassOf>
451             <Class IRI="#indlovu"/>
452             <Class IRI="#isilwane"/>
453         </SubClassOf>
454         <SubClassOf>
455             <Class IRI="#indlovu"/>
456             <ObjectSomeValuesFrom>
457                 <ObjectProperty IRI="#dla"/>
458                 <Class IRI="#ihlamvana"/>
459             </ObjectSomeValuesFrom>
460         </SubClassOf>
```

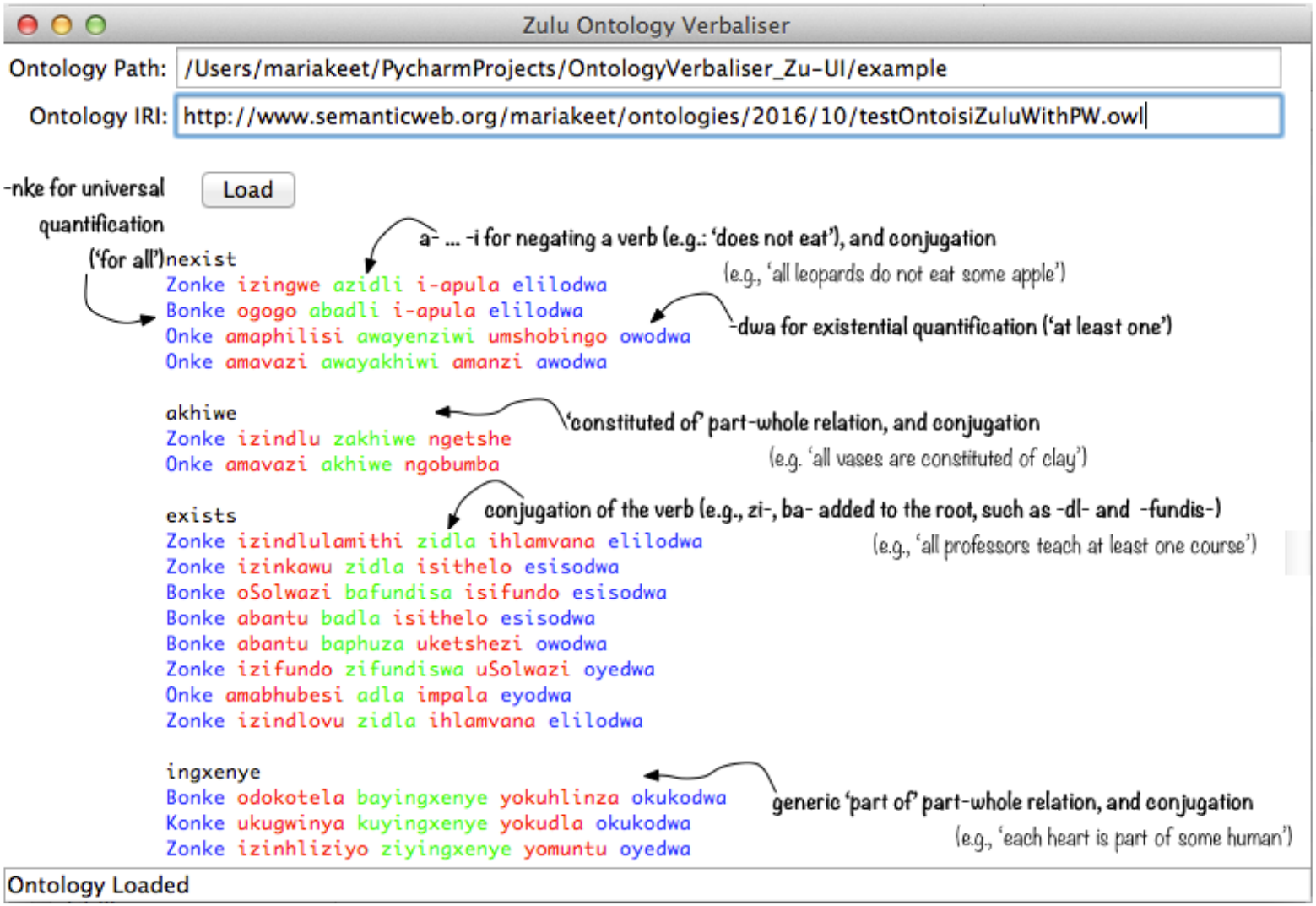
https://github.com/mkeet/GENIproject/

Implementation (2/3)

```
484  #simple existential quantification
485  # modified cf zulurules to handle also vowel-commencing vroots
486  def exists_zu(sub,op,super):
487      nc1m = find_nc(sub)
488      nc2m = find_nc(super)
489      pl = plural_zu(sub,nc1m)
490      nc2 = strip_m(nc2m)
491      ncp = look_ncp(nc1m)
492      qca = look_qca(ncp)
493      rc = look_relc(nc2)
494      qc = look_qce(nc2)
495      rt = find_rt(op)
496      if rt[0] in 'aeiou':
497          conjugrt = sc_vowel_vroot(rt,ncp)
498      else:
499          sc = look_sc(ncp)
500          conjugrt = sc + rt
501      return qca + ' ' + pl + ' ' + conjugrt + 'a' + ' ' + super + ' ' + rc + qc + 'dwa'
```

https://github.com/mkeet/GENIproject/

Sentences outputted as pretty printing or plaintext (3/3)



Choices and requirements (see also [2])

- Input
- Output
- Context

Choices and requirements (see also [2])

- Input
  - File type (OWL (species, syntax), CLIF, ...) and size
  - Subject domain- , task- dependent or not
- Output

- Context

Choices and requirements (see also [2])

- Input
  - File type (OWL (species, syntax), CLIF, ...) and size
  - Subject domain- , task- dependent or not
- Output
  - Sentences or paragraphs; in writing or TTS
  - Fluency of the text (syntactically, orthographically, grammatically correct or almost so)
  - Target language(s)
- Context

Choices and requirements (see also [2])

- Input
  - File type (OWL (species, syntax), CLIF, ...) and size
  - Subject domain- , task- dependent or not
- Output
  - Sentences or paragraphs; in writing or TTS
  - Fluency of the text (syntactically, orthographically, grammatically correct or almost so)
  - Target language(s)
- Context
  - Target genre and type of sentences, audience and user profile
  - Communicative goal
  - Setting of NL consumption (mobile, desktop, in museum etc)

What is there to configure, as a minimum? (1/2)

- One sentence per axiom type or more for variation

What is there to configure, as a minimum? (1/2)

- One sentence per axiom type or more for variation
- Template specification language

- Choice of whether to add some grammar; if so, how

What is there to configure, as a minimum? (1/2)

- One sentence per axiom type or more for variation
- Template specification language
  - Mostly done ad hoc (XML xsd, dtd)
  - Use an ontology for that for additional validation [20]
- Choice of whether to add some grammar; if so, how

What is there to configure, as a minimum? (1/2)

- One sentence per axiom type or more for variation
- Template specification language
  - Mostly done ad hoc (XML xsd, dtd)
  - Use an ontology for that for additional validation [20]
- Choice of whether to add some grammar; if so, how
  - Only 'prettify'? e.g., "a(n)" vs `indefArt(...)`, from "camelCase" to "camel case"
  - More extensive; e.g., referring expression generation, sentence aggregation
    "X produces Y." + "X produces Z." → "X produces Y and Z."
  - Grammar specification or engine

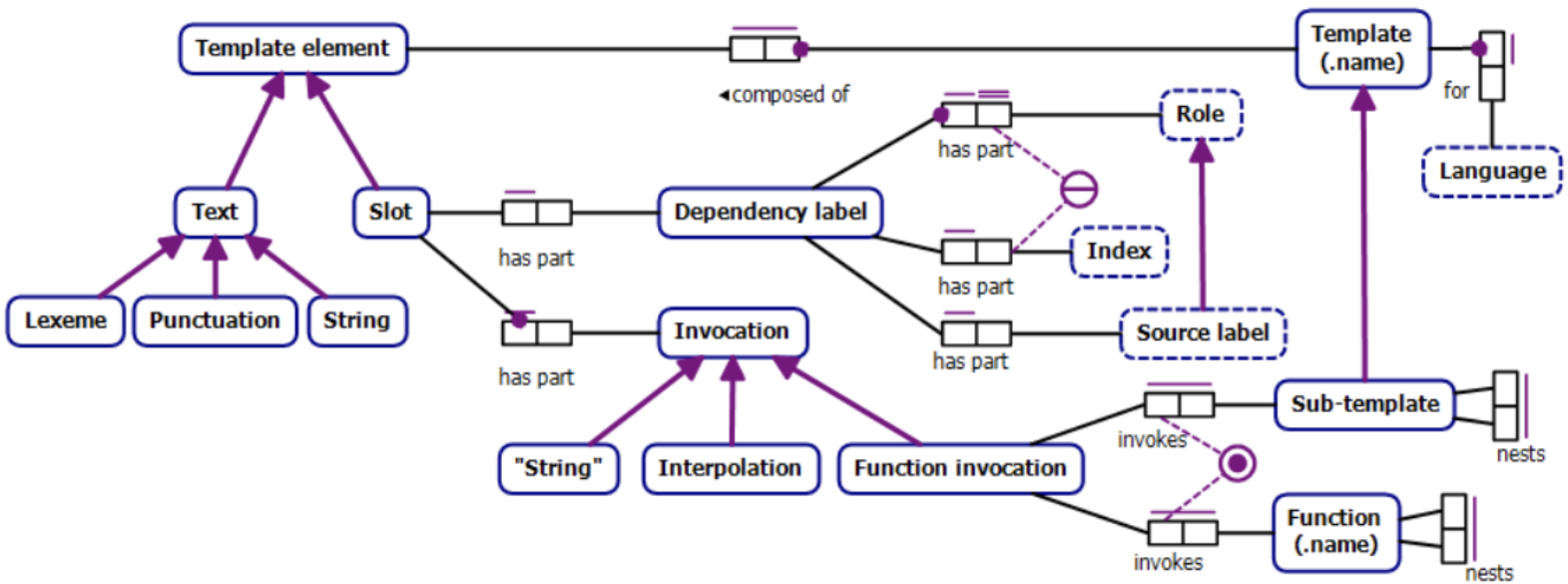
A note on template languages

- There are many

A note on template languages

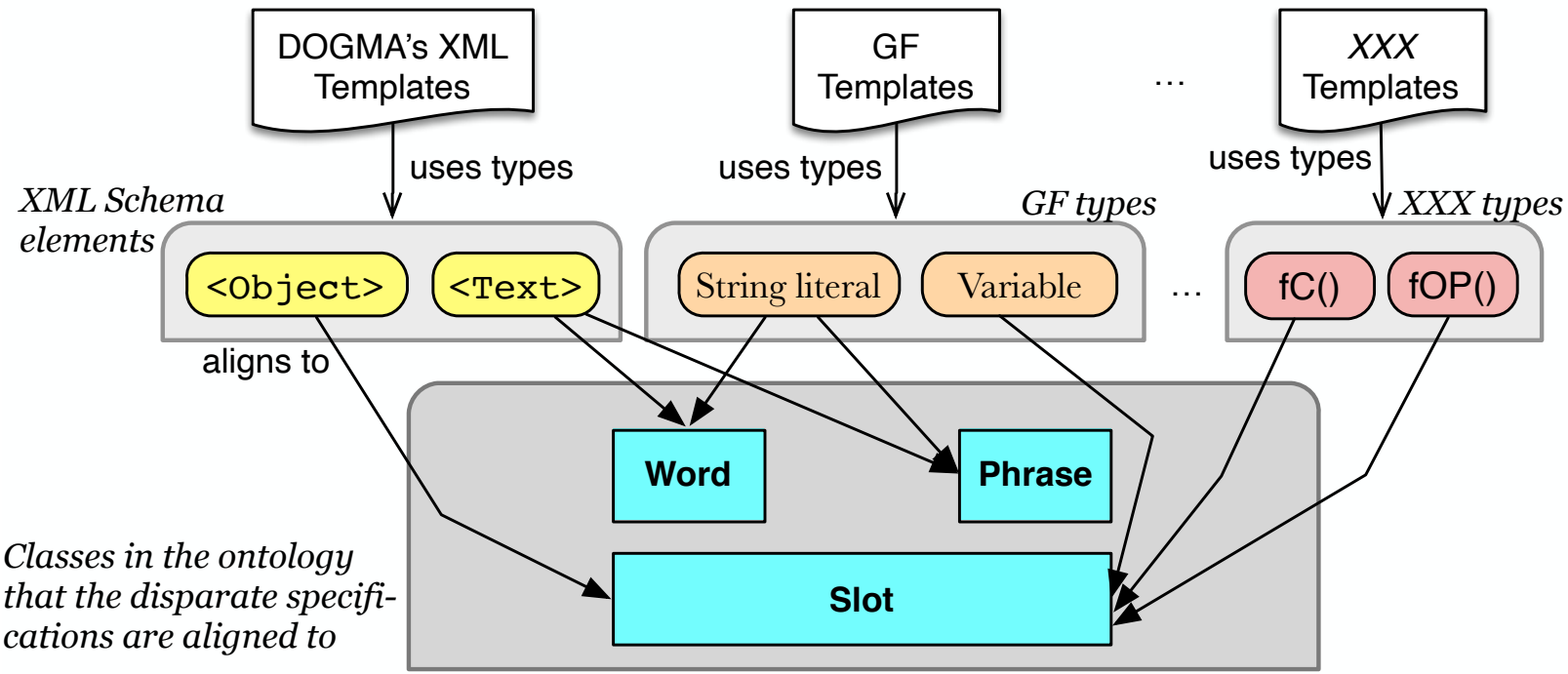
- There are many
- Fully declarative ... functional
- How template components interact with the grammar (if allowed)
- The tasks of other modules in the realiser pipeline, if any
- Type of input it needs to process
- For which natural language
- Language to declare the templates in (XML, JSON, ...)

Example: proposed template language for Abstract Wikipedia (summarised)

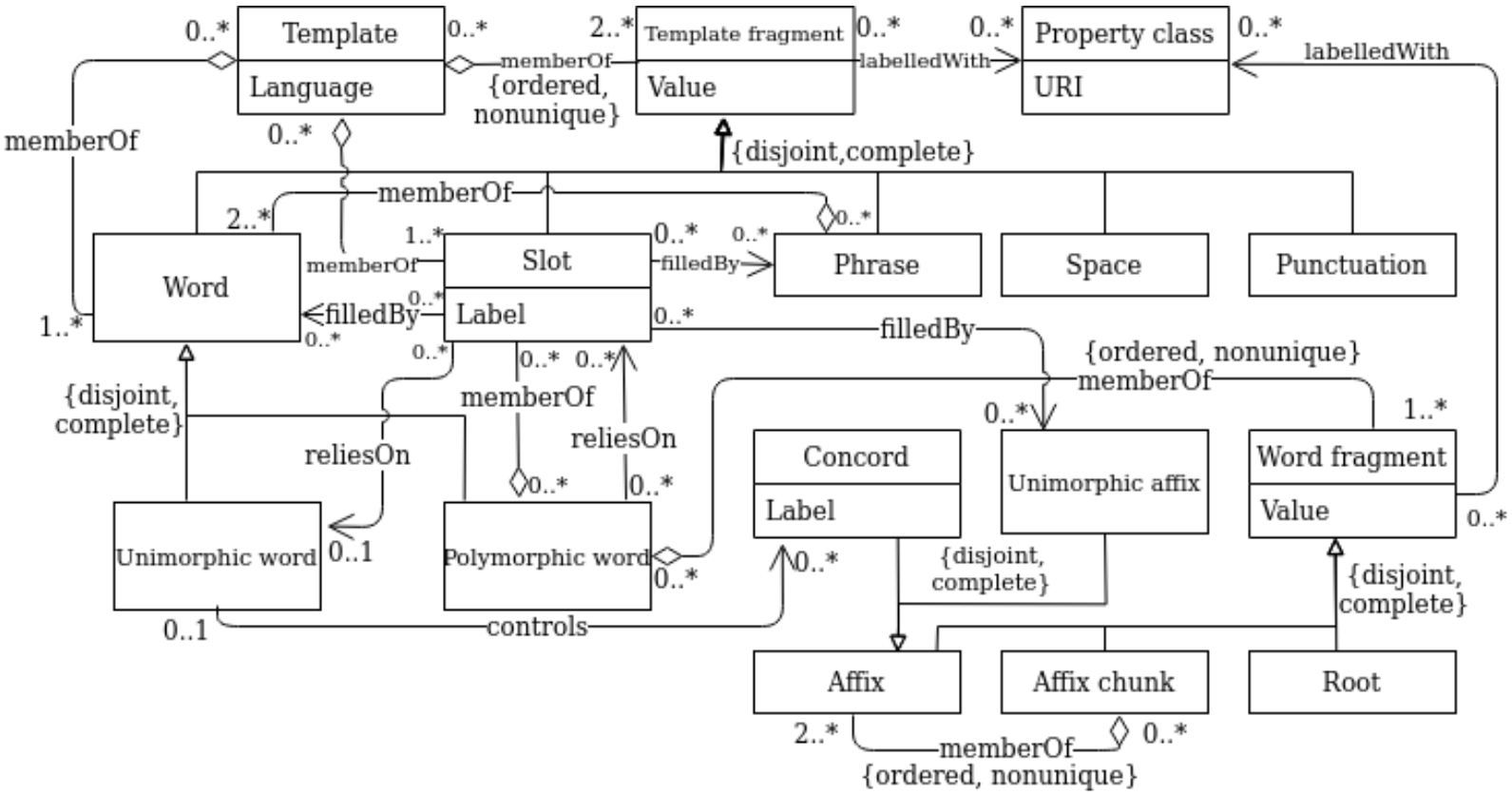


https://meta.wikimedia.org/wiki/Abstract_Wikipedia/Template_Language_for_Wikifunctions

An ontology for template languages?



An ontology for template language: ToCT [20]



What is there to configure, as a minimum? (2/2)

- The technology of the generation aspects:
- Where the templates will come from
- Build your own system or reuse

What is there to configure, as a minimum? (2/2)

- The technology of the generation aspects:
- Where the templates will come from
  - Hand-crafted from the start
  - Patterns from elsewhere
  - Mine text for sample sentence structures
- Build your own system or reuse

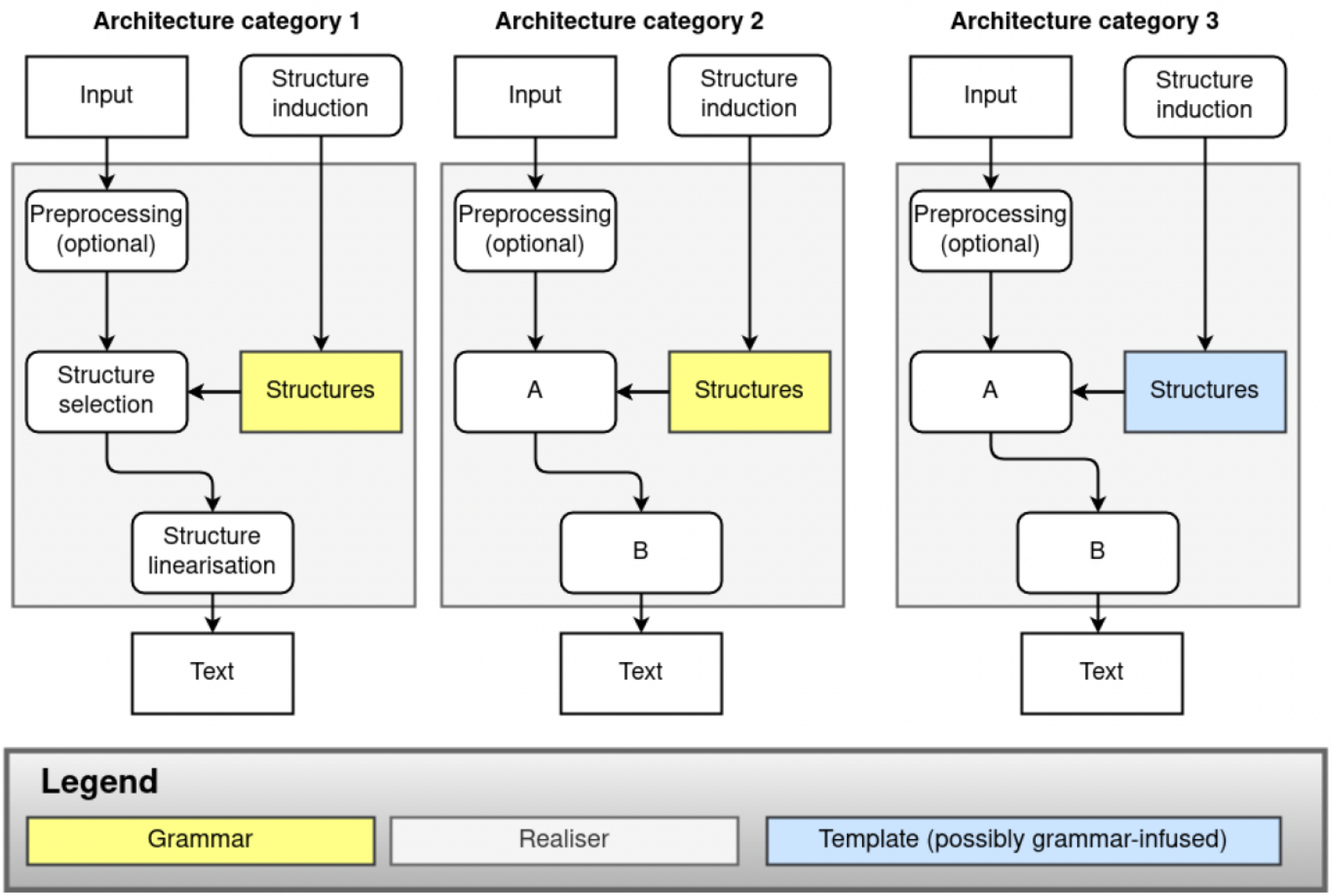
What is there to configure, as a minimum? (2/2)

- The technology of the generation aspects:
- Where the templates will come from
  - Hand-crafted from the start
  - Patterns from elsewhere
  - Mine text for sample sentence structures
- Build your own system or reuse
  - Java with OWL API
  - Python with Owlready
  - ?? with ?? for ontology language X
  - Generic systems like SimpleNLG

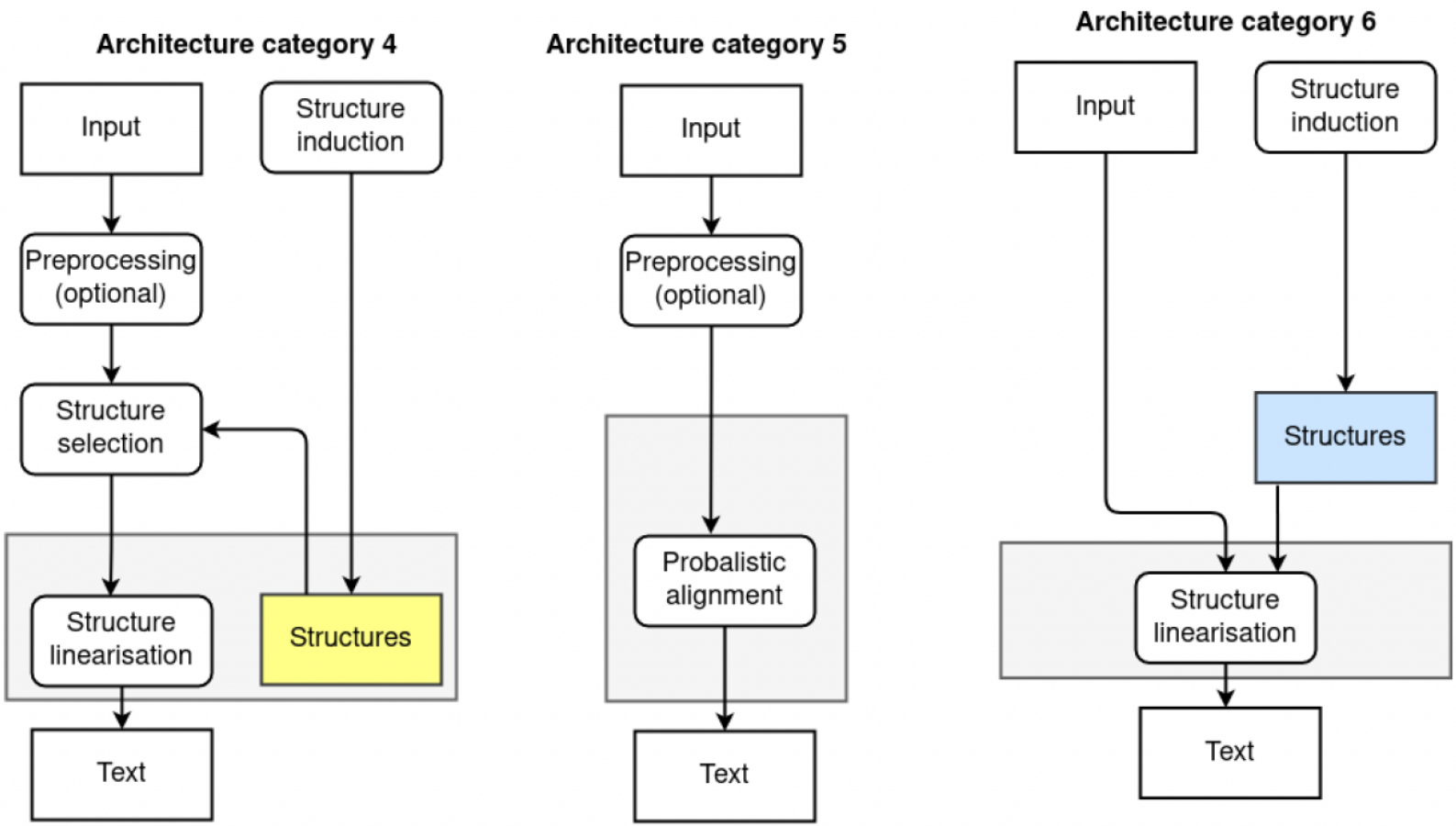
Realisation architectures: some terminology [19]

- Structure selection: connects the surface realiser to prior modules; responsible for making linguistic decisions given the semantic input
- Structure encoding: This is the method used to capture the sentential structure
- Structure induction: This refers to the method used to create the structures used for capturing sentences.
- Structure linearisation: This is the formation of text from some ordering structure.
- Ranking: The candidate filtering mechanism is responsible for selecting one sentence/structure out of many candidate output sentences or sentential structures

Realisation architecture categories



Realisation architecture categories



The modules labelled A and B represent different combinations of structure selection, linearisation, and ranking.

Outline

How to evaluate?

- Typical way of evaluating: ask linguists and/or intended target group
- Questions depend on what you want to know; e.g.,
  - Does the text capture the semantics adequately?
  - Must it really be grammatically correct or is understandable also acceptable?
  - Compared against alternate representation (figures, tables) or human-authored text?

How to evaluate?

- Typical way of evaluating: ask linguists and/or intended target group
- Questions depend on what you want to know; e.g.,
  - Does the text capture the semantics adequately?
  - Must it really be grammatically correct or is understandable also acceptable?
  - Compared against alternate representation (figures, tables) or human-authored text?
- In times of ML/DL go fast & break things: BLEU & ROUGE (automated): similarity of output to training sentences

Evaluation – certain examples (1/3)

Which sentence verbalises $\texttt{Lion} \sqsubseteq = 4\,\texttt{hasPart.Leg}$ best?

1. Each Lion hasPart exactly 4 Leg.
2. Each lion has part exactly 4 legs.
3. Each lion has as part exactly 4 legs.
4. Each lion has exactly 4 legs as part.
5. Each lion must have exactly 4 legs.
6. All lions have exactly 4 legs as part.
7. All lions have 4 legs.
8. A lion has 4 legs.

Evaluation – certain examples (2/3)

Judge the sentence "Each lion has as part exactly 4 legs."
Syntax:

1. Correct
2. Minor flaw, but acceptable
3. Major flaw(s), unacceptable

Grammar and understandability:

1. Correct, understandable
2. Not incorrect but somewhat awkward or uncommon formulation, understandable
3. Incorrect, but understandable
4. Incorrect, incomprehensible

Evaluation – certain examples (3/3)

($\textsc{DevM}^{-}$) Mandatory dynamic evolution, past:
$o \in \textsc{DevM}^{-}{}^{\mathcal{I}(t)}_{C_1,C_2} \rightarrow (o \in \mathtt{C}_1{}^{\mathcal{I}(t)} \rightarrow \exists t' < t.o \in \textsc{Dev}^{\mathcal{I}(t')}_{C_1,C_2})$.
For instance, Butterfly and the Caterpillar it used to be.

a. *Each* ..$\mathtt{C}_1$.. *must have been a(n)* ..$\mathtt{C}_2$.. , *but is not a(n)* ..$\mathtt{C}_2$.. *anymore*.
b. *Each* ..$\mathtt{C}_1$.. *was a(n)* ..$\mathtt{C}_2$.. *before, but is not a(n)* ..$\mathtt{C}_2$.. *now*.
c. *If* ..$\mathtt{C}_1$.. , *then* ..$\mathtt{C}_1$.. *was a(n)* ..$\mathtt{C}_2$.. *before, but is not a(n)* ..$\mathtt{C}_2$.. *anymore*.

[12]

Outline

Tasks

- Main exercises at `http://www.meteck.org/MoReNL/JOWO22handsonNLGonto.pdf`
- Online verbalisers to explore:
  - Try out either one with your own ontology and one provided: ACE `http://attempto.ifi.uzh.ch/site/docs/owl_to_ace.html` or `https://github.com/Kaljurand/owl-verbalizer` or SWAT `http://mcs.open.ac.uk/nlg/SWAT/Verbaliser.html` with prolog or crowd2.0 `https://crowd-app.fi.uncoma.edu.ar`

Observations from the practical?

- Fetch name (`BFO_0000015`) vs Fetch label (BFO's 'process')
- How many naming variants did you detect?

Observations from the practical?

- Fetch name (`BFO_0000015`) vs Fetch label (BFO's 'process')
- How many naming variants did you detect?
- Some axiom types were easier to verbalise than others

Inconvenient cases (1/2): cido-base.owl

Description: COVID-19 disease process

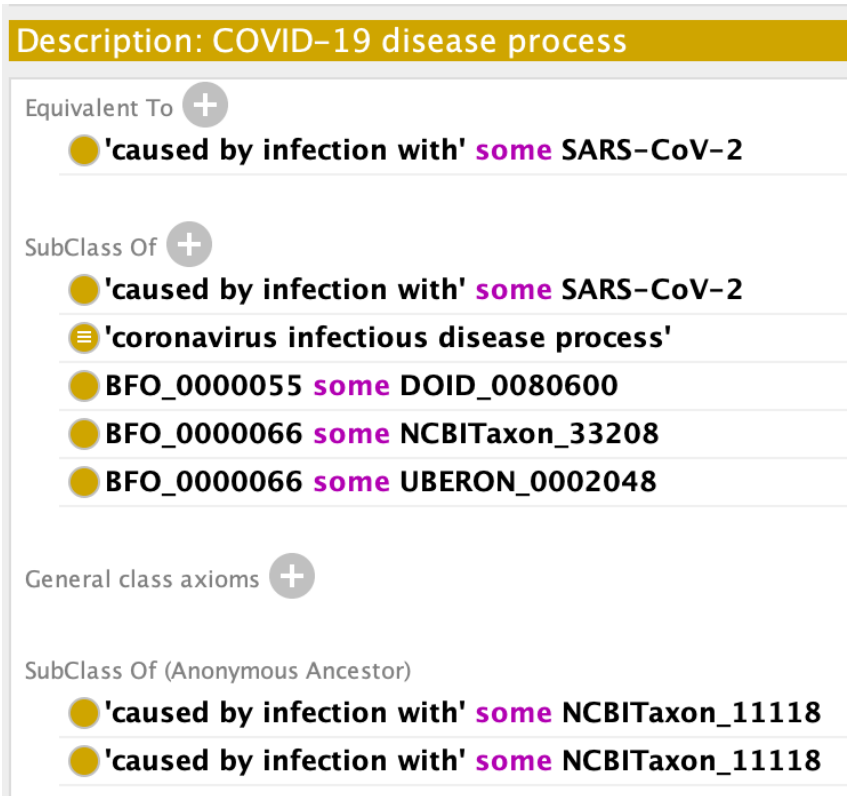
Equivalent To
- 'caused by infection with' some SARS-CoV-2

SubClass Of
- 'caused by infection with' some SARS-CoV-2
- 'coronavirus infectious disease process'
- BFO_0000055 some DOID_0080600
- BFO_0000066 some NCBITaxon_33208
- BFO_0000066 some UBERON_0002048

General class axioms

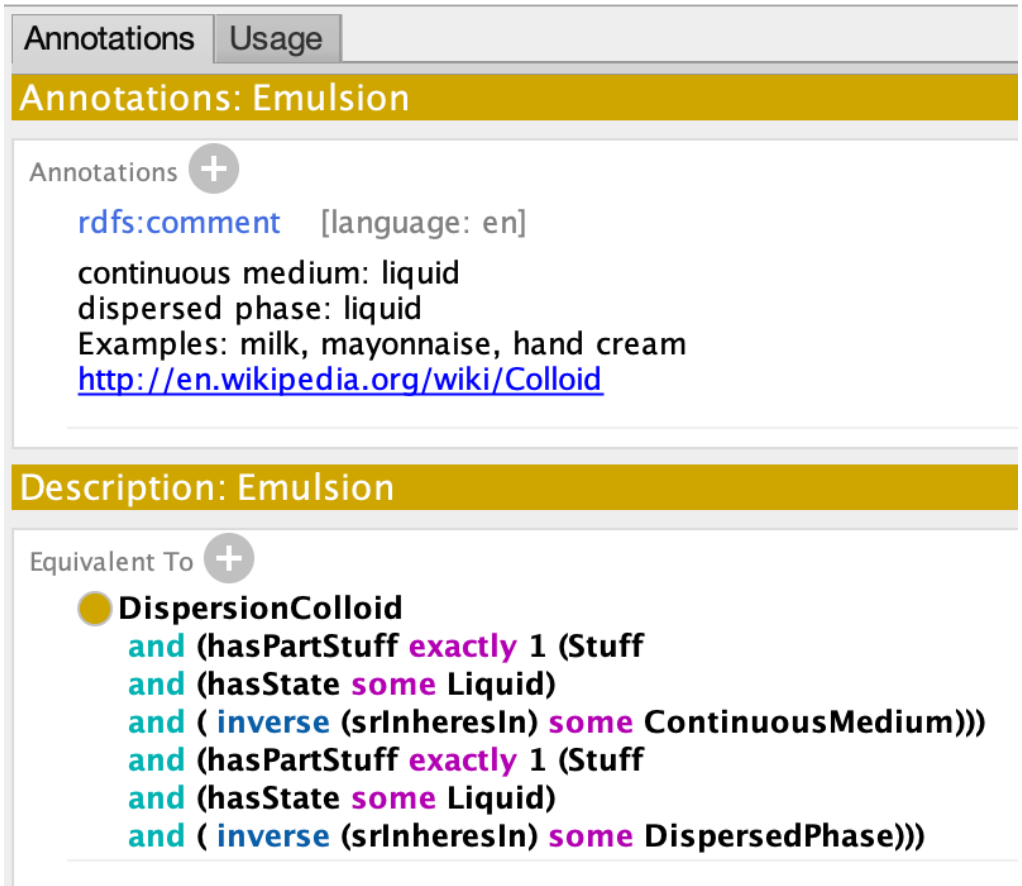
SubClass Of (Anonymous Ancestor)
- 'caused by infection with' some NCBITaxon_11118
- 'caused by infection with' some NCBITaxon_11118

Inconvenient cases (1/2): cido-base.owl



- Can't get a neat sentence out of that without additional processing
- Verbalise the equivalence axiom only? Asserted subclasses too? And the inherited ones?

Inconvenient cases (2/2): Stuff ontology



Inconvenient cases (2/2): Stuff ontology

- Each emulsion is equivalent to a dispersion colloid and has as part exactly 1 stuff that has as state a liquid state and the inverse of inheres in continuous medium and has as part exactly one stuff that has as state a liquid state and the inverse of inheres in dispersed phase.

Inconvenient cases (2/2): Stuff ontology

- Each emulsion is equivalent to a dispersion colloid and has as part exactly 1 stuff that has as state a liquid state and the inverse of inheres in continuous medium and has as part exactly one stuff that has as state a liquid state and the inverse of inheres in dispersed phase.
- Each emulsion is a dispersion colloid.

  Each emulsion has exactly one stuff part that is in the liquid state that has the role of continuous medium.

  Each emulsion has exactly one stuff part that is in the liquid state that has the role of dispersed phase.

Outline

Outline

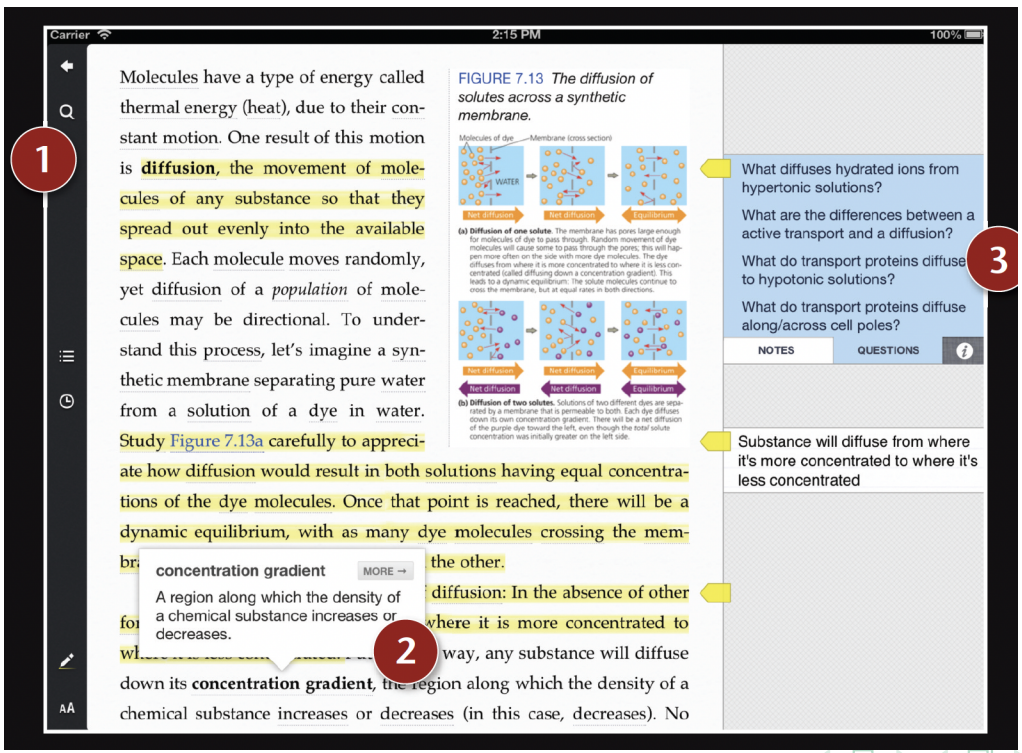
Sample scenario: Intelligent "Inquire biology" textbook

- Idea: annotate textbook with ontology, generate questions automatically, mark automatically

Sample scenario: Intelligent "Inquire biology" textbook

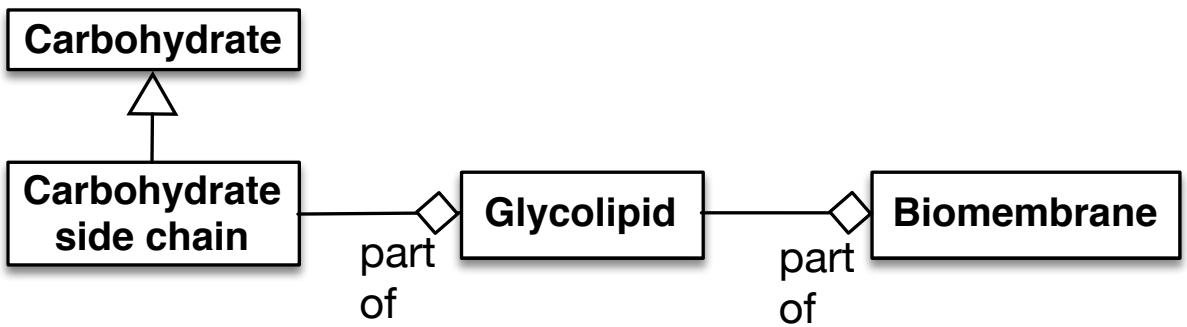
- Idea: annotate textbook with ontology, generate questions automatically, mark automatically

  Figure: Annotated p132 of Biology (9th edition) by Campbell and Reece [4]:



Intelligent "Inquire biology" textbook

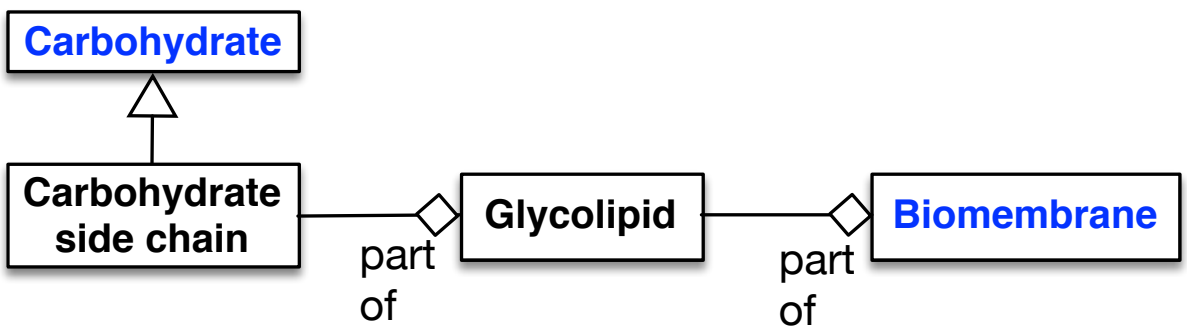
Q: What is the relation between a carbohydrate and a biomembrane?



(example from and based on [4])

Intelligent "Inquire biology" textbook

Q: What is the relation between a carbohydrate and a biomembrane?

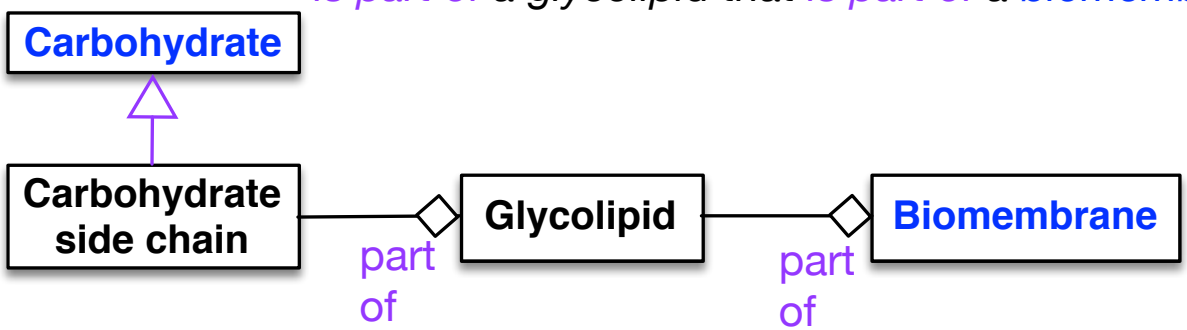


(example from and based on [4])

Intelligent "Inquire biology" textbook

Q: What is the relation between a carbohydrate and a biomembrane?

A: A carbohydrate side chain is a carbohydrate that is part of a glycolipid that is part of a biomembrane



(example from and based on [4])

Try to do that for *any* ontology [21]

- Question templates for different types of educational questions

  Is a <T_NOUN> <OP_IS_PARTICIPLE_BY> <Quantifier_some> <T_NOUN>?

  What does a <T_NOUN> <OP_VERB>?

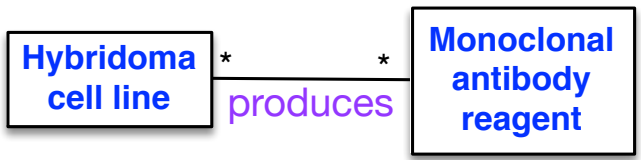
Try to do that for *any* ontology [21]

- Question templates for different types of educational questions

  Is a <T_NOUN> <OP_IS_PARTICIPLE_BY> <Quantifier_some> <T_NOUN>?

  What does a <T_NOUN> <OP_VERB>?
- Notion of "axiom prerequisites"
- (Assumes a certain way of representing something and of verbalising something)

Sample educational questions

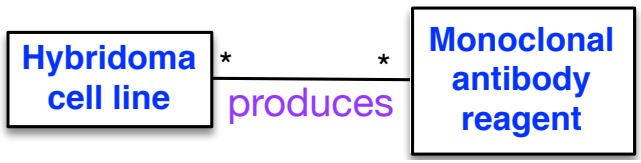


$\texttt{MonoclonalAntibodyReagent} \sqsubseteq \exists \texttt{produced\_by}.\texttt{HybridomaCellLine}$

$\texttt{produces} \equiv \texttt{produced\_by}^{-}$

Q: *Is Monoclonal antibody reagent produced by some Hybridoma cell line?*
A: Yes
Q: *What does a Hybridoma cell line produce?*
A: Monoclonal antibody reagent

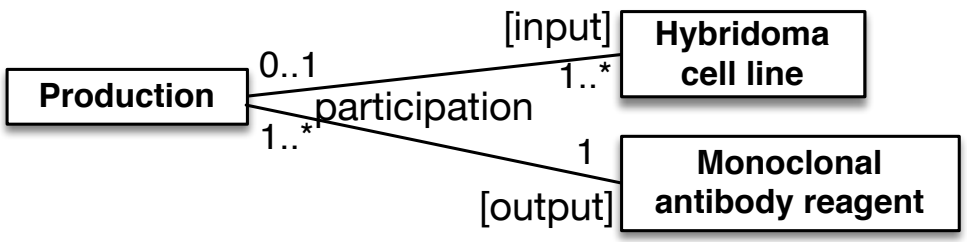
Sample educational questions



$\texttt{MonoclonalAntibodyReagent} \sqsubseteq \exists \texttt{produced\_by.HybridomaCellLine}$

$\texttt{produces} \equiv \texttt{produced\_by}^{-}$

Q: *Is Monoclonal antibody reagent produced by some Hybridoma cell line?*
A: Yes
Q: *What does a Hybridoma cell line produce?*
A: Monoclonal antibody reagent

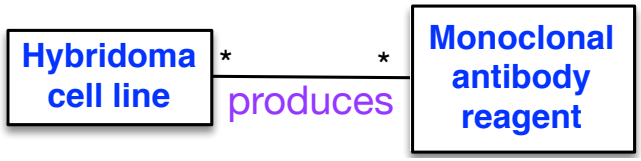


$\texttt{MonoclonalAntibodyReagent} \sqsubseteq \exists \texttt{participates\_in.Production}$

$\texttt{HybridomaCellLine} \sqsubseteq \forall \texttt{participates\_in.Production}$

.... something with the `[input]` and `[output]` roles ....
Algorithm TBD, for every ontology pattern

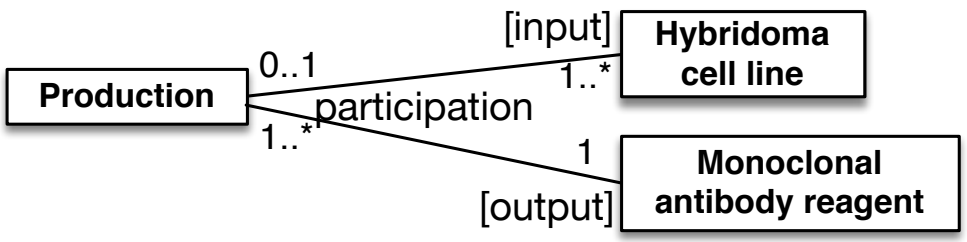
Sample educational questions



$\texttt{MonoclonalAntibodyReagent} \sqsubseteq \exists \texttt{produced\_by.HybridomaCellLine}$

$\texttt{produces} \equiv \texttt{produced\_by}^{-}$

Q: *Is Monoclonal antibody reagent produced by some Hybridoma cell line?*
A: Yes
Q: *What does a Hybridoma cell line produce?*
A: Monoclonal antibody reagent

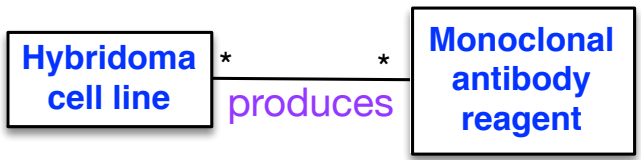


$\texttt{MonoclonalAntibodyReagent} \sqsubseteq \exists \texttt{participates\_in.Production}$

$\texttt{HybridomaCellLine} \sqsubseteq \forall \texttt{participates\_in.Production}$

... Q: *What does a Hybridoma cell line participate in?*
A: Production ..

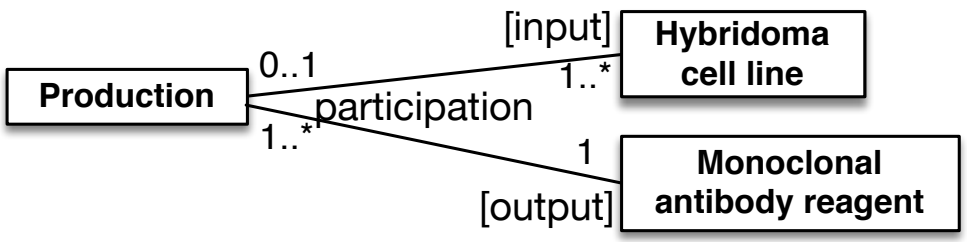
Sample educational questions



$\texttt{MonoclonalAntibodyReagent} \sqsubseteq \exists \texttt{produced\_by.HybridomaCellLine}$

$\texttt{produces} \equiv \texttt{produced\_by}^{-}$

Q: *Is Monoclonal antibody reagent produced by some Hybridoma cell line?*
A: Yes
Q: *What does a Hybridoma cell line produce?*
A: Monoclonal antibody reagent

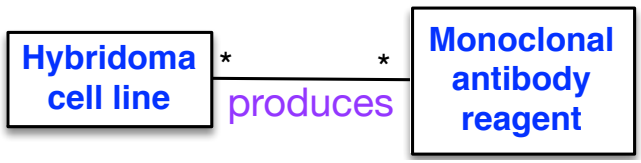


$\texttt{MonoclonalAntibodyReagent} \sqsubseteq \exists \texttt{participates\_in.Production}$

$\texttt{HybridomaCellLine} \sqsubseteq \forall \texttt{participates\_in.Production}$

... Q: *What does a Hybridoma cell line participate in?*
A: ~~Production~~ ..

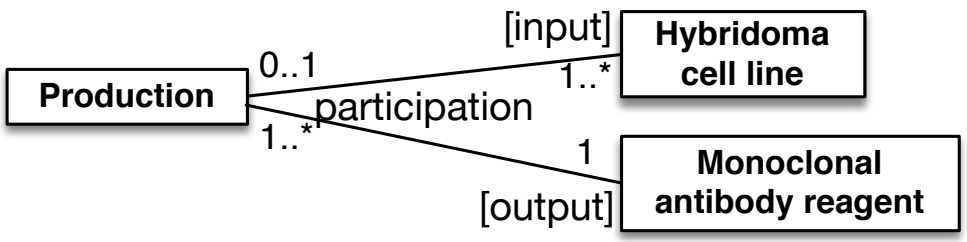
Sample educational questions



$\texttt{MonoclonalAntibodyReagent} \sqsubseteq \exists \texttt{produced\_by.HybridomaCellLine}$

$\texttt{produces} \equiv \texttt{produced\_by}^{-}$

Q: *Is Monoclonal antibody reagent produced by some Hybridoma cell line?*
A: Yes
Q: *What does a Hybridoma cell line produce?*
A: Monoclonal antibody reagent

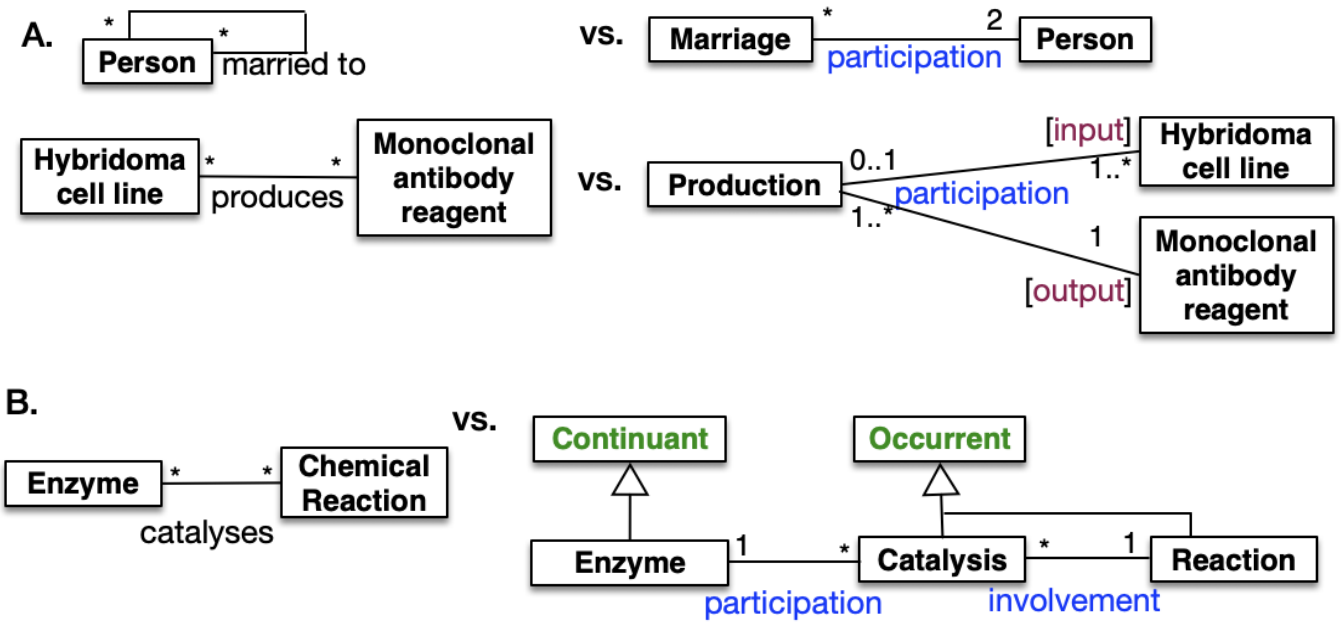


$\texttt{MonoclonalAntibodyReagent} \sqsubseteq \exists \texttt{participates\_in.Production}$

$\texttt{HybridomaCellLine} \sqsubseteq \forall \texttt{participates\_in.Production}$

... Q: *What does a Hybridoma cell line participate in?*
A: Production of monoclonal antibody reagents ..

Ontology patterns: to reify or not to reify... [6]



Correct but awkward or confusing sentences

(BioTop) Does a material object project onto an immaterial three dimensional physical entity?

(BioTop) A taxon quality projects onto a taxon value region. True or false?

note: taxon value region is an "abstract region in which the values of biological taxa are located (cf. Schulz et.al ISMB 2008)."

Correct but awkward or confusing sentences

(BioTop) Does a material object project onto an immaterial three dimensional physical entity?

(BioTop) A taxon quality projects onto a taxon value region. True or false?

note: taxon value region is an "abstract region in which the values of biological taxa are located (cf. Schulz et.al ISMB 2008)."

(SO) Does a mixed stuff have a part stuff that is a stuff?

$\texttt{MixedStuff} \equiv \texttt{Stuff} \sqcap \exists \texttt{hasPartStuff}.\texttt{Stuff}$

Correct but awkward or confusing sentences

(BioTop) Does a material object project onto an immaterial three dimensional physical entity?

(BioTop) A taxon quality projects onto a taxon value region. True or false?

note: taxon value region is an "abstract region in which the values of biological taxa are located (cf. Schulz et.al ISMB 2008)."

(SO) Does a mixed stuff have a part stuff that is a stuff?

$\texttt{MixedStuff} \equiv \texttt{Stuff} \sqcap \exists \texttt{hasPartStuff.Stuff}$

(BioTop) Which condition has a life that is some life?

$\texttt{condition} \equiv \texttt{disposition} \sqcup \texttt{function} \sqcup \texttt{material object} \sqcup \texttt{process}$

$\texttt{material object} \sqsubseteq \exists \texttt{has life.life}$

Better results with certain domain ontologies

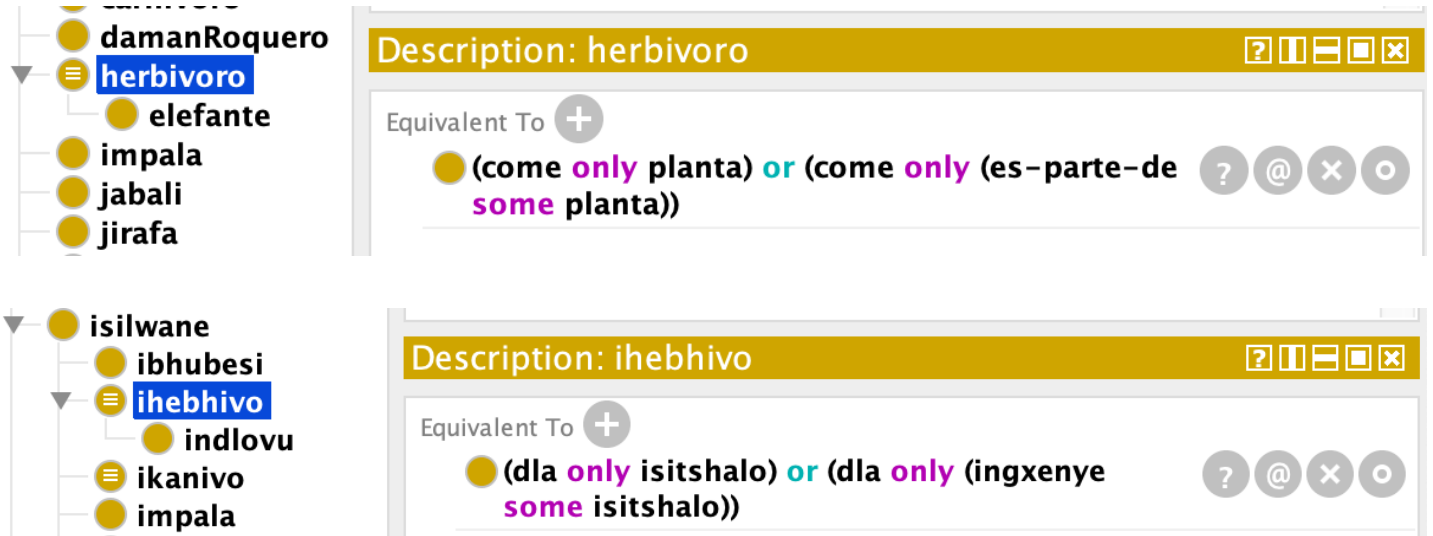
- An applied style
- With abundance in granularity of relations
- Not just/mainly a 'bare' hierarchy, not just domain & range axioms
- 'hasX' and other OP naming considerations
- Naming in ontology often within-context, but out-of-context in the self-standing questions

Ontology patterns and styles

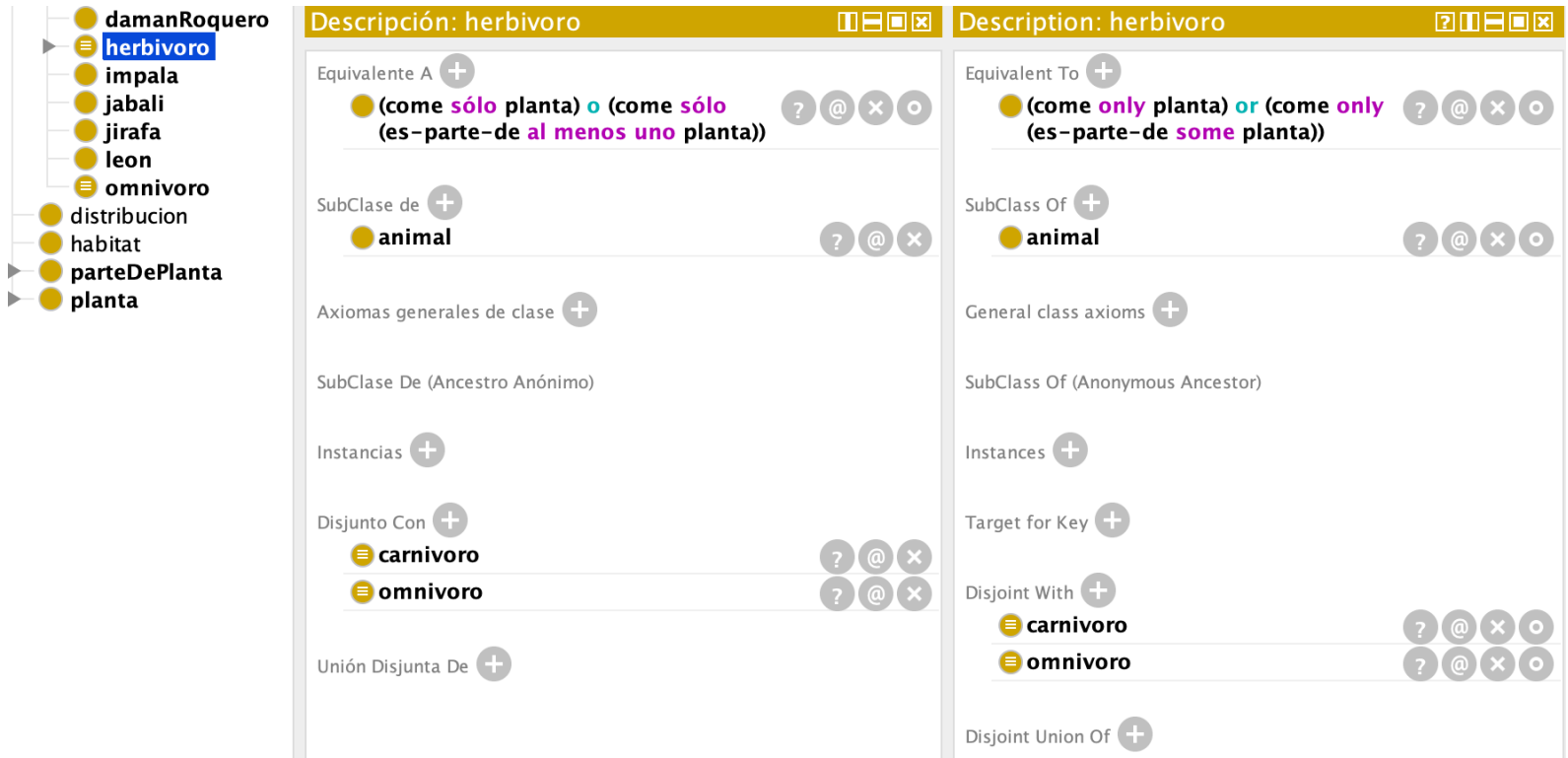
- Which patterns are there
- If the neatly combine, ontology 'styles' emerge
- Either swap styles or modify algorithms
- (More about that in my bio-ontologies keynote at ISMB'22)

Outline

A first hurdle



Tool localisation?



Localisation description, plugin, and code: `https://people.cs.uct.ac.za/~mkeet/OEbook/OEsoftware.html#odeloc`

NLG in languages other than English – more grammar

- Cases and gender in German, noun classes in isiZulu
- 'Deep prepositions' in isiZulu, Lithuanian
- Extensive phonological conditioning, agglutination, inflection, etc.

NLG in languages other than English – more grammar

- Cases and gender in German, noun classes in isiZulu
- 'Deep prepositions' in isiZulu, Lithuanian
- Extensive phonological conditioning, agglutination, inflection, etc.

$\Rightarrow$ Can't 'just fetch and tweak' vocabulary

Examples: German

- $Employee \sqsubseteq \exists worksFor.Company$

  arbeitet für 'works for' with 3rd case / dativ

  der Betrieb 'company', M; *die Firma*, F

- Alle Angestellten arbeiten für mindestens einem Betrieb
- Alle Angestellten arbeiten für mindestens einer Firma

Examples: isiZulu

Axiom type $\forall x(X(x) \rightarrow \exists y(R(x,y) \wedge Y(y)))$
$X \sqsubseteq \exists R.Y$

English All [noun x pl.] [verb 3rd pers. pl.] at least one [noun y]
All professors teach at least one course
All professors write at least one book
All carnivores eat at least one animal
All elephants eat at least one apple

IsiZulu [QCall$_{nc_x,pl}$] [noun x$_{nc_x}$ pl.] [SC$_{nc_x,pl}$-verb] [noun y$_{nc_y}$] RC$_{nc_y}$-QC$_{nc_y}$-dwa
Bonke oSolwazi bafundisa isifundo esisodwa
Bonke oSolwazi babhala incwadi eyodwa
Onke amakhanivo adla isilwane esisodwa
Zonke izindlovu zidla i-apula elilodwa

$\forall x(uSolwazi(x) \rightarrow \exists y(fundisa(x,y) \wedge isifundo(y)))$
$uSolwazi \sqsubseteq \exists fundisa.isifundo$

Possible consequences and design questions

- How to deal with the vocabulary?
- Where to store the grammatical features and the processing?
- How to connect that extra baggage?
- Where does it affect the verbalisation architecture?

Possible consequences and design questions

- How to deal with the vocabulary?
  - Monolingual ontology or multilingual ontology
- Where to store the grammatical features and the processing?
- How to connect that extra baggage?
- Where does it affect the verbalisation architecture?

Possible consequences and design questions

- How to deal with the vocabulary?
  - Monolingual ontology or multilingual ontology
- Where to store the grammatical features and the processing?
  - ad hoc in same file after the template spec
  - functions embedded in the templates
  - separate grammar engine
- How to connect that extra baggage?

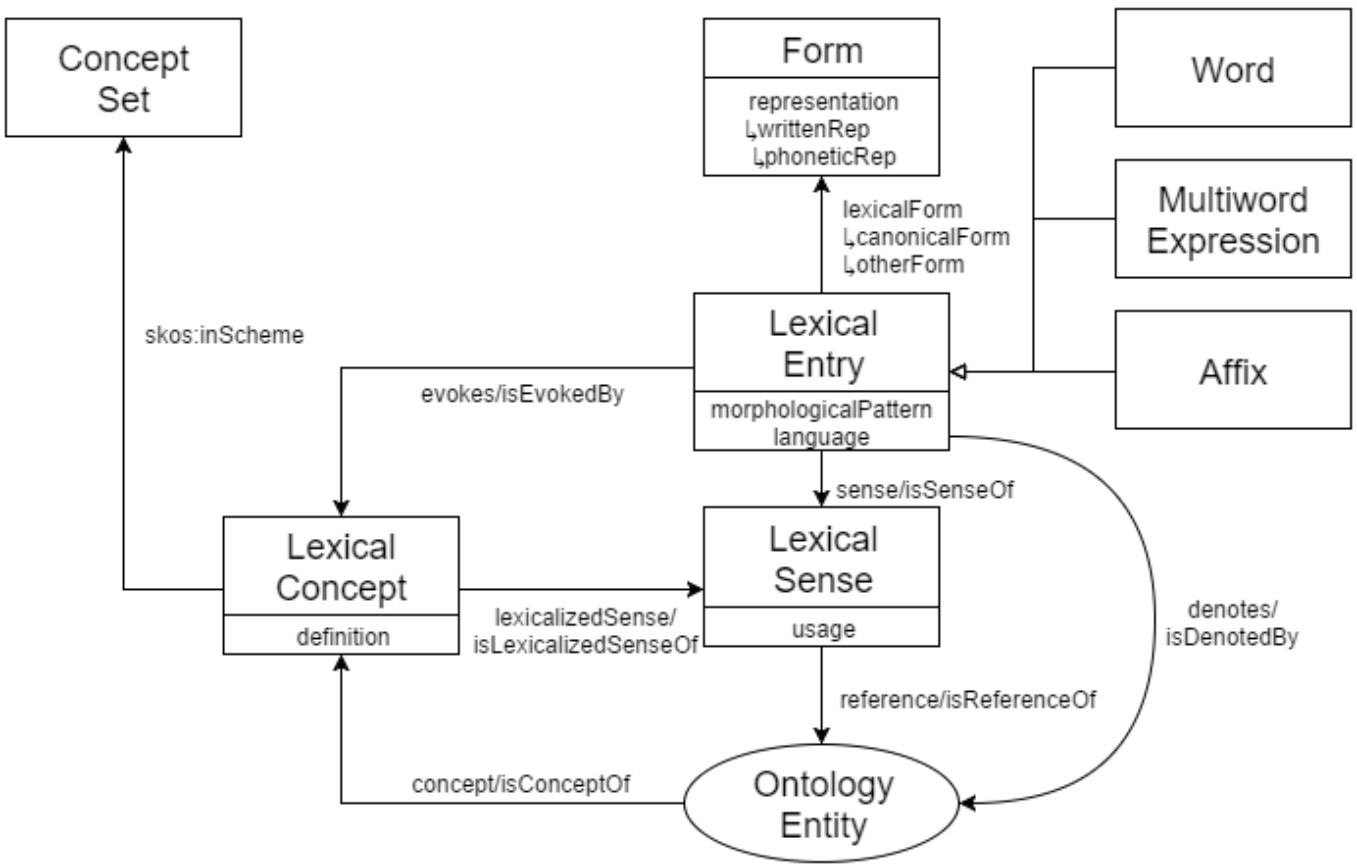
- Where does it affect the verbalisation architecture?

Possible consequences and design questions

- How to deal with the vocabulary?
  - Monolingual ontology or multilingual ontology
- Where to store the grammatical features and the processing?
  - ad hoc in same file after the template spec
  - functions embedded in the templates
  - separate grammar engine
- How to connect that extra baggage?
  - to templates
  - to ontology vocabulary in the annotation field
  - in a separate language model
- Where does it affect the verbalisation architecture?

Ontolex-Lemon

W3C community standard "to provide rich linguistic grounding for ontologies"



https://www.w3.org/2016/05/ontolex/

Sample entry linking an OWL class to language information

Link the word elephant to the entity in the ontology:

```
:lex_elephant a ontolex:LexicalEntry;
   ontolex:canonicalForm :form_elephant;
   ontolex:denotes <http://www.meteck.org/.../AWO/Elephant>.
```

And have it in the lexicon:

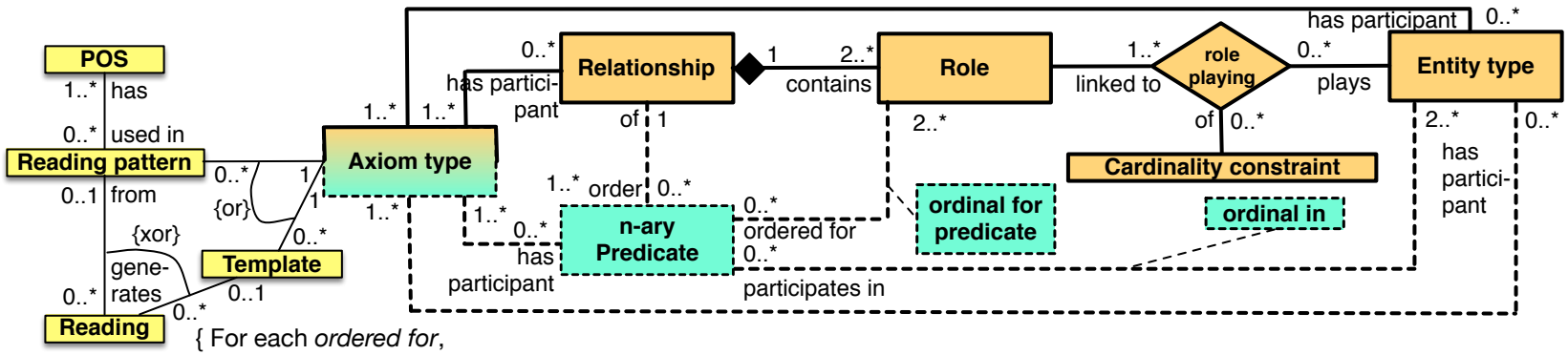
```
:form_elephant a ontolex:Form;
   ontolex:writtenRep "elephant"@en.
```

For multiple languages: add language (from ISO639), add sense, assert translation
`https://www.w3.org/2016/05/ontolex/#translation`

More detailed aspects: complications with OPs [13]

- That *für* with dative on the article of the noun of the entity that plays the object role
- The 'of' in 'part of', in isiZulu: attached to the noun of the entity that plays the role of whole
- *umuntu udla* but *inja idla* (human vs dog eats): verb conjugation depends on the noun class of the noun of the entity that plays the subject role

More detailed aspects: prepositions [13]



{ For each *ordered for*,
the *Roles ordered for* a *Predicate* are *contained in* a *Relationship* that is the *Relationship of* that *Predicate*.
(but a *Relationship* that *contains Role*s need not have a *Predicate*, and *Role*s need not be *ordered for* a *Predicate*)
Each *Axiom has participant* either a *Relationship* or an *Entity type* or both, or an *n-ary Predicate* or an *Entity type* or both. }

Role
identifier: Integer {id}
name: String
tense: {*//list from ISOcat*}
case: {*//list from ISOcat*}
hasPRE: Boolean
PRE: String [0..1]

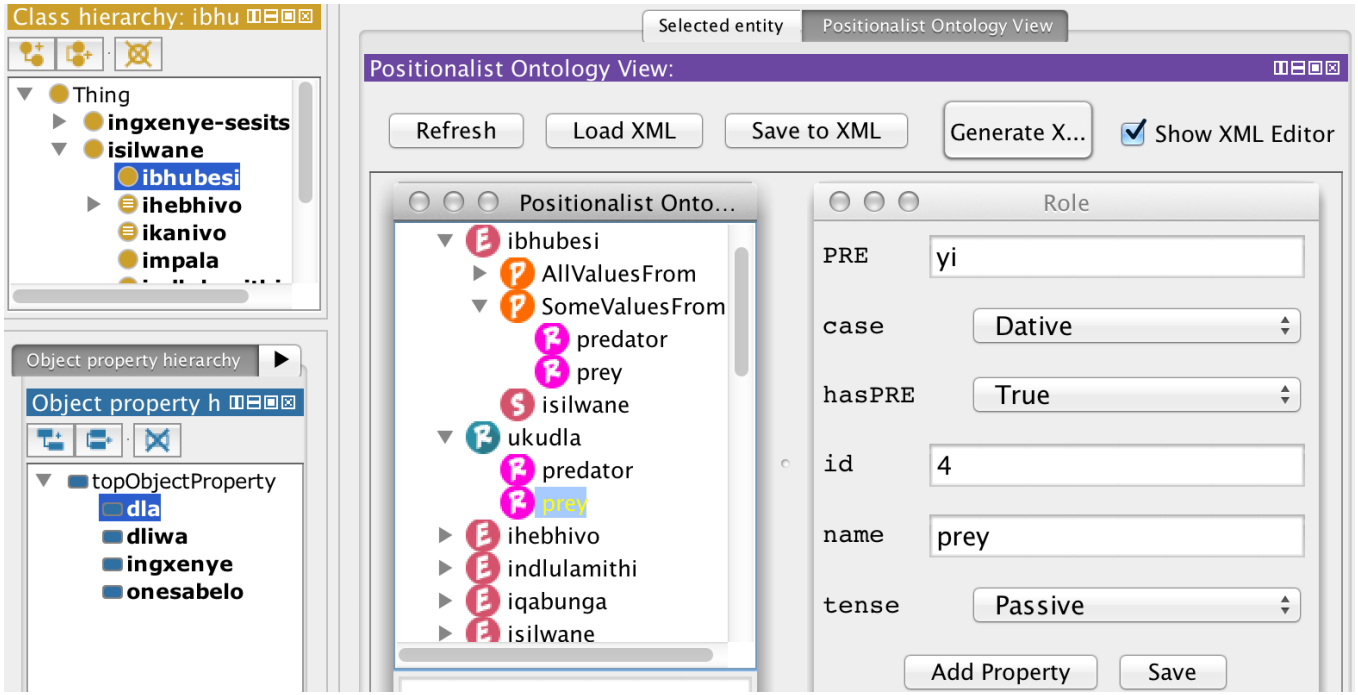
Entity type
identifier: Integer {id}
name: String
gender: {*//list from ISOcat*} [0..1]
noun class: {*//list from NCS ontology*} [0..1]
grammNumber: Integer

Reading pattern
identifier: Integer {id}
pattern: String
language: {*//ISO abbrev.*}

Axiom type
identifier: Integer {id}
positionalist: Boolean

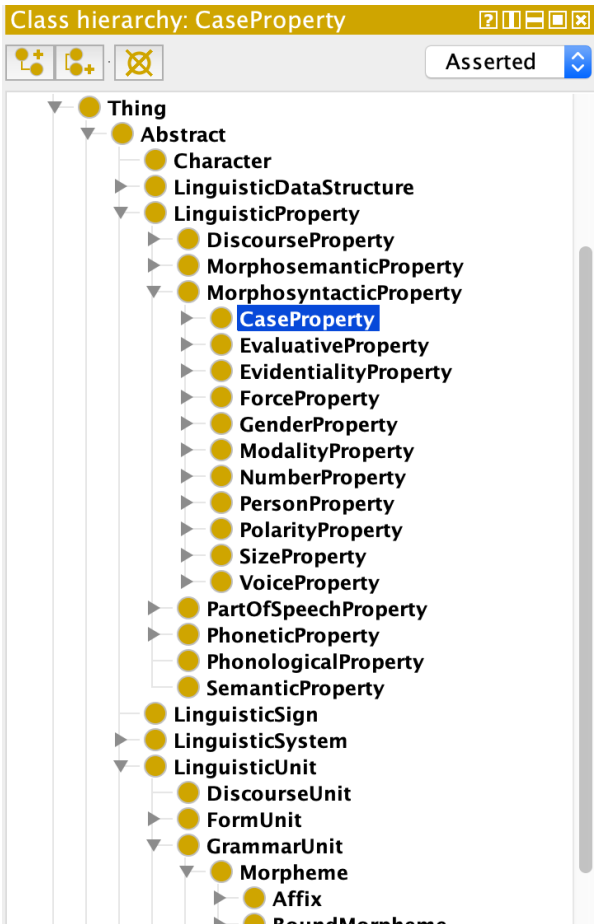
Reading
identifier: Integer {id}
sentence: String

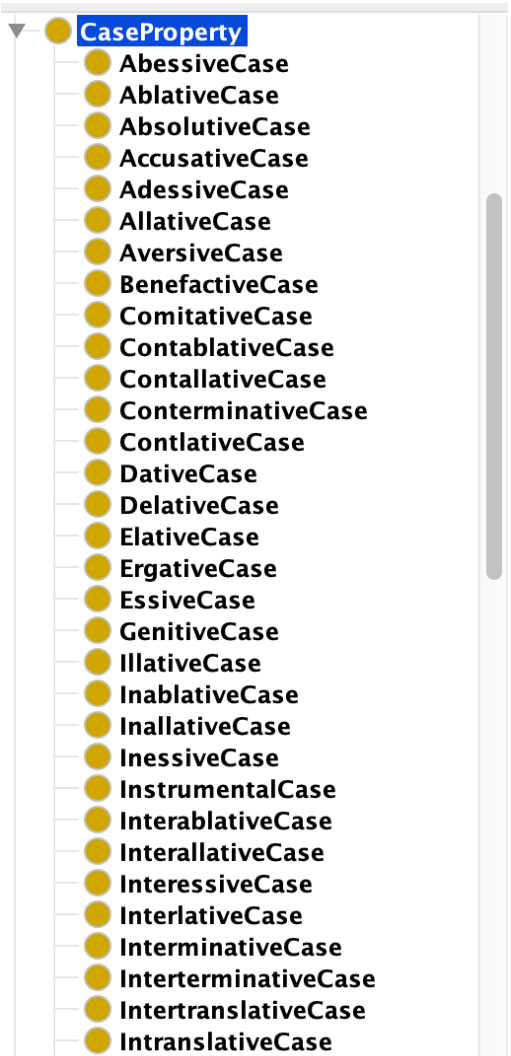
More detailed aspects: prepositions [13]

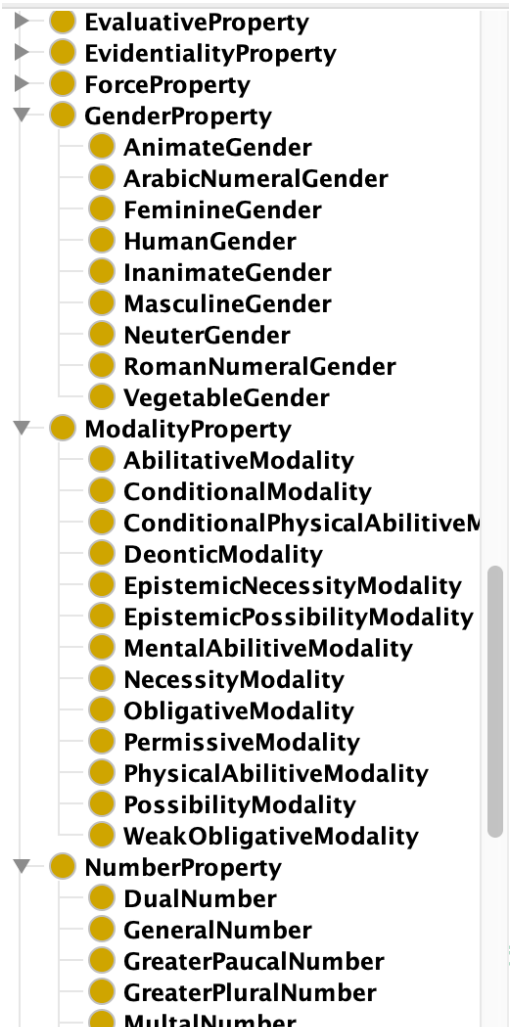


More linguistics for annotations?

Example: GOLD ontology [5]. Selection:







Two key options for the system

1. Source language → English → verbalise → translate to source language

   E.g.: use grammatical framework or some translator

Two key options for the system

1. Source language → English → verbalise → translate to source language

   E.g.: use grammatical framework or some translator

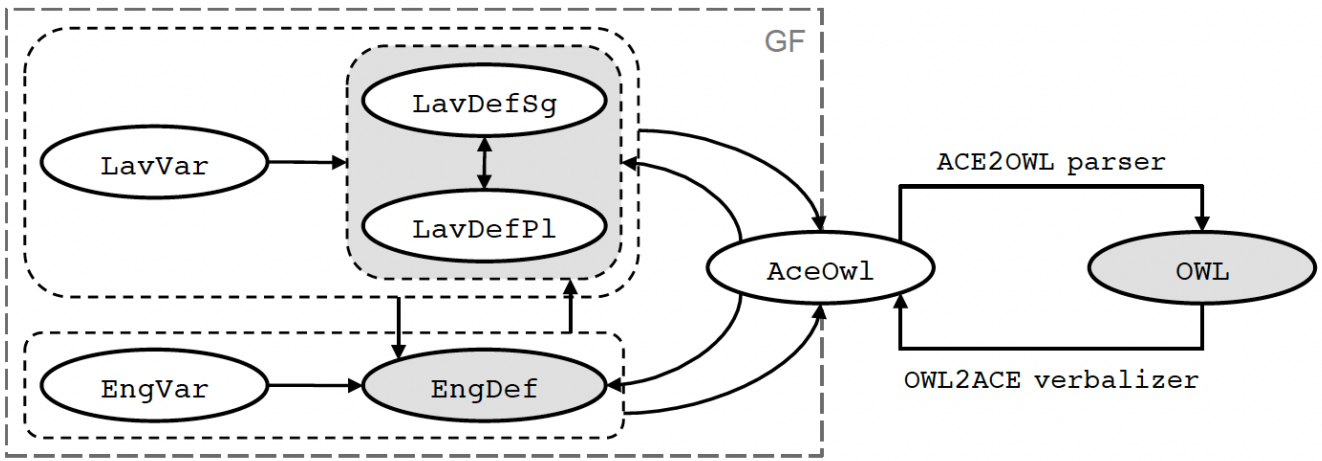
2. *de novo*, with language-specific additional data structures and rules

   Can reuse idea of pipeline modules

   Can reuse model for multilingual labels and attendant models (e.g., ontolex-lemon, [13], MoLA [9])

Example of a 'detour' approach [10]

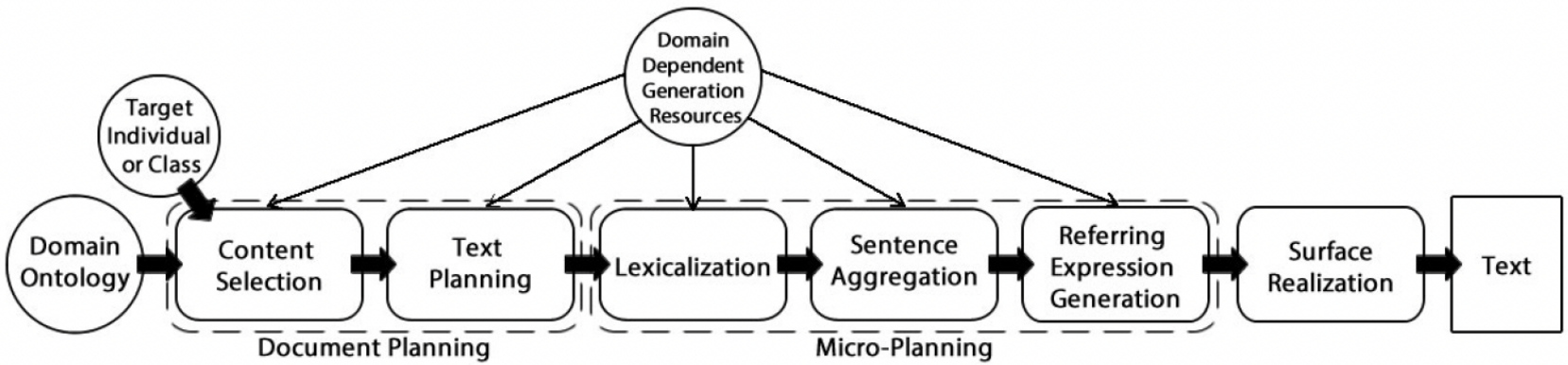
Depiction of the data flow that includes the automatic translations among the Latvian and English CNLs and OWL.



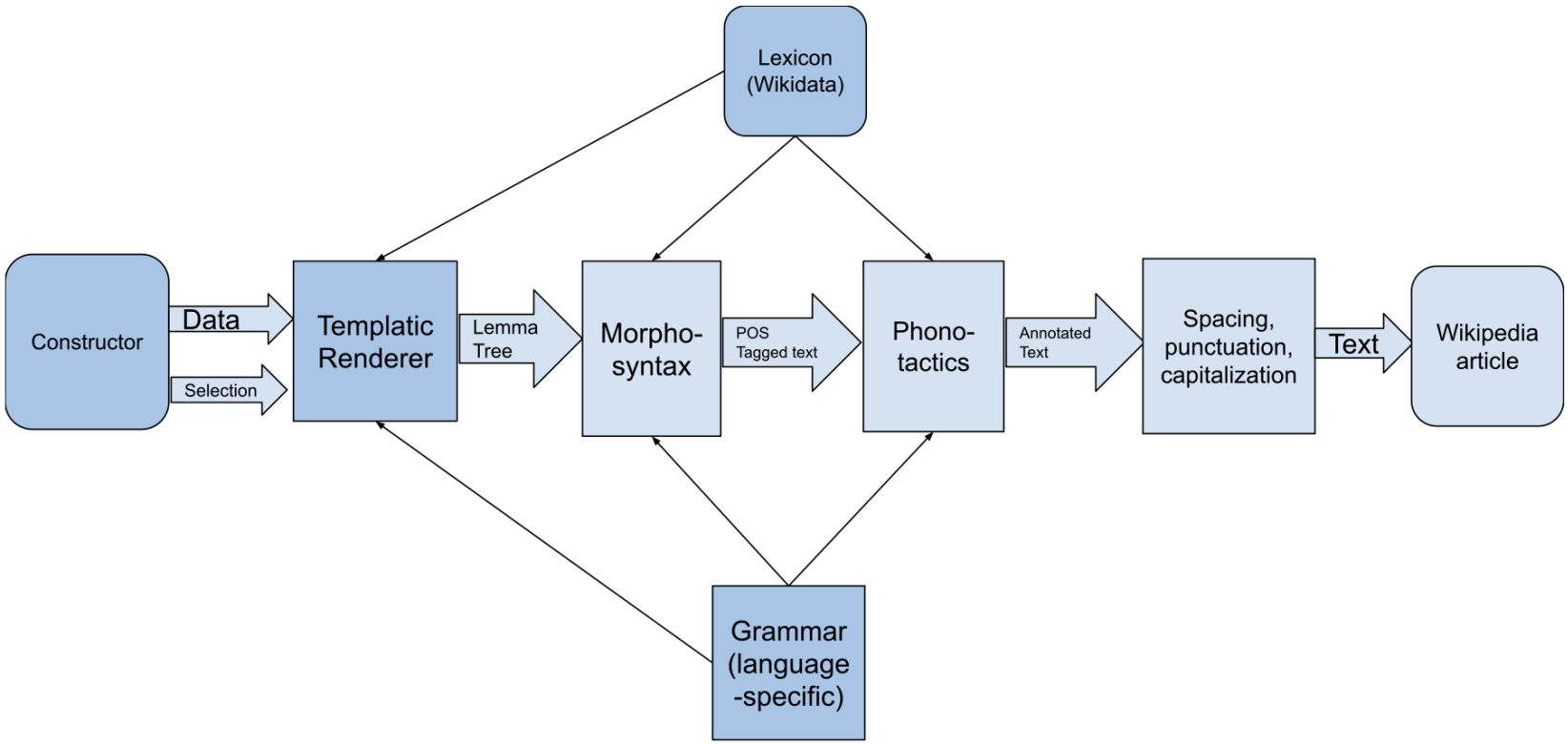
Tooling: ACE-OWL (to/from OWL) and the GF grammars for Latvian↔English.

Example of 'reuse' option [1]

NLG pipeline – with common modules – for generating museum text (in Greek) from ontologies.



Example of 'for any language' option (perhaps)



https://meta.wikimedia.org/wiki/Abstract_Wikipedia/Natural_language_generation_system_architecture_proposal (image by By Ariel Gutman)

Outline

Summary

- Core design choices for CNLs & NLG for ontologies
- Recurring hurdles
- Modelling patterns and styles affecting verbalisation
- Languages other than English, multilingualism
- A few applications and application scenarios

Collaborators and Funding

- Pablo Fillottrani, Ariel Gutman, Langa Khumalo
- Current/former students wrt NLG and ontologies: Mary-Jane Antia, Joan Byamugisha, Catherine Chavula, Takunda Chirema, Leighton Dawson, Francis Gillis-Webber, Zola Mahlaza, Sindiso Mkhatshwa, Junior Moraba, Gerald Ngumbulu, Toky Raboanary, Musa Xakaza, Steve Wang
- NRF grant (for this tutorial): MoRENL project `http://www.meteck.org/MoReNL/`

References I

I. Androutsopoulos, G. Lampouras, and D. Galanis.
Generating natural language descriptions from owl ontologies: the naturalowl system.
Journal of Artificial Intelligence Research, 48:671–715, 2013.

N. Bouayad-Agha, G. Casamayor, and L. Wanner.
Natural language generation in the context of the semantic web.
Semantic Web Journal, 5(6):493–513, 2014.

G. Braun, P. R. Fillottrani, and C. M. Keet.
A framework for interoperability between models with hybrid tools.
Journal of Intelligent Information Systems, yy(xx):in print, 2022.

V.K. Chaudhri, B. Cheng, A. Overholtzer, J. Roschelle, A. Spaulding, P. Clark, M. Greaves, and D Gunning.
Inquire biology: A textbook that answers questions.
AI Magazine, 34(3):55–72, 2013.

S. Farrar and D. T. Langendoen.
A linguistic ontology for the semantic web.
In *GLOT International*, volume 7 of *3*, pages 97–100, 2003.

Pablo R. Fillottrani and C. Maria Keet.
Patterns for heterogeneous tbox mappings to bridge different modelling decisions.
In E. Blomqvist et al., editors, *Proceeding of the 14th Extended Semantic Web Conference (ESWC'17)*, volume 10249 of *LNCS*, pages 371–386. Springer, 2017.
30 May - 1 June 2017, Portoroz, Slovenia.

References II

Enrico Franconi, Paolo Guagliardo, and Marco Trevisan.

An intelligent query interface based on ontology navigation.
In *Workshop on Visual Interfaces to the Social and Semantic Web (VISSW'10)*, 2010.
Hong Kong, February 2010.

A. Gatt and E. Reiter.

Simplenlg: A realisation engine for practical applications.
In E. Krahmer and M. Theune, editors, *Proceedings of the 12th European Workshop on Natural Language Generation (ENLG'09)*, pages 90–93. ACL, 2009.
March 30-31, 2009, Athens, Greece.

F. Gillis-Webber, S. Tittel, and C. M. Keet.

A model for language annotations on the web.
In *1st Iberoamerican conference on Knowledge Graphs and Semantic Web (KGSWC'19)*, volume 1029 of *CCIS*, pages 1–16. Springer, 2019.
24-28 June 2019, Villa Clara, Cuba.

Normunds Gruzitis and Guntis Barzdins.

Towards a more natural multilingual controlled language interface to OWL.
In *Proceedings of the Ninth International Conference on Computational Semantics*, IWCS '11, pages 335–339, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

Mustafa Jarrar, C. Maria Keet, and Paolo Dongilli.

Multilingual verbalization of ORM conceptual models and axiomatized ontologies.
Starlab technical report, Vrije Universiteit Brussel, Belgium, February 2006.

References III

C. M. Keet.

Natural language template selection for temporal constraints.
In *CREOL: Contextual Representation of Events and Objects in Language, Joint Ontology Workshops 2017*, volume 2050 of *CEUR-WS*, page 12, 2017.
21-23 September 2017, Bolzano, Italy.

C. M. Keet and T. Chirema.

A model for verbalising relations with roles in multiple languages.
In E. Blomqvist, P. Ciancarini, F. Poggi, and F. Vitali, editors, *Proceedings of the 20th International Conference on Knowledge Engineering and Knowledge Management (EKAW'16)*, volume 10024 of *LNAI*, pages 384–399. Springer, 2016.
19-23 November 2016, Bologna, Italy.

C. M. Keet and L. Khumalo.

Toward a knowledge-to-text controlled natural language of isiZulu.
Language Resources and Evaluation, 51(1):131–157, 2017.

C. Maria Keet and Langa Khumalo.

Basics for a grammar engine to verbalize logical theories in isiZulu.
In A. Bikakis et al., editors, *Proceedings of the 8th International Web Rule Symposium (RuleML'14)*, volume 8620 of *LNCS*, pages 216–225. Springer, 2014.
August 18-20, 2014, Prague, Czech Republic.

References IV

C. Maria Keet and Langa Khumalo.

Toward verbalizing logical theories in isiZulu.
In B. Davis, T. Kuhn, and K. Kaljurand, editors, *Proceedings of the 4th Workshop on Controlled Natural Language (CNL'14)*, volume 8625 of *LNAI*, pages 78–89. Springer, 2014.
20-22 August 2014, Galway, Ireland.

Tobias Kuhn.

A principled approach to grammars for controlled natural languages and predictive editors.
Journal of Logic, Language and Information, 22(1):33–70, 2013.

Z. Mahlaza and C. Maria Keet.

Formalisation and classification of grammar and template-mediated techniques to model and ontology verbalisation.
Int. J. Metadata, Semantics and Ontologies, 14(3):249–262, 2020.

Z. Mahlaza and C. Maria Keet.

Surface realisation architecture for low-resourced african languages.
Submitted to an international Journal, page xx, 2022.

Z. Mahlaza and C.M. Keet.

Toct: A task ontology to manage complex templates.
In Emilio M. Sanfilippo et al., editors, *FOIS 2021 Ontology Showcase, The Joint Ontology Workshops (JOWO'21)*, volume 2969 of *CEUR-WS*, 2021.

References V

T. Raboanary, S. Wang, and C. M. Keet.

Generating answerable questions from ontologies for educational exercises.
In E. Garoufallou, M-A. Ovalle-Perandones, and A Vlachidis, editors, *15th Metadata and Semantics Research Conference (MTSR'21)*, volume 1537 of *CCIS*, pages 28–40. Springer, 2021.
29 Nov - 3 Dec 2021, online.

Hazem Safwat and Brian Davis.

CNLs for the semantic web: a state of the art.
Language Resources & Evaluation, 51(1):191–220, 2017.

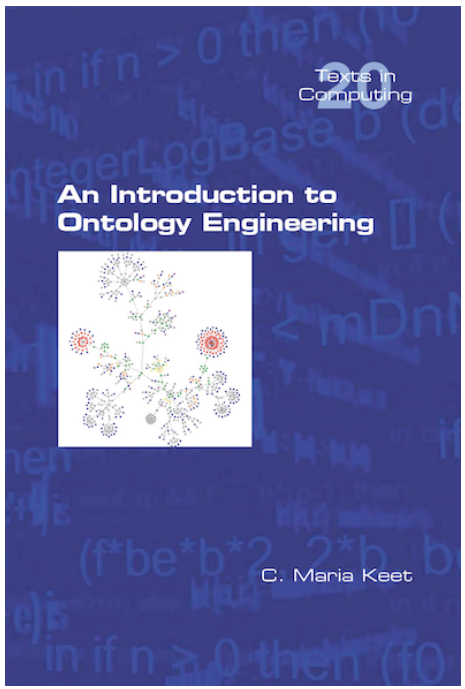
K. Winkler, T. Kuhn, and M Volk.

Evaluating the fully automatic multi-language translation of the swiss avalanche bulletin.
In B. Davis, T. Kuhn, and K. Kaljurand, editors, *Proceedings of the 4th Workshop on Controlled Natural Language (CNL'14)*, volume 8625 of *LNAI*, pages 44–54. Springer, 2014.
20-22 August 2014, Galway, Ireland.

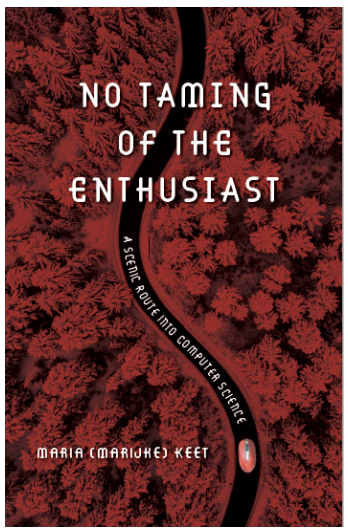
Thank you!

Questions?

My award-winning textbook



A memoir



Question

- Can we use any of these approaches for agglutinating Niger-Congo B languages?

Question

- Can we use any of these approaches for agglutinating Niger-Congo B languages?
  - It depends... but mostly: no

Question

- Can we use any of these approaches for agglutinating Niger-Congo B languages?
  - It depends... but mostly: no
- Tasks:
  - For structured input: use a practically useful language with tool support already (Semantic Web technologies)
  - Start with basics for a grammar engine (develop the new algorithms)
  - Pick an appealing sample domain (e.g., health)
  - Do it in a way so as to benefit both ICT and linguists

Question

- Can we use any of these approaches for agglutinating Niger-Congo B languages?
  - It depends... but mostly: no
- Tasks:
  - For structured input: use a practically useful language with tool support already (Semantic Web technologies)
  - Start with basics for a grammar engine (develop the new algorithms)
  - Pick an appealing sample domain (e.g., health)
  - Do it in a way so as to benefit both ICT and linguists
- First language to experiment with: isiZulu [15, 16, 14]

A logic foundation for isiZulu knowledge-to-text

- Roughly OWL 2 EL
- OWL 2 EL is a W3C-standardised profile of OWL 2
- Tools, ontologies in OWL 2 (notably SNOMED CT)

$\mathcal{ALC}$ syntax

- *Concepts* denoting entity types/classes/unary predicates/universals, including top $\top$ and bottom $\bot$;
- *Roles* denoting relationships/associations/n-ary predicates/properties;
- *Constructors*: and $\sqcap$, or $\sqcup$, and not $\neg$; quantifiers 'for all' $\forall$ and 'there exists' $\exists$
- *Complex concepts* using constructors: Let C and D be concept names, R a role name, then
  - $\neg C$, $C \sqcap D$, and $C \sqcup D$ are concepts, and
  - $\forall R.C$ and $\exists R.C$ are concepts
- *Individuals*
- e.g., $Lion \sqsubseteq \exists eats.Herbivore \sqcap \forall eats.Herbivore$

Universal Quantification

- Consider here only the universal quantification at the start of the concept inclusion axiom ('nominal head')
- 'all'/'each' uses *-onke*, prefixed with the oral prefix of the noun class of that first noun (OWL class/DL concept) on lhs of $\sqsubseteq$

(U1) `Boy` $\sqsubseteq$ ...

wonke umfana ...	('each boy...'; *u-* + *-onke*)
bonke abafana ...	('all boys...'; *ba-* + *-onke*)

(U2) `Phone` $\sqsubseteq$ ...

lonke ifoni ...	('each phone...'; *li-* + *-onke*)
onke amafoni ...	('all phones...'; *a-* + *-onke*)

NC	QC (all)		NEG SC	PRON	RC	QC_{dwa}	EC
	$QC_{oral+onke}$	QC_{nke}					
1	u-onke → wonke	wo-	aka-	yena	o-	ye-	mu-
2	ba-onke → bonke	bo-	aba-	bona	aba-	bo-	ba-
1a	u-onke → wonke	wo-	aka-	yena	o-	ye-	mu-
2a	ba-onke → bonke	bo-	aba-	bona	aba-	bo-	ba-
3a	u-onke → wonke	wo-	aka-	wona	o-	ye-	mu-
(2a)	ba-onke → bonke	bo-	aba-	bona	aba-	bo-	ba-
3	u-onke → wonke	wo-	awu-	wona	o-	wo-	mu-
4	i-onke → yonke	yo-	ayi-	yona	e-	yo-	mi-
5	li-onke → lonke	lo-	ali-	lona	eli-	lo-	li-
6	a-onke → onke	o-	awa-	wona	a-	wo-	ma-
7	si-onke → sonke	so-	asi-	sona	esi-	so-	si-
8	zi-onke → zonke	zo-	azi-	zona	ezi	zo-	zi-
9a	i-onke → yonke	yo-	ayi-	yona	e-	yo-	yi-
(6)	a-onke → onke	o-	awa-	wona	a-	wo-	ma-
9	i-onke → yonke	yo-	ayi-	yona	e-	yo-	yi-
10	zi-onke → zonke	zo-	azi-	zona	ezi-	zo-	zi-
11	lu-onke → lonke	lo-	alu-	lona	olu-	lo-	lu-
(10)	zi-onke → zonke	zo-	azi-	zona	ezi-	zo-	zi-
14	ba-onke → bonke	bo-	abu-	bona	obu-	bo-	bu-
15	ku-onke → konke	zo-	aku-	khona	oku-	zo-	ku-

NC	QC_{oral}	QC (all) -onke	QC_{nke}	NEG SC	PRON	RC	QC_{dwa}	EC
1	u-onke →	wonke	wo-	aka-	yena	o-	ye-	mu-
2	ba-onke →	bonke	bo-	aba-	bona	aba-	bo-	ba-
1a	u-onke →	wonke	wo-	aka-	yena	o-	ye-	mu-
2a	ba-onke →	bonke	bo-	aba-	bona	aba-	bo-	ba-
3a	u-onke →	wonke	wo-	aka-	wona	o-	ye-	mu-
(2a)	ba-onke →	bonke	bo-	aba-	bona	aba-	bo-	ba-
3	u-onke →	wonke	wo-	awu-	wona	o-	wo-	mu-
4	i-onke →	yonke	yo-	ayi-	yona	e-	yo-	mi-
5	li-onke →	lonke	lo-	ali-	lona	eli-	lo-	li-
6	a-onke →	onke	o-	awa-	wona	a-	wo-	ma-
7	si-onke →	sonke	so-	asi-	sona	esi-	so-	si-
8	zi-onke →	zonke	zo-	azi-	zona	ezi	zo-	zi-
9a	i-onke →	yonke	yo-	ayi-	yona	e-	yo-	yi-
(6)	a-onke →	onke	o-	awa-	wona	a-	wo-	ma-
9	i-onke →	yonke	yo-	ayi-	yona	e-	yo-	yi-
10	zi-onke →	zonke	zo-	azi-	zona	ezi-	zo-	zi-
11	lu-onke →	lonke	lo-	alu-	lona	olu-	lo-	lu-
(10)	zi-onke →	zonke	zo-	azi-	zona	ezi-	zo-	zi-
14	ba-onke →	bonke	bo-	abu-	bona	obu-	bo-	bu-
15	ku-onke →	konke	zo-	aku-	khona	oku-	zo-	ku-

Subsumption

- Two different ways of carving up the nouns to determine which rules apply: semantic and syntactic
- Need to choose between
  - singular and plural
  - with or without the universal quantification voiced
  - generic or determinate

(S1) `MedicinalHerb` $\sqsubseteq$ `Plant`

ikhambi ngumuthi ('medicinal herb is a plant')

amakhambi yimithi ('medicinal herbs are plants')

wonke amakhambi ngumuthi ('all medicinal herbs are a plant')

(S2) (generic)

(S3) (determinate)

Possible subsumption patterns

a. N_1 <copulative *ng*/*y* depending on first letter of N_2>N_2.

b. <plural of N_1> <copulative *ng*/*y* depending on first letter of plural of N_2><plural of N_2>.

c. <All-concord for NC_x> + onke <plural of N_1, being of NC_x> <copulative *ng*/*y* depending on first letter of N_2>N_2.

Existential Quantification

(E1) Giraffe $\sqsubseteq \exists$eats.Twig

yonke indlulamithi idla ihlamvana elilodwa ('each giraffe eats at least one twig')

zonke izindlulamithi zidla ihlamvana elilodwa ('all giraffes eat at least one twig')

a. <All-concord for NC_x> + onke <pl. N_1, is in NC_x> <conjugated verb> <N_2 of NC_y> <RC for NC_y><QC for NC_y> + dwa.

Example

- $\forall x\ (\text{Professor}(x) \rightarrow \exists y\ (\text{teaches}(x, y) \wedge \text{Course}(y)))$
- $\text{Professor} \sqsubseteq \exists\ \text{teaches}.\text{Course}$
- **Each** Professor teaches **at least one** Course

Example

- $\forall x\ (\text{uSolwazi}(x) \rightarrow \exists y\ (\text{-fundisa}(x, y) \wedge \text{Isifundo}(y)))$
- $\text{uSolwazi} \sqsubseteq \exists\ \text{-fundisa}.\text{Isifundo}$
- ?

$$\forall x\ (\text{uSolwazi}(x) \rightarrow \exists y\ (\text{-fundisa}(x, y) \wedge \text{Isifundo}(y)))$$

$$\text{uSolwazi} \sqsubseteq \exists\ \text{-fundisa}.\text{Isifundo}$$

$\forall x\ (\text{uSolwazi}(x) \rightarrow$ ... $(x, ..) \wedge$ ...$(..)))$

$\text{uSolwazi} \sqsubseteq \exists$ -func...

look-up NC

pluralise

for-all

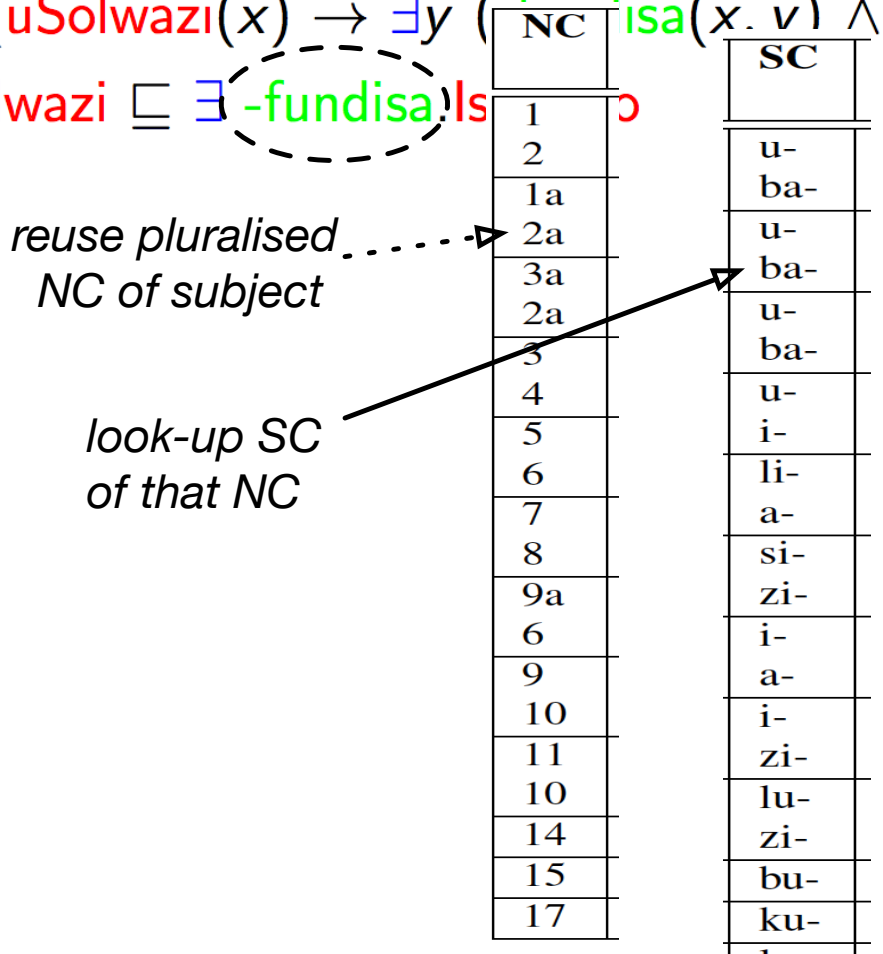
NC	AU	PRE
1	u-	m(u)-
2	a-	ba-
1a	u-	-
2a	o-	-
3a	u-	-
(2a)	o-	-
3	u-	m(u)-
4	i-	mi-
5	i-	(li)-
6	a-	ma-
7	i-	si-
8	i-	zi-
9a	i-	-
(6)	a-	ma-
9	i(n)-	-
10	i-	zi(n)-
11	u-	(lu)-
(10)	i-	zi(n)-
14	u-	bu-
15	u-	ku-
17		ku-

NC	QC (all) $QC_{oral+onke}$
1	u-onke → wonke
2	ba-onke → bonke
1a	u-onke → wonke
2a	ba-onke → bonke
3a	u-onke → wonke
(2a)	ba-onke → bonke
3	u-onke → wonke
4	i-onke → yonke
5	li-onke → lonke
6	a-onke → onke
7	si-onke → sonke
8	zi-onke → zonke
9a	i-onke → yonke
(6)	a-onke → onke
9	i-onke → yonke
10	zi-onke → zonke
11	lu-onke → lonke
(10)	zi-onke → zonke
14	ba-onke → bonke
15	ku-onke → konke

Bonke oSolwazi

$\forall x\ (\text{uSolwazi}(x) \rightarrow \exists y\ (\text{f}\ldots\text{isa}(x, y) \wedge \text{Isifundo}(y)))$

uSolwazi $\sqsubseteq \exists$ -fundisa.Is…



Bonke oSolwazi bafundisa

$$\forall x\ (\text{uSolwazi}(x) \rightarrow \exists y\ (\text{-fundisa}(x, y) \wedge \text{Isifundo}(y)))$$

$$\text{uSolwazi} \sqsubseteq \exists\ \text{-fundisa}.\text{Isifundo}$$

Bonke oSolwazi bafundisa Isifundo

$\forall x\ (\text{uSolwazi}(x) \rightarrow \exists y$) $\wedge$))

uSolwazi $\sqsubseteq \exists$ -fundisa.I

look-up NC

get RC

get QC

add -dwa

NC	AU	PRE
1	u-	m(u)-
2	a-	ba-
1a	u-	-
2a	o-	-
3a	u-	-
(2a)	o-	-
3	u-	m(u)-
4	i-	mi-
5	i-	(li)-
6	a-	ma-
7	i-	si-
8	i-	zi-
9a	i-	-
(6)	a-	ma-
9	i(n)-	-
10	i-	zi(n)-
11	u-	(lu)-
(10)	i-	zi(n)-
14	u-	bu-
15	u-	ku-
17		ku-

RC	QC_{dwa}
o-	ye-
aba-	bo-
o-	ye-
aba-	bo-
o-	ye-
aba-	bo-
o-	wo-
e-	yo-
eli-	lo-
a-	wo-
esi-	so-
ezi	zo-
e-	yo-
a-	wo-
e-	yo-
ezi-	zo-
olu-	lo-
ezi-	zo-
obu-	bo-
oku-	zo-

Bonke oSolwazi bafundisa Isifundo esisodwa