The Universal Data Cube Core Library

Curran Kelleher

Abstract

The UDC Core Library provides a concrete software specification of the UDC Theoretical Model.

1 Introduction

The UDC Core Library provides two distinct components: the metadata framework, a specification enabling the UDC metadata system to be realized in memory; and the data cube API, a specification for data cube implementations. The aim of the UDC Core Library is to provide a minimalistic yet comprehensive API specification (*not* an implementation) which provides a common language for all components of the UDC System.

2 The Metadata Framework

The UDC Core Library specifies a system in which knowledge in the Universal Data Cube metadata system can be partially reconstructed in memory. Interfaces were created for each UDC Ontology class. The purpose of this is to enable semantic web Resources published using the UDC Ontology to be dynamically loaded into memory as class instances.

A key contract which must be adhered to is that each resource be represented by exactly one in-memory class instance. Implementations of the UDC Core API should enforce this contract by using the factory pattern, always consulting a global index mapping URIs to class instances before creating new ones. This contract affords efficient implementation of data cubes whether they reside in a database or in memory, and convenient usage of their resulting class instances by clients of UDC Core.

Uniqueness of resource instances is preserved by maintaining an index mapping global URIs to local class instances, and enforcing (through use of the Factory design pattern) its use whenever new resources are consumed.

In addition to semantic web resources being mapped to class instances, the properties of these semantic web resources are mapped to class member variables. This enables very efficient implementations of metadata queries relying on graph traversal such as "What are all the counties of Texas?" or "What are all the months from 1990 to 1995?". Traversing the JVM object graph directly in order to answer these queries is likely much more efficient than relying on a SPARQL implementation. In addition, providing classes, properties, and operations at the programming language level makes code much cleaner than it would be if one had to instead always interact with an RDF model via SPARQL and other tools.

3 The Data Cube Interface

In addition to the classes and properties found in the UDC Ontology, interfaces for defining data cube implementations are also part of the UDC Core Library.