Ontology Metadata Vocabulary - OMV

Tuesday, 18 September 2007

Core and Extensions:

Following the usability constraints identifies during the requirements analysis, we decided to design the OMV scheme modularly; OMV distinguishes between the OMV Core and various OMV Extensions. The former captures information which is expected to be relevant to the majority of ontology reuse settings. However, in order to allow ontology developers and users to specify task- or application-specific ontology-related information we foresee the development of OMV extension modules, which are physically separated from the core scheme, while remaining compatible to its elements.

Ontological representation:

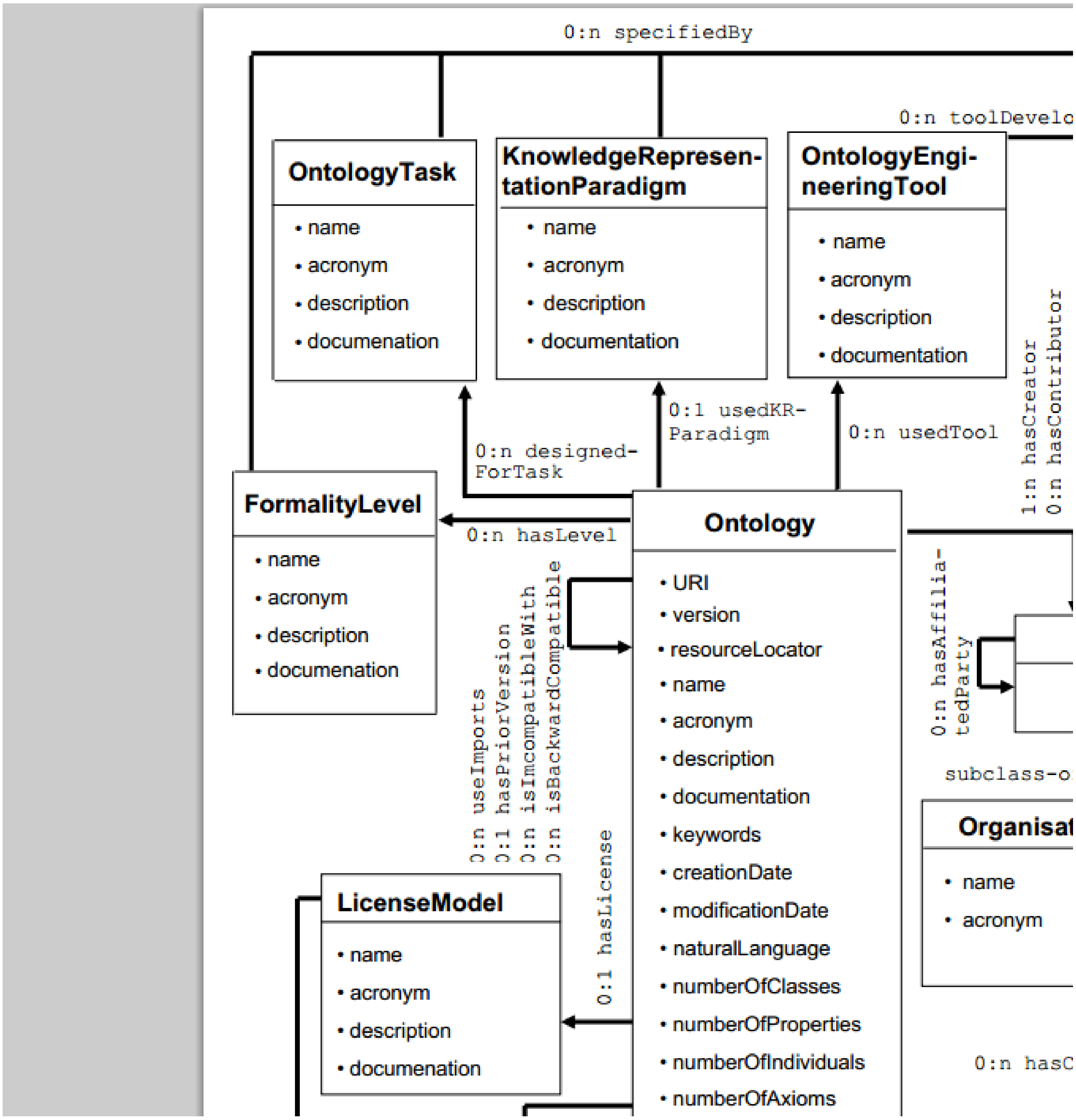
Due to the high accessibility and interoperability requirements, as well as the nature of the metadata, which is intended to describe Semantic Web ontologies, the conceptual model designed in the previous step was implemented in the OWL language. An implementation as XML-Schema or DTD was estimated to restrict the functionality of the ontology management tools using the metadata information (mainly in terms of retrieval capabilities) and to impede metadata exchange at semantical level. Further on, a language such as RDFS does not provide a means to distinguish between required and optional metadata properties. The implementation was performed manually by means of a common ontology editor.

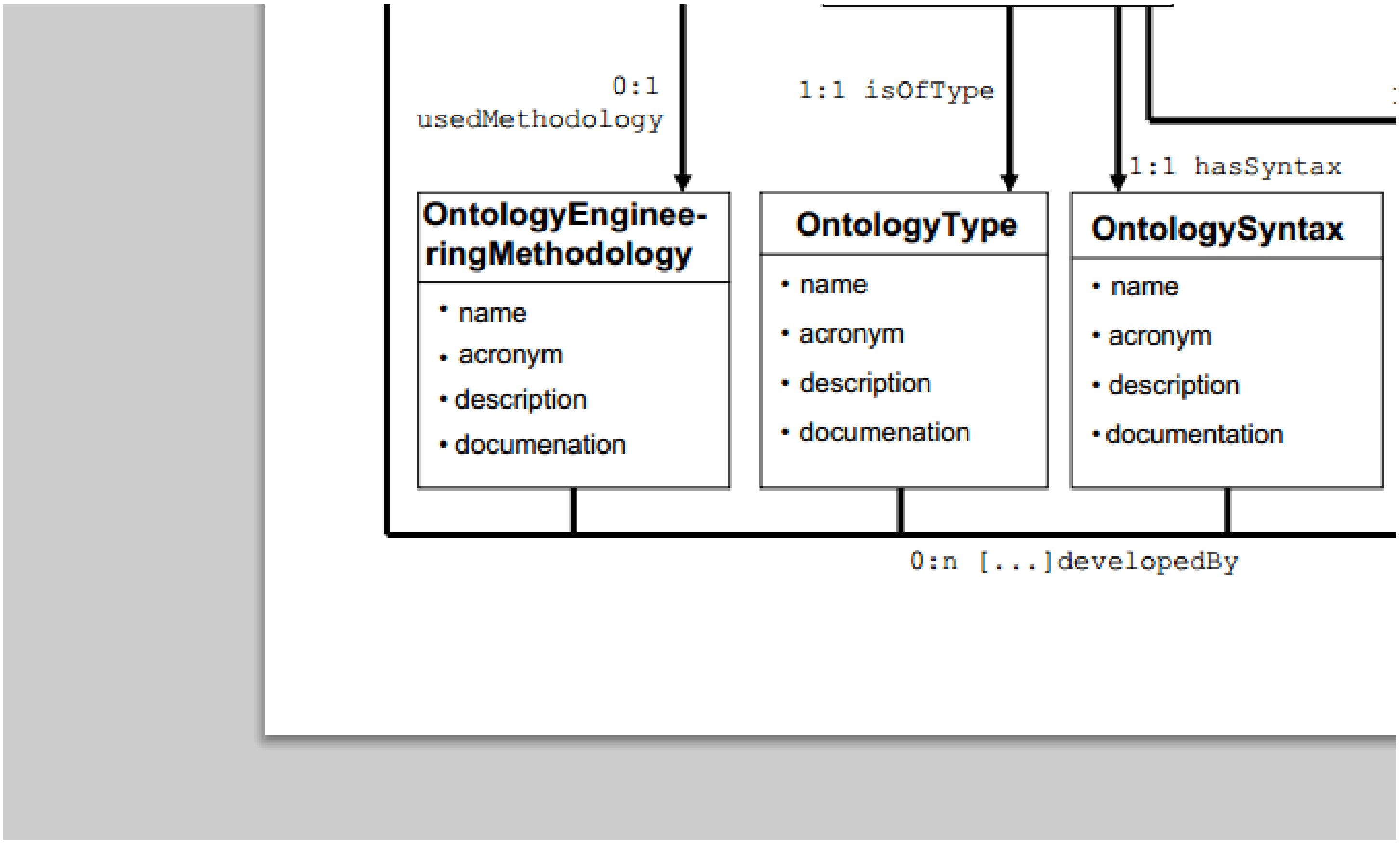
Identification, Versioning and Location:

In OMV we describesa particular representation of an ontology, i.e. an ontology in a particular version at a particular physical location. That means that every different version of an ontology has different OMV related metadata and consequently, a particular OMV instance is identified by the URI plus the version of the ontology it is describing. In addition to the issue of versioning, an ontology (or a version of an ontology) can be located at different locations.

Thus, ontologies with the same logical URI may exist at different physical locations, possibly even with different content. Similar to approach for versioning, we rely on a composite identifier consisting of the logical identifier (URI plus optional version identifier) and a resource locator that specifies the actual physical location.

Of course, the optional version identifier and the optional resource locator can be combined, such that we end up with a tripartite identifier (URI, version, resource locator).




0:1
usedMethodology
1:1 isOfType
1:1 hasSyntax
OntologyEnginee-ringMethodology
• name
• acronym
• description
• documenation
OntologyType
• name
• acronym
• description
• documenation
OntologySyntax
• name
• acronym
• description
• documentation
0:n [...]developedBy