Lemon Design Patterns

John P. McCrae

Insight Centre for Data Analytics, National University of Ireland Galway
Why Design Patterns?

- Simplify usage of lemon
- Factorize common patterns
- Cover 95% of entries
- Expressible as simple language
Using the compiler

http://server1.nlp.insight-centre.org/lemonpatterns/

Documentation:

https://github.com/jmccrae/lemon.patterns
The Design Patterns Language

- Header consisting of prefix declarations
- Any number of lexicons
  - With a URI
  - And a language
- Any number of patterns
@prefix dbpedia: <http://dbpedia.org/resource/> .

Lexicon(#test>, "en",
    ClassNoun("test", dbpedia:Test))
Name, a proper noun referring to an individual

Name("Microsoft", <http://www.microsoft.com/>)

Class Noun, a noun referring to a class in the ontology

ClassNoun("test", dbpedia:Test)
If there is no class we may refer to a particular property value

```
ObjectPropertyNoun("oenophile", ontology:loves, ontology:wine)
```

Relational Noun, a noun with a prepositional phrase referring to a property in the ontology

```
RelationalNoun("father", dbpedia:father, propObj = PossessiveAdjunct)
```

We use PossessiveAdjunct to allow "X is father of Y" and "X is Y's father"
State Verb, a verb indicating a single property in the ontology

\textbf{StateVerb}("know", \text{foaf:knows})
Consequence Verb, a verb causing a property in the ontology

```python
ConsequenceVerb("die", dbpedia:deathDate,
    propObj = PrepositionalObject("on"))
```
Intersective Adjectives, an adjective referring to a class

`IntersectiveAdjective("Belgian", 
                     ontology:Belgian)`

`IntersectiveObjectPropertyAdjective("Belgian", 
                                        dbpedia-owl:nationality, dbpedia:Belgium)`

Relational Adjective, like relational nouns

Scalar Adjectives

`ScalarAdjective("big", 
                 [ontology:size > 5.0 for ontology:Building])`
Inflectional form, e.g.,

`ClassNoun("Katze", onto:Cat) with plural "Katzen"

Gender (nouns-only)

`ClassNoun("Katze", onto:Cat) feminine (with ...)"
Multiword terms can be specified using the following syntax

```
ClassNoun(["blood"/noun  "vessel"/noun],
          dbpedia:BloodVessel)
```
In your language model the following DBpedia entities:

- http://dbpedia.org/ontology/Person
- http://dbpedia.org/ontology/Surname
- http://dbpedia.org/resource/Germany
- http://dbpedia.org/property/spouse*
- People with profession = Historian
  http://dbpedia.org/property/profession
  http://dbpedia.org/resource/Historian

Step 1. What is the ontology type (class, property, individual)?
Step 2. What is the part of speech?
Step 3. What is the pattern?
Step 4. Write the Pattern

* Bonus: also model as ‘X is married to Y’
Lexicon(<>, "en",
    ClassNoun("person", dbo:Person),
    ClassNoun("surname", dbo:Surname),
    ClassNoun(["given"/noun "name"/noun], dbo:GivenName),
    ObjectPropertyNoun("historian", dbp:profession, dbr:Historian),
    Name("Germany", dbr:Germany),
    IntersectiveObjectPropertyAdjective("German", dbp:nationality, dbr:Germany),
    StateVerb("come", dbp:stateOfOrigin, propObj = PrepositionalObject("from"))
        with imperfect "came" with past participle "come",
    RelationalAdjective("related", dbp:relative, relationalArg = PrepositionalObject("to")),
    StateVerb("preceed",dbp:predecessor, propSubj = DirectObject, propObj = Subject),
    ConsequenceVerb("marry",dbp:spouse),
    RelationalNoun("husband",dbp:spouse,propObj = PossessiveAdjunct)
)