Dublin Core Metadata Initiative Abstract Model
Jenn Riley
Digital Library Brown Bag Series
February 11, 2009

Resources

Presentation slides: <http://www.dlib.indiana.edu/~jenlrile/presentations/bbspr09/dcam/dcmi-am.ppt>

Handout: <http://www.dlib.indiana.edu/~jenlrile/presentations/bbspr09/dcam/handout.pdf>

DCMI Abstract Model home page: <http://dublincore.org/documents/abstract-model/>


DCMI/RDA Task Group home page: <http://dublincore.org/dcmirdataskgroup/>

Further Reading


Understanding the Diagrams Illustrating the Abstract Model

The rest of this handout uses information from the DCMI Abstract Model home page, at <http://dublincore.org/documents/abstract-model/>. From that resource:

Lines ending in a block-arrow should be read as 'is' or 'is a' (for example, "a value is a resource") and that lines starting with a block-diamond should be read as 'contains a' or 'has a' (for example, "a statement contains a property URI").
The abstract model of the resources described by descriptions is as follows:

- Each described resource is described using one or more property-value pairs.
- Each property-value pair is made up of one property and one value.
- Each value is a resource - the physical, digital or conceptual entity or literal that is associated with a property when a property-value pair is used to describe a resource. Therefore, each value is either a literal value or a non-literal value:
  - A literal value is a value which is a literal.
  - A non-literal value is a value which is a physical, digital or conceptual entity.
- A literal is an entity which uses a Unicode string as a lexical form, together with an optional language tag or datatype, to denote a resource (i.e. "literal" as defined by RDF).
DCMI Description Set Model
From <http://dublincore.org/documents/abstract-model/>

The abstract model of DC metadata *description sets* is as follows:

- A *description set* is a set of one or more *descriptions*, each of which describes a single *resource*.
- A *description* is made up of one or more *statements* (about one, and only one, *resource*) and zero or one *described resource URI* (a *URI* that identifies the *described resource*).
- Each *statement* instantiates a *property-value pair*, and is made up of a *property URI* (a *URI* that identifies a *property*) and a *value surrogate*.
- A *value surrogate* is either a *literal value surrogate* or a *non-literal value surrogate*:
  - A *literal value surrogate* is a *value surrogate* for a *literal value*, and is made up of exactly one *value string*. The *value string* is a *literal* which encodes the *literal value*.
  - A *non-literal value surrogate* is a *value surrogate* for a *non-literal value*, and is made up of zero or one *value URI* (a *URI* that identifies the *non-literal value* associated with the *property*), zero or one *vocabulary encoding scheme URI* (a *URI* that identifies the *vocabulary encoding scheme* of which the *non-literal value* is a member), and zero or more *value strings*. Each *value string* is a *literal* which represents the *non-literal value*.
- A *value string* is either a *plain value string* or a *typed value string*:
  - A *plain value string* may have an associated *value string language* that is an ISO language tag (for example en-GB). *Plain value strings* are intended to be human-readable.
  - A *typed value string* has an associated *syntax encoding scheme URI* that identifies a *syntax encoding scheme*. 
Singapore Framework
From <http://dublincore.org/documents/singapore-framework/>