Reasonable Ontology Templates

Martin G. Skjæveland, Leif Harald Karlsen, Daniel P. Lupp Department of Informatics, University of Oslo

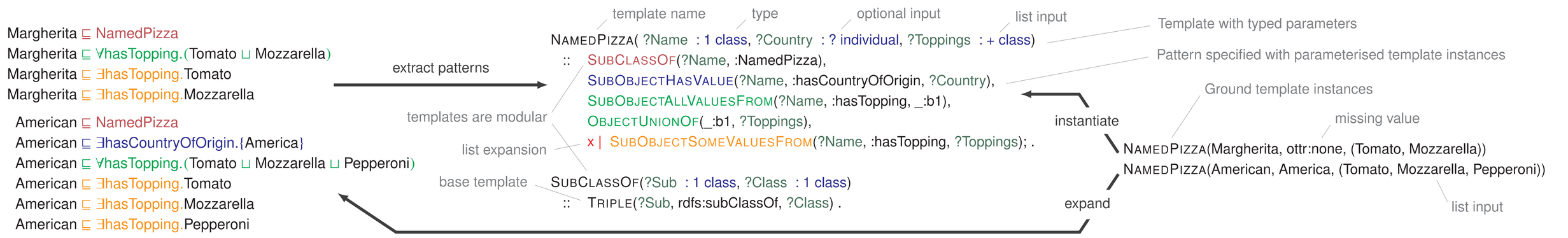
Problem

- Lack of established abstraction mechanism for RDF/OWL to capture modelling patterns
- Repetitive and tedious ontology construction
- Difficult to engage domain experts
- Low-level maintenance methods

Reasonable Ontology Templates (OTTR)

- Practical macro language for RDF/OWL
- Formal foundation, built with W3C standards
- Declarative and modular patterns
- Ensure uniform modelling
- Secure completeness of input
- Separate modelling patterns and bulk content
- Advanced template maintenance
- Available at `https://ottr.xyz` :
  - Open source Java implementation *Lutra*
  - Template library, specifications, demos

Construct ontologies by instantiating OTTR templates



Multiple serialisations: stOTTR (compact), tabOTTR (tabular), wOTTR (RDF), qOTTR* (SPARQL)

tabOTTR – Tabular format for data payload

#OTTR	prefix	
op	http://draft.ottr.xyz/pizza/	
	http://example.net/ns#	
#OTTR	**template**	op:NamedPizza
Name	Country	Toppings
1	2	3
iri	iri	iri+
:Margherita	:Italy	:Tomato | :Mozzarella
:American	:America	:Tomato | :Mozzarella | :Pepperoni
#OTTR	**template**	op:ItalianPizza
Name	Toppings	
1	2	
iri	iri+	
:QuattroFormaggi	:FourCheeses | :Tomato	
⋮	⋮	⋮

wOTTR – RDF format for templates and instances

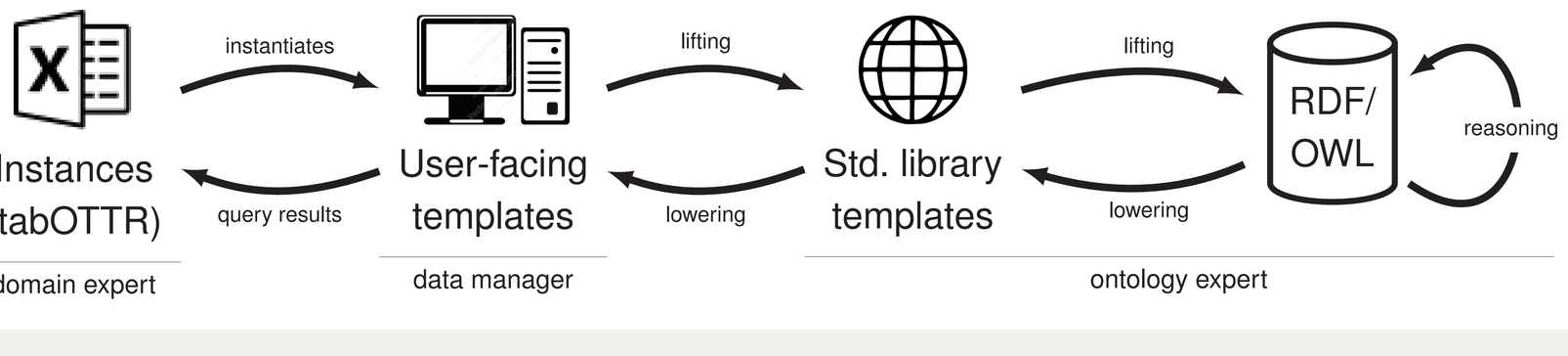
```
<http://draft.ottr.xyz/pizza/NamedPizza> a ottr:Template ;
 ottr:hasParameter
  [ ottr:index 1 ; ottr:classVariable :pizza ] ,
  [ ottr:index 2 ; ottr:individualVariable :country;
      ottr:optional true ] ,
  [ ottr:index 3 ; ottr:listVariable (:toppings) ]  .
   ### body:
[] ottr:templateRef t-owl-axiom:SubClassOf ;
     ottr:withValues ( :pizza p:NamedPizza ) .
[] ottr:templateRef t-owl-axiom:SubObjectHasValue ;
     ottr:withValues ( :pizza p:hasCountryOfOrigin :country ) .
[] ottr:templateRef t-owl-axiom:SubObjectAllValuesFrom ;
     ottr:withValues ( :pizza p:hasTopping _:alltoppings ) .
[] ottr:templateRef t-owl-rstr:ObjectUnionOf ;
     ottr:withValues ( _:alltoppings (:toppings) ) .
[] ottr:templateRef t-owl-axiom:SubObjectSomeValuesFrom ;
     ottr:hasArgument [ ottr:index 1; ottr:value :pizza ] ,
        [ ottr:index 2; ottr:value p:hasTopping ] ,
        [ ottr:index 3; ottr:eachValue (:toppings) ] .
```

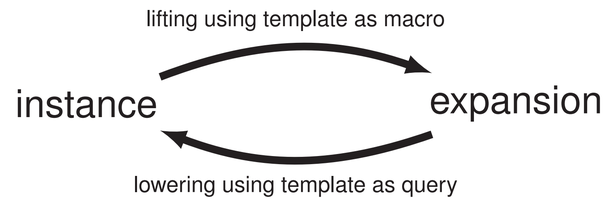
qOTTR – Templates as queries for extracting pattern instances

```
SELECT  *
  { ?param1   rdfs:subClassOf     p:NamedPizza ,
      [ owl:onProperty      p:hasTopping ;
        owl:someValuesFrom  param3item ;
        rdf:type            owl:Restriction ] ,
      [ owl:allValuesFrom   [ owl:unionOf  ?param3 ;
                              rdf:type     owl:Class ] ;
        owl:onProperty      p:hasTopping ;
        rdf:type            owl:Restriction ]
  OPTIONAL {
    ?param1   rdfs:subClassOf  [
        owl:hasValue        ?param2 ;
        owl:onProperty      p:hasCountryOfOrigin ;
        rdf:type            owl:Restriction ]
  }
    ?param3  (rdf:rest)*/rdf:first ?param3item
}
```

(* experimental)

Ontology engineering methodology: Template abstractions adapted to different user types





Maintain ontologies by maintaining templates: Detect and remove redundancies

Library with redundancies:

NamedPizza(?Name : 1 class, ?Country : ? individual, ?Toppings : + class)
:: SubClassOf(?Name, :NamedPizza),
× | SubObjectSomeValuesFrom(?Name, :hasTopping, ?Toppings),
SubObjectAllValuesFrom(?Name, :hasTopping, _:b1),
ObjectUnionOf(_:b1, ?Toppings),
SubObjectHasValue(?Name, :hasCountryOfOrigin, ?Country) .

Burger(?Name : 1 class, ?Condiments : + class, ?Label : + literal, ?PrefLabel : ? literal, ?Definition : ? literal)
:: SubClassOf(?Name, :Burger),
× | SubObjectSomeValuesFrom(?Name, :hasCondiment, ?Condiments),
SubObjectAllValuesFrom(?Name, :hasCondiment, _:b2),
ObjectUnionOf(_:b2, ?Condiments),
× | (?Name, rdfs:label, ?Label), (?Name, skos:prefLabel, ?PrefLabel), (?Name, skos:definition, ?Definition) .

Annotation(?Name : 1 class, ?Label : + literal, ?PrefLabel : ? literal, ?Definition : ? literal)
:: × | (?Name, rdfs:label, ?Label), (?Name, skos:prefLabel, ?PrefLabel), (?Name, skos:definition, ?Definition) .

Uncaptured pattern: Repeated use of a pattern not captured by a template → create new template and refactor
Lack of reuse: Use of pattern already captured by a template → refactor

Fixed library:

NamedPizza(?Name : 1 class, ?Country : ? individual, ?Toppings : + class)
:: SubObjectHasValue(?Name, :hasCountryOfOrigin, ?Country),
NamedFood(?Name, :NamedPizza,?Toppings,:hasTopping) .

Burger(?Name : 1 class, ?Condiments : + class, ?Label : + literal, ?PrefLabel : ? literal, ?Definition : ? literal)
:: NamedFood(?Name, :Burger,?Condiments,:hasCondiment),
Annotation(?Name,?Label, ?PrefLabel, ?Definition) .

NamedFood(?Name : 1 class, ?Category : 1 class, ?Extras : + class, ?hasExtra : 1 objectProperty)
:: SubClassOf(?Name, ?:Category),
× | SubObjectSomeValuesFrom(?Name, ?:hasExtra, ?Extras),
SubObjectAllValuesFrom(?Name, ?:hasExtra,_:b3),
ObjectUnionOf(_:b3, ?Extras) .

UiO : University of Oslo

Full paper available: Martin G. Skjæveland, Daniel P. Lupp, Leif Harald Karlsen, and Henrik Forssell. Practical Ontology Pattern Instantiation, Discovery, and Maintenance with Reasonable Ontology Templates. Accepted for ISWC 2018's research track.

Poster by Leif Harald and Martin