— a DOL library, or

— a `NativeDocument`, which is the verbatim inclusion of an OMS written in an OMS language that conforms with DOL; cf. 2.2).

A DOL library consists of a collection of (named) OMS, OMS networks, and mappings between these. More specifically, a DOL library consists of a name, followed by a list of `LibraryItems`. A `LibraryItem` is either a `Definition`, an import of another DOL library (`LibraryImport`), or a `Qualification` selecting a specific OMS language, logic and/or syntax that is used to interpret the subsequent `LibraryItems`. A `LibraryImport` leads to the inclusion of all `LibraryItems` of the imported DOL library into the importing one. A `Definition` assigns an IRI to an OMS (`OMSDefinition`), to a mapping between OMS (`MappingDefinition`), or an OMS network (`NetworkDefinition`). Moreover, annex L informatively introduces `QueryRelatedDefinition`.

At the beginning of a DOL library, one can declare a `PrefixMap` for abbreviating long IRIs using CURIEs; see clause 9.7 for further details. Examples of the use of DOL library can be found in Appendix M and Section 7.

9.3.2 Concrete Syntax

9.3.2.1 Documents

```
Document            ::= DOLLibrary | NativeDocument
DOLLibrary          ::= [PrefixMap] 'library' LibraryName
                            Qualification LibraryItem*
NativeDocument      ::= <language and serialization specific >
LibraryItem         ::= LibraryImport | Definition | Qualification
Definition          ::= OMSDefinition
                      | NetworkDefinition
                      | MappingDefinition
LibraryImport       ::= 'import' LibraryName
Qualification       ::= LanguageQualification
                      | LogicQualification
                      | SyntaxQualification
LanguageQualification ::= 'language' LanguageRef
LogicQualification ::= 'logic' LogicRef
SyntaxQualification ::= 'serialization' SyntaxRef
LibraryName         ::= IRI
LanguageRef         ::= IRI
LogicRef            ::= IRI
SyntaxRef           ::= IRI
```

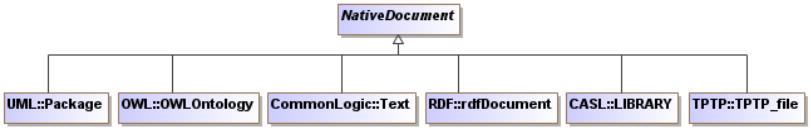

Figure 9.1 – Informative diagram showing subclasses of `NativeDocument` [new figure]

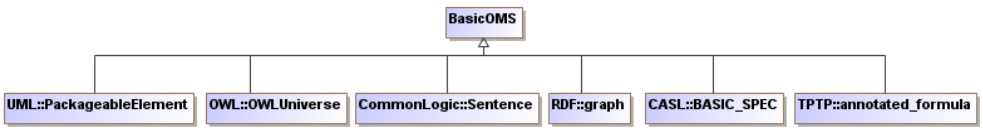

Figure 9.2 – Informative diagram showing subclasses of `BasicOMS` [new figure]

Annex C
(informative)
Conformance of OWL 2 DL With DOL

C.1 General

The semantic conformance of OWL 2 DL (as specified in W3C/TR REC-owl2-syntax:2012) with DOL is established in [56].

C.2 Abstract Syntax Conformance of OWL 2 With DOL

The metaclass `OWLOntology` **NR24**, 11.2 is a subclass (in the sense of SMOF multiple classification) of `NativeDocument`. The metaclass `OWLUniverse` **NR24**, 11.7 is a subclass (in the sense of SMOF multiple classification) of `BasicOMS`.

C.3 Conformance of the OWL Serializations With DOL

C.3.1 Text Conformance of the OWL 2 Manchester Syntax With DOL

The OWL 2 Manchester syntax satisfies the criteria for text conformance established in clause 2.3 in a straightforward way thanks to its line-based comment syntax (comments starting with #) and its flexible handling of line breaks.

C.3.2 Conformance of the XML and RDF Serializations of OWL With DOL

C.3.2.1 General Issues

With minor modifications detailed below, the OWL/XML serialization [25] satisfies the criteria for XML conformance and the serialization of OWL in RDF (**NR20**) satisfies RDF the criteria for RDF conformance. Both modifications define a super-language of the respective OWL serialization. Any OWL ontology serialization S' in one of these two super-languages can be translated into an OWL ontology serialization S that fully conforms to the original specification OWL/XML or "OWL serialized in RDF" and is semantically equivalent to the extended serialization S' with regard to the semantics of OWL. Without these modifications, neither OWL/XML nor "OWL serialized in RDF" satisfies the XML or RDF conformance requirements, respectively. The reason is that with imports there is a structural element supported by OWL that cannot have identifiers nor carry annotations, and that these two OWL serializations do not permit the use of XML or RDF constructs that would enable assigning identifiers to imports.

C.3.2.2 XML Conformance of a Modified OWL/XML With DOL

In the OWL/XML serialization, the *Import* element does not have annotations and is only allowed to carry the attributes *xml:base*, *xml:lang* and *xml:space*, but no further attributes or child elements from foreign namespaces (requirement (3b)), and therefore in particularly not a *dol:id* attribute or child elements, as would be required for adding identifiers (cf. clause 9.9).

An extended specification of OWL/XML that does allow the *dol:id* attribute on *Import* satisfies the XML conformance criteria. From an ontology serialized in this super-language of OWL/XML, one can obtain a semantically equivalent ontology (with regard to the semantics of OWL) by stripping all *dol:id* attributes.

Annex D
(informative)
Conformance of Common Logic with DOL

D.1 Abstract Syntax Conformance of Common Logic With DOL

The metaclass `Text` NR24, 12.2 is a subclass (in the sense of SMOF **NR26** multiple classification) of `NativeDocument`. The metaclass `Sentence` NR24, 12.2 is a subclass (in the sense of SMOF **NR26** multiple classification) of `BasicOMS`.

D.2 Serialization Conformance of Common Logic With DOL

The semantic conformance of Common Logic (as specified in **NR7**) with DOL is established in [56].

The XCF dialect of Common Logic has a serialization that satisfies the criteria for XML conformance. The CLIF dialect of Common Logic has a serialization that satisfies the criteria for text conformance.

D.3 Semantic Conformance of Common Logic With DOL

Common Logic can be defined as an institution as follows:

Definition 22 ***Common Logic.*** *A common logic signature Σ (called vocabulary in Common Logic terminology) consists of a set of names, with a subset called the set of discourse names, and a set of sequence markers. An signature morphism maps names and sequence markers separately, subject to the requirement that a name is a discourse name in the smaller signature if and only if it is one in the larger signature. A Σ-model $I = (UR, UD, rel, fun, int, seq)$ consists of a set UR, the universe of reference, with a non-empty subset $UD \subseteq UR$, the universe of discourse, and four mappings:*

- *rel from UR to subsets of $UD^* = \{\langle x_1, \ldots, x_n\rangle \mid x_1, \ldots, x_n \in UD\}$ (i.e., the set of finite sequences of elements of UD);*
- *fun from UR to total functions from UD^* into UD;*
- *int from names in Σ to UR, such that $int(v)$ is in UD if and only if v is a discourse name;*
- *seq from sequence markers in Σ to UD^*.*

A Σ-sentence is a first-order sentence, where predications and function applications are written in a higher-order like syntax: $t(s)$. Here, t is an arbitrary term, and s is a sequence term, which can be a sequence of terms $t_1 \ldots t_n$, or a sequence marker. A predication $t(s)$ is interpreted by evaluating the term t, mapping it to a relation using rel, and then asking whether the sequence given by the interpretation s is in this relation. Similarly, a function application $t(s)$ is interpreted using fun. Otherwise, interpretation of terms and formulae is as in first-order logic. A further difference to first-order logic is the presence of sequence terms (namely sequence markers and juxtapositions of terms), which denote sequences in UD^*, with term juxtaposition interpreted by sequence concatenation. Note that sequences are essentially a non-first-order feature that can be expressed in second-order logic.

Model reducts are defined in the following way: Given a signature morphism $\sigma : \Sigma_1 \to \Sigma_2$ and a Σ_2-model $I_2 = (UR, UD, rel, fun, int, seq)$, $I|_\sigma = (UR, UD, rel, fun, int \circ \sigma, seq \circ \sigma)$.

Given two CL models $I_1 = (UR_1, UD_1, rel_1, fun_1, int_1, seq_1)$ and $I_2 = (UR_2, UD_2, rel_2, fun_2, int_2, seq_2)$, a homomorphism $h : I_1 \to I_2$ is a function $h : UR_1 \to UR_2$ such that

- *h restricts to $k : UD_1 \to UD_2$,*
- *for each $x \in UR_1$ and $s \in UD_1^*$, if $s \in rel_1(x)$, then $k^*(s) \in rel_2(h(x))$[33],*
- *for each $x \in UR_1$, $k \circ fun_1(x) = fun_2(h(x)) \circ k^*$,*

[33] k^* is the extension of h to sequences.

Annex E
(informative)
Conformance of RDF and RDF Schema with DOL

E.1 Abstract Syntax Conformance of RDF and RDF Schema With DOL

The metaclass `rdfDocument` **NR24**, 14.2.2 is a subclass (in the sense of SMOF **NR26** multiple classification) of `NativeDocument`. The metaclass `graph` **NR24**, 14.2.3 is a subclass (in the sense of SMOF **NR26** multiple classification) of `BasicOMS`.

E.2 Serialization Conformance of RDF and RDF Schema With DOL

The way of representing RDF Schema ontologies as RDF graphs satisfies the criteria for RDF conformance.

E.3 Semantic Conformance of RDF and RDF Schema With DOL

The semantic conformance of RDF Schema (as specified in **NR18**) with DOL is established in [56].

Definition 23 (RDF and RDF Schema) *The institutions for the Resource Description Framework (RDF) and RDF Schema (also known as RDFS), respectively, are defined in the following [44]. Both RDF and RDFS are based on a logic called* bare RDF *(SimpleRDF), which consists of triples only (without any predefined resources).*

A signature $\mathbf{R_s}$ *in SimpleRDF is a set of resource references. For* $sub, pred, obj \in \mathbf{R_s}$*, a triple of the form* $(sub, pred, obj)$ *is a sentence in SimpleRDF, where sub, pred, obj represent subject name, predicate name, object name, respectively. An* $\mathbf{R_s}$*-model* $M = \langle R_m, P_m, S_m, EXT_m \rangle$ *consists of a set* R_m *of resources, a set* $P_m \subseteq R_m$ *of predicates, a mapping function* $S_m : \mathbf{R_s} \to R_m$*, and an extension function* $EXT_m : P_m \to \mathcal{P}(R_m \times R_m)$ *mapping every predicate to a set of pairs of resources. Satisfaction is defined as follows:*

$$\mathfrak{M} \models_{\mathbf{R_s}} (sub, pred, obj) \Leftrightarrow (S_m(sub), (S_m(obj)) \in EXT_m(S_m(pred)).$$

Both RDF and RDFS are built on top of SimpleRDF by fixing a certain standard vocabulary both as part of each signature and in the models.

Actually, the standard vocabulary is given by a certain theory. In case of RDF, it contains e.g. resources `rdf:type` *and* `rdf:Property` *and* `rdf:subject`*, and sentences like, e.g.*

$$(\texttt{rdf:type}, \texttt{rdf:type}, \texttt{rdf:Property}) \text{ , and}$$
$$(\texttt{rdf:subject}, \texttt{rdf:type}, \texttt{rdf:Property}) \text{ .}$$

In the models, the standard vocabulary is interpreted with a fixed model. Moreover, for each RDF-model $M = \langle R_m, P_m, S_m, EXT_m \rangle$*, if* $p \in P_m$*, then it must hold* $(p, S_m(\texttt{rdf:Property})) \in EXT_m(\texttt{rdf:type})$*. For RDFS, similar conditions are formulated (here, for example also the subclass relation is fixed).*

In the case of RDFS, the standard vocabulary contains more elements, like `rdfs:domain`*,* `rdfs:range`*,* `rdfs:Resource`*,* `rdfs:Literal`*,* `rdfs:Datatype`*,* `rdfs:Class`*,* `rdfs:subClassOf`*,* `rdfs:subPropertyOf`*,* `rdfs:member`*,* `rdfs:Container`*,* `rdfs:ContainerMembershipProperty`*.*

There is also OWL Full, an extension of RDFS with resources such as `owl:Thing` *and* `owl:oneOf`*, tailored towards the representation of OWL [24].* □

Annex F
(informative)
Conformance of UML class and object models with DOL

F.1 General

This informative annex demonstrates conformance of a subset of UML class and object models with DOL by defining an institution for both. The subset is restricted to the static aspects of class models; that is, change of state is ignored. This means that all operations are query operations.

F.2 Abstract Syntax Conformance of UML With DOL

The metaclass `Package` NR8 is a subclass (in the sense of SMOF **NR26** multiple classification) of `NativeDocument`. The metaclass `PackageableElement` NR8 is a subclass (in the sense of SMOF **NR26** multiple classification) of `BasicOMS`.

F.3 Serialization Conformance of UML With DOL

The XMI **NR27** serialization, derived from the MOF metamodel, is widely used for UML. Hence, UML is serialization conformant with DOL.

F.4 Semantic Conformance of UML With DOL

The institution of UML class and object models is defined using a translation of UML class models to Common Logic, following the fUML specification and [69].

F.4.1 Preliminaries

The axioms for primitive types are imported from the fUML specification, section 10.3.1: Booleans, numbers, sequences and strings. These axiomatize (among others) predicates corresponding to primitive types, e.g. `buml:Boolean`, `form:Number`, `form:NaturalNumber`, `buml:Integer`, `form:Sequence`, `form:Character`, and `buml:String`.

The following infrastructure, consisting off a number of predicates axiomatized in Common Logic, provides a foundation for an institution for UML class models described in the later sections of this Annex.

```
logic CLIF

oms pairs =
  (forall (x y) (= (form:first (form:pair x y)) x))
  (forall (x y) (= (form:second (form:pair x y)) y))
  (forall (x y) (form:Pair (form:pair x y)))
  (forall (p) (if (form:Pair p)
                  (= (form:pair (form:first p) (form:second p)) p)))
end

oms sequences =
  fuml:sequences.clif and pairs
then
  // Membership of an element in a sequence
  (forall (x s)
          (if (form:sequence-member x s)
              (form:Sequence s)))
  (forall (x s)
          (iff (form:sequence-member x s)
               (exists (p)
```

Annex G
(informative)
Conformance of TPTP with DOL

G.1 General

TPTP [73, 75, 74] is a language spoken by dozens of first-order theorem provers, and large libraries have been formalized in TPTP. The underlying logic is unsorted first-order logic.

G.2 Abstract Syntax Conformance of TPTP With DOL

The ~~metaclass~~ BNF nonterminal `TPTP_file` of the TPTP concrete syntax [72] is construed as a metaclass, and as such it is a subclass (in the sense of SMOF **NR26** multiple classification) of `NativeDocument`. The ~~metaclass~~ BNF nonterminal `annotated_formula` of the TPTP concrete syntax [72] is construed as a metaclass, and as such is a subclass (in the sense of SMOF **NR26** multiple classification) of `BasicOMS`.

G.3 Serialization Conformance of TPTP With DOL

The TPTP text syntax is text conformant with DOL.

G.4 Semantic Conformance of TPTP With DOL

In [18], many-sorted first has been formalized as an institution; the single-sorted sublogic (using only a fixed set of sorts $\{s\}$ is isomorphic to unsorted first-order logic).

Annex H
(informative)
Conformance of CASL with DOL

H.1 General

CASL [10] extends many-sorted first-order logic with partial functions and subsorting. It also provides induction sentences, expressing the (free) generation of datatypes.

H.2 Abstract Syntax Conformance of CASL With DOL

The ~~metaclass~~ EBNF nonterminal `LIBRARY` for the CASLabstract syntax [10] is construed as a metaclass, and as such it is a subclass (in the sense of SMOF **NR26** multiple classification) of `NativeDocument`. The ~~metaclass~~ EBNF nonterminal `BASIC_SPEC` for the CASLabstract syntax [10] is construed as a metaclass, and as such it is a subclass (in the sense of SMOF **NR26** multiple classification) of `BasicOMS`.

H.3 Serialization Conformance of CASL With DOL

The CASL text syntax is text conformant with DOL.

H.4 Semantic Conformance of CASL With DOL

CASL has been presented as an institution in [52, 10]. This section presents a sketch of this institution.

CASL signatures consist of a set S of sorts with a subsort relation $\leq$ between them together with families $\{PF_{w,s}\}_{w\in S^*,s\in S}$ of partial functions, $\{TF_{w,s}\}_{w\in S^*,s\in S}$ of total functions and $\{P_w\}_{w\in S^*}$ of predicate symbols. If Σ is a signature, two operation symbols with the same name f and with profiles $w \to s$ and $w' \to s'$, denoted $f_{w,s}$ and $f_{w',s'}$, are in the overloading relation if there are $w_0 \in S^*$ and $s_0 \in S$ such that $w_0 \leq w, w'$ and $s, s' \leq s_0$. Overloading of predicates is defined in a similar way. Signature morphisms consist of maps taking sort, function and predicate symbols respectively to a symbol of the same kind in the target signature, and they must preserve subsorting, typing of function and predicate symbols and totality of function symbols, and overloading.

For a signature Σ, terms are formed starting with variables from a sorted set X using applications of function symbols to terms of appropriate sorts, while sentences are partial first-order formulas extended with *sort generation constraints* which are triples (S', F', σ') such that $\sigma' : \Sigma' \to \Sigma$ and S' and F' are respectively sort and function symbols of Σ'. Partial first-order formulas are translated along a signature morphism $\varphi : \Sigma \to \Sigma''$ by replacing symbols as prescribed by φ while sort generation constraints are translated by composing the morphism σ' in their third component with φ.

Models interpret sorts as nonempty sets such that subsorts are injected into supersorts, partial/total function symbols as partial/total functions and predicate symbols as relations, such that the embeddings of subsorts into supersorts are monotone w.r.t. overloading.

The satisfaction relation is the expected one for partial first-order sentences. A sort generation constraint (S', F', σ') holds in a model M if the carriers of the reduct of M along σ' of the sorts in S' are generated by function symbols in F'.