also be an instance of B. Only this will ensure that we respect the built-in semantics of primitives such as rdfs:subClassOf.

### 7.2.5 Define Properties

This step is often interleaved with the previous one: it is natural to organize the properties that link the classes while organizing these classes in a hierarchy.

Remember that the semantics of the subClassOf relation demands that whenever A is a subclass of B, every property statement that holds for instances of B must also apply to instances of A. Because of this inheritance, it makes sense to attach properties to the highest class in the hierarchy to which they apply.

While attaching properties to classes, it also makes sense to immediately provide statements about the domain and range of these properties. There is a methodological tension here between generality and specificity. On the one hand, it is attractive to give properties as general a domain and range as possible, enabling the properties to be used (through inheritance) by subclasses. On the other hand, it is useful to define domain and range as narrowly as possible, enabling us to detect potential inconsistencies and misconceptions in the ontology by spotting domain and range violations.

### 7.2.6 Define Facets

It is interesting to note that after all these steps, the ontology will only require the expressivity provided by RDF Schema and does not use any of the additional primitives in OWL. This will change in the current step, which is enriching the previously defined properties with facets:

- Cardinality. Specify for as many properties as possible whether they are allowed or required to have a certain number of different values. Often, occurring cases are “at least one value” (i.e., required properties) and “at most one value” (i.e., single-valued properties).