**Participants:**

Bettina

Max

Christian C.

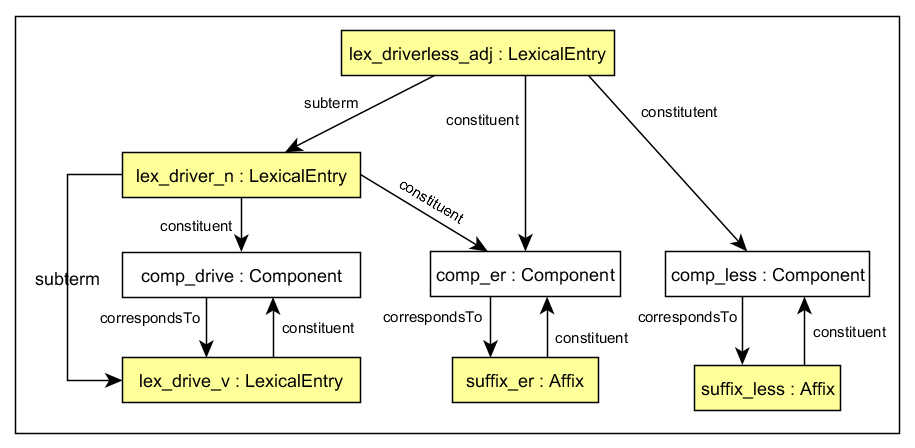
Maria

Fahad

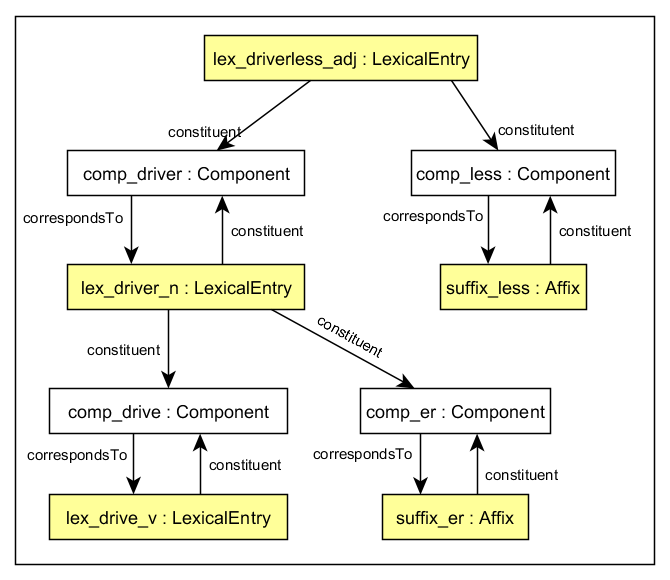
Thierry

John

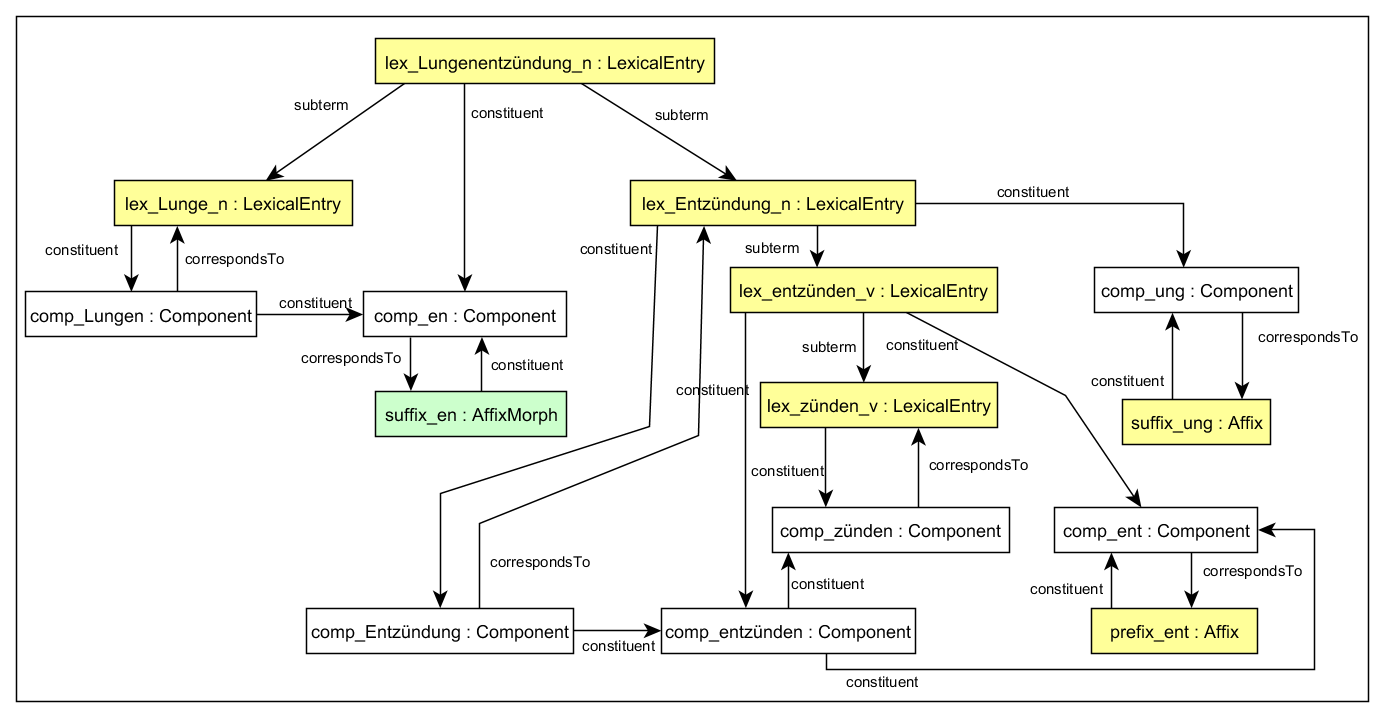
**E1a: English adjective *driverless*: full segmentation including 2 subterms and 3 Component constituents:**



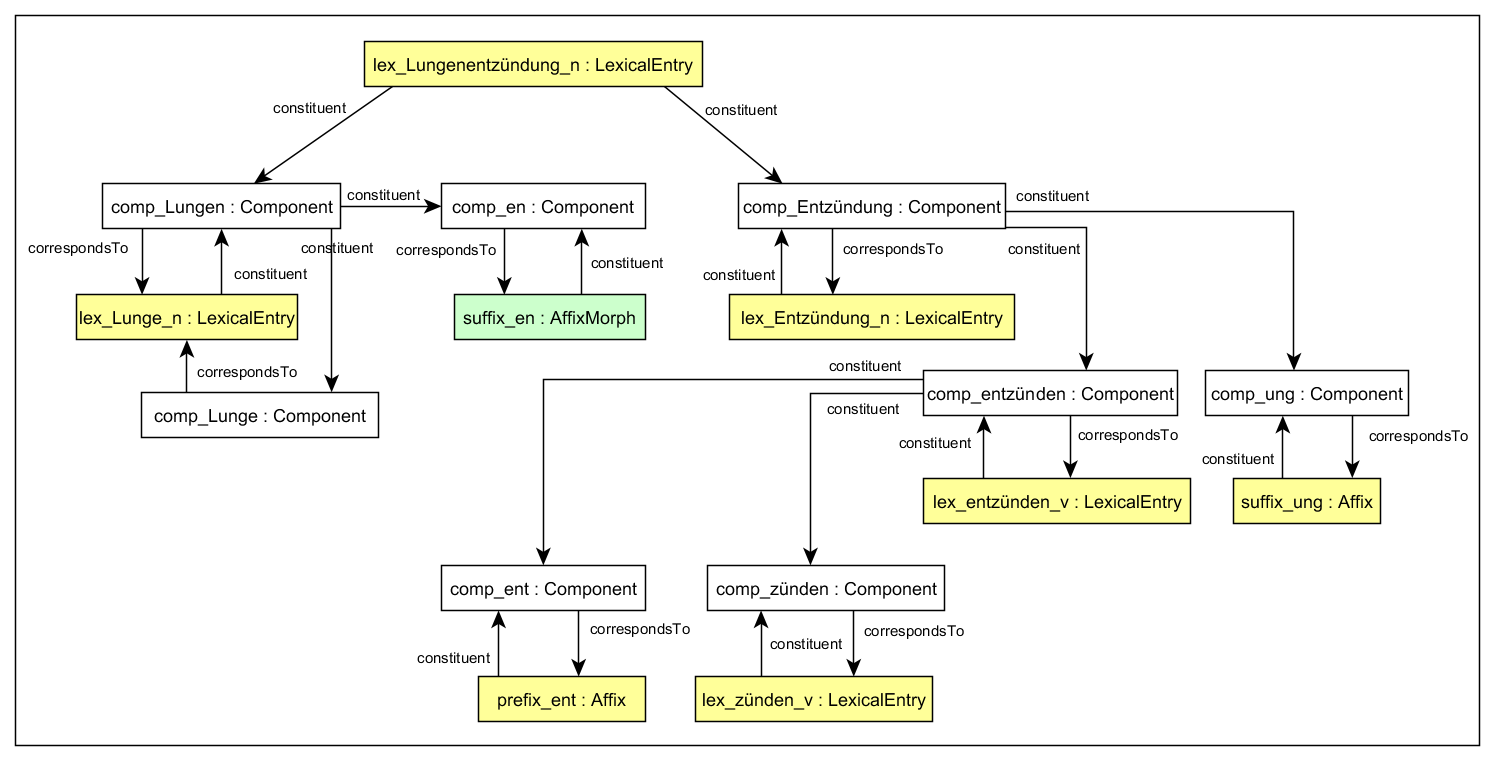
**E1b: English adjective *driverless*: binary segmentation involving 4 Component constituents (subterm relations could be added)**

****

**E2a: German compound *Lungenentzündung (*'pneumonia' literally 'lung inflammation'*)*: full segmentation including 4 subterms and 7 Component constituents:**

****

**E2b: German compound *Lungenentzündung (*'pneumonia' literally 'lung inflammation'*)*: binary segmentation involving 8 Component constituents (subterm relations could be added)**

****

Components apply to the character string in the Lexical Entry

1. Representing ordering of elements in lexical entries and morphs in forms

CC: segmentation ordering via dedicated properties rather than rdf list property, :next property

JM: Other options are RDF Seqs and RDF Lists, see <https://www.w3.org/TR/rdf-schema/#ch_containervocab>

With RDF Seqs one knows the starting point

Thierry: we are using RDF:\_1, RDF:\_2 (or so) in the decomp module

CC: If we want to specify the order of the components, we can use the RDF properties rdf:\_1, rdf:\_2, etc." (report).

Note that it says **can**, i.e., the LexicalEntry **can** be an instance of rdf:List, it doesn't have to be. And as an optional component, this is ok.

However, for decomposition, sequential order can normally confirmed from the surface string, for morphology, this is not the case, because there are more elements, and they are smaller:

German schön-er-er "a more beautiful (X)", first -er being comparative, second being inflection.

So, a systematic treatment of order is more essential for morphology than it is for decomposition.

Currently rdf\_1 applies to decomp:constituent, extend it to model discontinuous elements, order list of Components which correspond to Lexical Entries or Morphs

CC: take way of validation of OntoLex data into account, maybe using SHACL

Morph representation with using rdf:Label, exactly one label or allow more labels per instance?

ontolex:writtenRep does not work,

or use RDF:value