KnowledgeStore Core Data Model ontology 0.2

Namespace Document 09 Jan 2014

Document URL:
http://dkm.fbk.eu/ontologies/knowledgestore.html (HTML, PDF)
Vocabulary URL:
http://dkm.fbk.eu/ontologies/knowledgestore.owl (RDF/XML, TURTLE, Manchester Syntax)
Authors:
Marco Rospocher, Luciano Serafini (FBK-Irst)
Francesco Corcoglioniti

Abstract

This specification describes the KnowledgeStore Core Data Model Ontology, an OWL 2 ontology that formalizes the core concepts of the KnowledgeStore data model. The ontology is centred around the concepts of Resource, Mention and Entity, the latter described by Axioms holding in Contexts. The Core Data Model Ontology is embodied in the KnowledgeStore implementation, and is extended in each KnowledgeStore deploy with a domain-specific ontology that refines the key concepts here formalized (for NewsReader, the specialized ontology is specified here).

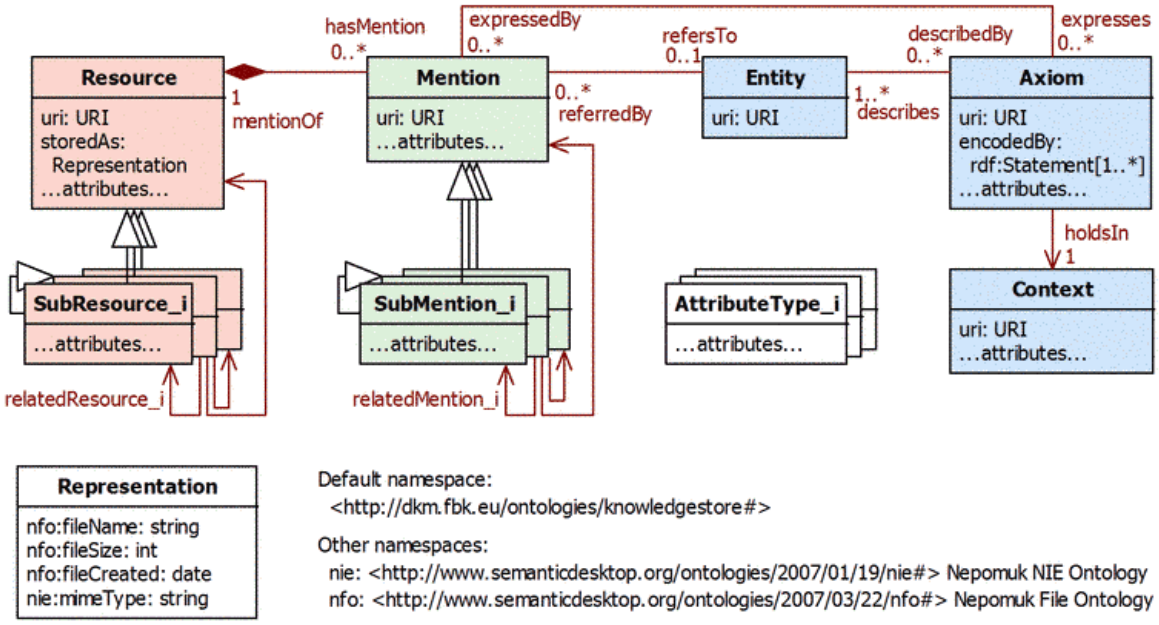
Status of This Document

The KnowledgeStore Core Data Model ontology is a work in progress. This document describes the latest version of the ontology as supported by the KnowledgeStore implementation.

Table of Contents

Overview

The following UML class diagram informally presents an overview of the ontology. Classes are rendered as UML classes, datatype properties as attributes and object properties as UML relations; minimum and maximum cardinalities and expected datatypes are also shown. The components of the ontology are detailed in the following sections.



The vocabulary namespace is `http://dkm.fbk.eu/ontologies/knowledgestore`. The suggested prefix for referencing the vocabulary is `ks:`.

A list of classes and properties is reported below, with links to their reference documentation:

Core concepts

The KnowledgeStore Core Data Model ontology is organized in the three *resource*, *mention* and *entity* representation layers.

The *resource layer* (red part in the UML diagram) stores unstructured content in the form of information objects called *resources*. A Resource is a self-contained piece of unstructured content, such as a news article or a multimedia object, having some descriptive metadata (e.g., from the Dublin Core Terms vocabulary and a digital Representation (attribute storedAs). Representations encode the actual bytes realizing a resource, and are associated to a builtin set of metadata that comprise the file name (attribute nfo:fileName), the size in bytes (attribute nfo:fileSize), the date and time the representation was created (attribute nfo:fileCreated) and the MIME type (attribute nie:mimeType). Information stored in the resource layer is typically noisy, ambiguous, and redundant, with the same piece of information potentially represented in different ways in multiple resources.

The *entity layer* (blue part in the UML diagram) is the home of structured content, and consists in formal *entities*' descriptions made with *contextualized axioms*. An Entity is anything identifiable with a URI, such as persons, organizations, locations and events. Entities are described (property describedBy, inverse describes) by Axioms. An axiom is a logical formula (e.g., that "Barack Obama is president of USA") that is encoded (property encodedBy) with one or more <subject, predicate, object> RDF triples and that possibly holds in (attribute holdsIn) a specific Context (e.g., the time period 2009-2016); axioms may be annotated with additional metadata (e.g., to encode their provenance), while contexts are specified by a number of properties called *contextual dimensions* (they are defined in extensions of this ontology). Both axioms and contexts are identified with URIs automatically assigned by the system based on the RDF statements and context of the former and the contextual attributes of the latter. The entity layer builds on Knowledge Representation and Semantic Web best practices and, differently from the resource layer, it aims at providing a formal and concise representation of the world, abstracting from the many ways it can be encoded in natural language or in multimedia, and thus allowing the use of automated reasoning to derive new statements from asserted ones.

Between the aforementioned two layers is the *mention layer* (green part in the UML diagram), consisting of *mention* descriptions. A Mention is a snippet of a resource (property mentionOf, inverse hasMention), such as some characters in a text document or some pixels in an image, that may refer to some entity of interest (property refersTo, inverse referredBy) and/or can express a number of axioms (property expresses, inverse expressedBy). Mentions can be automatically extracted by natural language and multimedia processing tools, that can enrich them with additional attributes about how they denote their referent (e.g., name, qualifiers, 'sentiment'). Therefore, mentions present both unstructured and structured facets not available in resources and entities layers alone, and are thus a valuable source of information on their own.

Extending the model

Being embodied in the implementation, the KnowledgeStore Core Data Model ontology is kept as small as possible in order not to sacrifice flexibility. Therefore, relevant information such as resource types and metadata, contextual

dimensions, mention types and linguistic attributes are not defined in this ontology, due to the fact that a stable, exhaustive and cross-domain characterization of them cannot be drawn. Instead, this information is supplied in an *extension ontology* (such as the NewsReader one) that is specific to a KnowledgeStore deployment and extend the ontology here specified.

More in details, an extension ontology can supply the following information:

- The subclass hierarchy of Resource and Mention (entities excluded as described via axioms); subclasses are not assumed to be disjoint.
- The additional attributes of Resource, Mention, Axiom, Context and their subclasses. Context attributes define the contextual dimensions for a particular scenario and are used by the system to generate the context URI. In case of objects belonging to multiple subclasses, their description can make use of all their combined attributes.
- Additional relations among resources or among mentions (but not between the two).
- Enumerations and classes used as attribute types (similarly to Representation).
- Restrictions on the domain and range of fixed-part relations.

Data representation using Named Graphs

While axioms are just bunches of triples that can be encoded with plain RDF, axiom metadata and contextual information are more complex to represent in RDF; still, their RDF representation is a requirement for enabling import and export of RDF entity data and thus making the KnowledgeStore compatible with existing RDF datasets. This issue is addressed using named graphs, an extension of RDF supported by the majority of tools and by several RDF syntaxes, and following and extending the CKR approach. Using named graphs, an axiom together with its context and metadata can be represented as shown below.

```
@prefix ckr: <http://dkm.fbk.eu/ckr/meta#> .

<module_uri> { ... axiom triples ... }

ckr:global {

  <module_uri>  <metadata_property_1> <metadata_value_1> ; ... ;
                <metadata_property_N> <metadata_value_N> .

  <context_uri> a ckr:Context ;
                ckr:hasModule <module_uri> ;
                <contextual_dimension_1> <contextual_value_1> ; ... ;
                <contextual_dimension_M> <contextual_value_M> ;
}
```

Triples encoding the axiom are stored in a graph called *module*, which in turn is associated to the axiom metadata inside a special ckr:global graph; contextual information is also encoded in ckr:global, and attached to the axiom module via a ckr:hasModule triple. While seemingly verbose, this representation allows putting multiple axioms in the same module in case they share the same context and metadata (this is often the case for axioms coming from the same source), thus limiting the number of triples in ckr:global and making the associated overhead negligible. A concrete example of this representation is reported below.

```
# ckr, ks, nwr, sem, dbo, ex, dbpedia, dbo, xsd prefix definitions omitted

ex:mod01 { dbpedia:Barack_Obama dbo:birthPlace dbpedia:Honolulu }  # the axiom

ckr:global {

  ex:mod01 nwr:crystallized "true"^^xsd:boolean ;
           nwr:confidence 1.0 ;
           nwr:source dbpedia:DBPedia ;                  # comes from DBPedia
           ks:expressedBy ex:mention15 , ex:mention127 . # but also extracted from
                                                         # two mentions
  ex:ctx01 a ckr:Context ;
           ckr:hasModule ex:mod01 ;
           sem:hasTimeValidity ex:any-time ;  # open interval, definition omitted
           sem:hasPointOfView ex:common-pov . # express common POV without particular
                                              # authority, definition omitted
}
```

Terms reference

Classes: | Axiom | Combination | Context | Entity | Mention | Representation | Resource |

Properties: | describedBy | describes | encodedBy | expressedBy | expresses | hasMention | holdsIn | matchedAxiom | matchedEntity | matchedMention | matchedResource | mentionOf | referredBy | refersTo |

Classes and Properties (full detail)

Classes

Class: ks:Axiom

An ontological axiom describing or relating one or more entities (being them particulars or universals) and holding in a specific context, possibly extracted from some mentions and annotated with some metadata.

Properties include: ks:holdsIn ks:encodedBy
Used with: ks:matchedAxiom ks:expresses ks:describedBy
Restriction(s): The property ks:describes must *have some* ks:Entity value(s)
The property ks:holdsIn must *have some* ks:Context value(s)
The property ks:encodedBy must *have some* rdf:Statement value(s)

Class: ks:Combination

A combination returned by the match() operation, including a mention, its containing resource, the entity it refers to and the axioms expressed by the mention.

Properties include: ks:matchedMention ks:matchedResource ks:matchedEntity ks:matchedAxiom

Class: ks:Context

A region in a space of contextual dimensions (e.g., time, point of view) where certain statements hold, identified by a URI.

Used with: ks:holdsIn

Class: ks:Entity

Any identifiable entity in the domain of discourse, extracted from text and/or imported from some source of background knowledge.

Properties include: ks:describedBy
Used with: ks:refersTo ks:matchedEntity

Class: ks:Mention

A fragment of resource (e.g., a piece of text or a bunch of pixels) that denotes something of interest, such as an entity or a relation among entity, a concept.

Properties include: ks:expresses ks:refersTo
Used with: ks:matchedMention ks:hasMention
Restriction(s): The property ks:mentionOf must http://dkm.fbk.eu/ontologies/knowledgestore#Resource value(s) and 1 time(s)

Class: ks:Representation

A digital representation of a resource. Instances of this class are automatically managed by the KnowledgeStore based on uploaded resource representations

Class: ks:Resource

A self-contained and identifiable information object with a digital representation, such as a news article, a photo or a video.

Properties include: ks:hasMention
Used with: ks:matchedResource

Properties

Property: ks:describedBy

Denotes the axioms describing a certain entity.

Domain: ks:Entity
Range: ks:Axiom
Has inverse property ks:describes

Property: ks:describes

Denotes the entity/ies this axiom describes / put in some relation

Inverse property of ks:describedBy

Property: ks:encodedBy

Denotes the RDF statements an axiom is encoded by.

Domain: ks:Axiom
Range: rdf:Statement

Property: ks:expressedBy

Denotes a mention an axiom is expressed by (i.e., the meaning of the mention is formally captured by the axiom).

Inverse property of ks:expresses

Property: ks:expresses

Denotes the axiom(s) that formally express the meaning of the mention. Those axioms may be the result of some NLP task that extracts structured information out of the unstructured text of mentions.

Domain: ks:Mention
Range: ks:Axiom
Has inverse property ks:expressedBy

Property: ks:hasMention

Denotes the mention(s) contained in a resource.

Domain: ks:Resource
Range: ks:Mention
Has inverse property ks:mentionOf

Property: ks:holdsIn

Denotes the context an axiom holds in.

Domain: ks:Axiom
Range: ks:Context

Property: ks:matchedAxiom

Links a combination instance to its matched axioms.

Domain: ks:Combination
Range: ks:Axiom

Property: ks:matchedEntity

Links a combination instance to its matched entity.

Domain: ks:Combination
Range: ks:Entity

Property: ks:matchedMention

Links a combination instance to its matched mention.

Domain: ks:Combination

Range: ks:Mention

Property: ks:matchedResource

Links a combination instance to its matched resource.

Domain: ks:Combination
Range: ks:Resource

Property: ks:mentionOf

Denotes the resource a mention is part of.

Inverse property of ks:hasMention

Property: ks:referredBy

Denotes the mention(s) that refer to this entity.

Inverse property of ks:refersTo

Property: ks:refersTo

Denotes the entity a given mention refers to.

Domain: ks:Mention
Range: ks:Entity
Has inverse property ks:referredBy