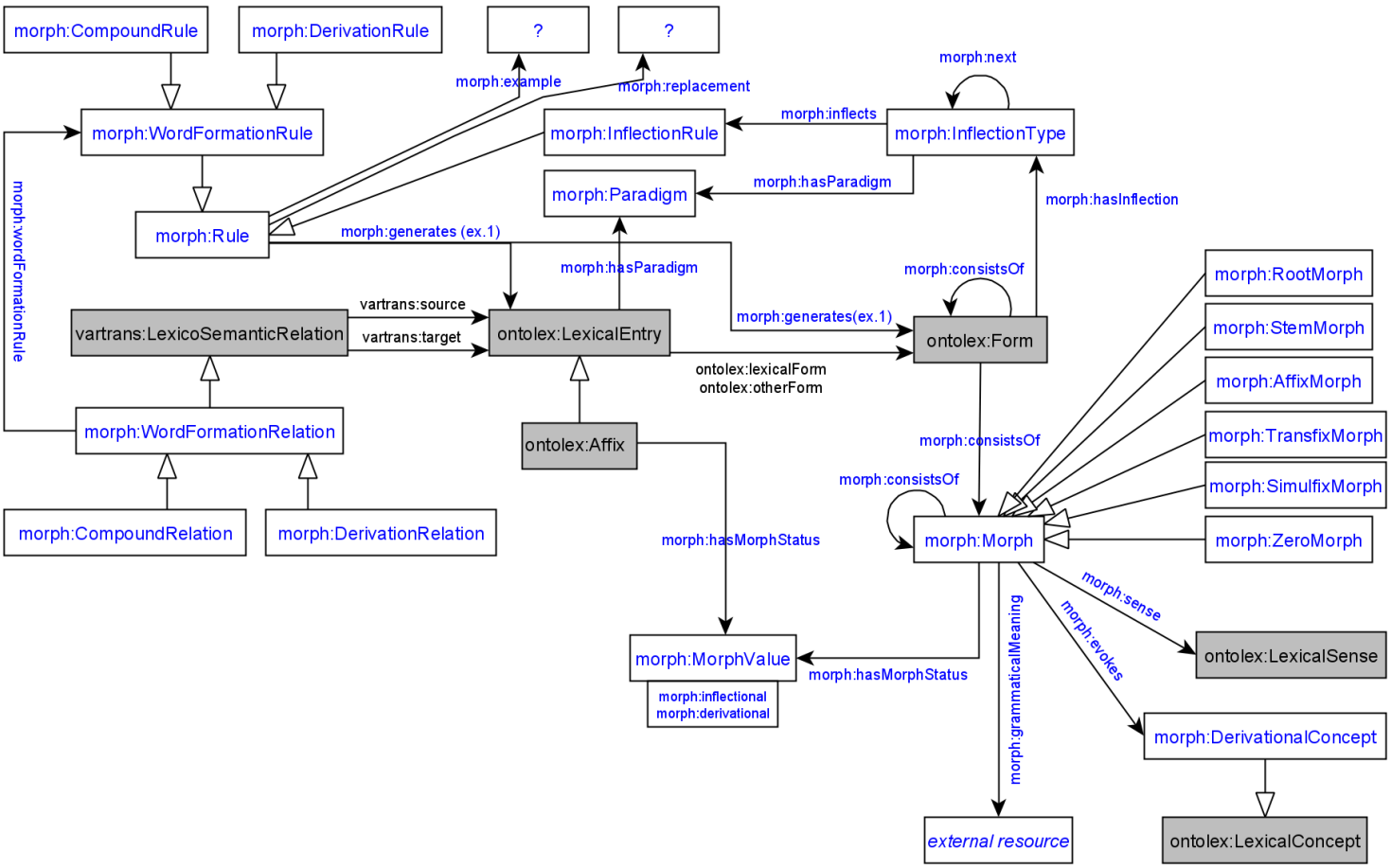
Participants:
Bettina Klimek (BK)
Jakub Šimek
Stefania Racioppa (DFKI)
Matteo Pellegrini
Fahad Khan
Katerina Gkirtzou

1. Module draft 4.3



Adaptions of module draft 4.3 to be included for next telco:

- turn morph:example and morph:replacement into datatype properties (rdf:string)
- remove morph:Rule superclass and interconnect morph:generates with the two subclasses as domain

General notes:
this is one way of distinguishing morph, morpheme, allomorph
http://www.sfu.ca/person/dearmond/323/323%20/323.morph.htm

- discuss: connection between morph:Morph and morph:Rule

- add as recommendation that input of morph:InflectionRule should be an ontolex:LexicalEntry with ontolex:canonicalForm

2. **Review Stefania's example, proposed solution to problem nb. 2 (Max)**

MI: regarding nb.1: in my proposal there was a link from a word to the inflection type of its first affix. Was it deliberately removed?

```
<#kissa> a ontolex:Word ;
ontolex:canonicalForm [ontolex:writtenRep "kissa"@fi];
morph:inflectionType <#finnish_noun_type_kissa> .

# this is more verbose
:lex_abdicatio a ontolex:LexicalEntry ;
   lexinfo:gender lexinfo:feminine ;
   lexinfo:partOfSpeech lexinfo:noun ;
   morph:hasParadigm :la-noun_3 ;
   morph:inflectionType :la-noun_3n-stem-var ; # this is what I added
   morph:inflectionType :la-noun_3n-no-stem-var ;
   ontolex:evokes :a00031 ;
   ontolex:canonicalForm [ ontolex:writtenRep "abdicatio"@la ] .


:la-noun_3n-stem-var a morph:InflectionType ;
   morph:hasParadigm :la-noun_3 ;
   morph:inflects :la-noun_3n_other_stem ,
      :la-noun_3n_form_1 ,
      [...] ;
   morph:next :la-noun_3n-infl .

:la-noun_3n_other_stem a morph:InflectionRule ;
   morph:generates [ :stem :oblique ] ;
   morph:replacement [ morph:source "$" ;
      morph:target "n" ] .

:la-noun_3n-infl a morph:InflectionType ;
   morph:hasParadigm :la-noun_3 ;
   morph:inflects :la-noun_3n_abl_m-f_pl ,
      :la-noun_3n_abl_m-f_sg ,
      [...] .
```

3. **German compound example data for evaluation (Stefania)**

Source

```
Frauenbewegung <6> :0: Bewegung <6> ## (1) Frauenbewegung
Frauenbewegung <6> :3: Frau <6> ## (1) Frauenbewegung
Abendgesellschaft <6> :0: Gesellschaft <6> ## (1) Abendgesellschaft
Abendgesellschaft <6> :3: Abend <7> ## (1) Abendgesellschaft
```

RDF output

```
:lex_frauenbewegung a ontolex:LexicalEntry ;
    decomp:component :lex_bewegung,
        :lex_frau ;
    lexinfo:partOfSpeech lexinfo:noun ;
    ontolex:canonicalForm [ ontolex:writtenRep "Frauenbewegung"@de ] ;
    ontolex:morphologicalPattern :de-noun_008 .

:lex_abendgesellschaft a ontolex:LexicalEntry ;
    decomp:component :lex_abend,
        :lex_gesellschaft ;
    lexinfo:gender lexinfo:feminine ;
    lexinfo:partOfSpeech lexinfo:noun ;
    morph:hasParadigm :de-noun_008 ;
    ontolex:canonicalForm [ ontolex:writtenRep "Abendgesellschaft"@de ] .
```

Components

```
:lex_abend a ontolex:LexicalEntry ;
    lexinfo:gender lexinfo:masculine ;
    lexinfo:partOfSpeech lexinfo:noun ;
    morph:hasParadigm :de-noun_017 ;
    ontolex:canonicalForm [ ontolex:writtenRep "Abend"@de ] .

:lex_bewegung a ontolex:LexicalEntry ;
    lexinfo:gender lexinfo:feminine ;
    lexinfo:partOfSpeech lexinfo:noun ;
    morph:hasParadigm :de-noun_008 ;
    ontolex:canonicalForm [ ontolex:writtenRep "Bewegung"@de ] .

:lex_frau a ontolex:LexicalEntry ;
    lexinfo:gender lexinfo:feminine ;
    lexinfo:partOfSpeech lexinfo:noun ;
    morph:hasParadigm :de-noun_008 ;
    ontolex:canonicalForm [ ontolex:writtenRep "Frau"@de ] .

:lex_gesellschaft a ontolex:LexicalEntry ;
    lexinfo:gender lexinfo:feminine ;
    lexinfo:partOfSpeech lexinfo:noun ;
    morph:hasParadigm :de-noun_008 ;
    ontolex:canonicalForm [ ontolex:writtenRep "Gesellschaft"@de ] .
```

Inflections

```
:de-noun_008 a morph:Paradigm ;
    rdfs:comment "German noun inflection" .

:de-noun_008-infl a morph:InflectionType ;
```

```
    morph:hasParadigm :de-noun_008 ;
    morph:inflects :de-noun_008_acc-dat-gen-nom_pl,
        :de-noun_008_acc-dat-gen-nom_sg .

:de-noun_008_acc-dat-gen-nom_pl a morph:InflectionRule ;
    morph:generates [ lexinfo:case lexinfo:accusative,
                lexinfo:dative,
                lexinfo:genitive,
                lexinfo:nominative ;
            lexinfo:number lexinfo:plural ] ;
    morph:replacement [ morph:source "$" ;
            morph:target "en" ] .

:de-noun_008_acc-dat-gen-nom_sg a morph:InflectionRule ;
    morph:generates [ lexinfo:case lexinfo:accusative,
                lexinfo:dative,
                lexinfo:genitive,
                lexinfo:nominative ;
            lexinfo:number lexinfo:singular ] ;
    morph:replacement [ morph:source "$" ;
            morph:target "" ] .
```

[similar for `:de-noun_017`]

Fahad:
so in the case of passerby we would have a relationship between passer and by as lexical entries but then we would have a relationsjip between morph(emes) for the plural 'passersby'
*word formation relationship

4. Finnish example data for evaluation (Max)

```
# Words

<#kissa>    a ontolex:Word ;
        ontolex:canonicalForm [ontolex:writtenRep "kissa"@fi];
        morph:inflectionType <#finnish_noun_type_kissa_number>,
<#finnish_noun_type_kissa_number_obl> . # the first category after the stem

# Inflection types
# They are combined with the `:next` property.
# This is actually quite usual for linguists: the same approach is used in the `lexc` formalism widely used in computational morphology (incl. xfst and foma)

<#finnish_noun_type_kissa_number> a morph:InflectionType ;
                        morph:hasParadigm <#regular_finnish_noun> ;
                        morph:next <#finnish_noun_type_kissa_case> ;
                        morph:inflects <#finnish_noun_type_kissa_sg>,
<#finnish_noun_type_kissa_pl> .
```

```
<#finnish_noun_type_kissa_case> a morph:InflectionType ;
                        morph:hasParadigm <#regular_finnish_noun> ;
                        morph:inflects <#finnish_noun_type_kissa_nom>,
<#finnish_noun_type_kissa_ine> .


<#finnish_noun_type_kissa_nom> a morph:InflectionRule ;
                       morph:inflectionType <#finnish_noun_type_kissa_case> ;
                       morph:generates [ lexinfo:case lexinfo:nominative ] ;
                       morph:replacement [ morph:source "$"; morph:target "" ] .

<#finnish_noun_type_kissa_sg> a morph:InflectionRule ;
                      morph:inflectionType <#finnish_noun_type_kissa_number> ;
                      morph:generates [ lexinfo:number lexinfo:singular ] ;
                      morph:replacement [morph:source "$", morph:target ""] . # we can
actually create nodes for widely used replacements, like _no replacement_

<#finnish_noun_type_kissa_ine> a morph:InflectionRule ;
                       morph:inflectionType <#finnish_noun_type_kissa_case> ;
                       morph:generates [ lexinfo:case lexinfo:inessive ]; # it's a convenient
example even though the inessive of a cat is a bit weird
                       morph:replacement [morph:source "$", morph:target "ssa"] .

<#finnish_noun_type_kissa_pl> a morph:InflectionRule ;
                      morph:inflectionType <#finnish_noun_type_kissa_number> ;
                      morph:generates [ lexinfo:number lexinfo:plural ] ;
                      morph:replacement [morph:source "$", morph:target "t"] .
```

Todos:

- when CC is present discuss if morph:Morph and morph:Rule are different and then examine a direct interconnection between both classes by looking at Inuktitut data
- Bettina: add telco URL and meeting cycle/time to Morphology Module Wiki page