Review of the DCMI Abstract Model

Thomas Baker, DCMI
Joint Meeting of the DCMI Architecture Forum
and W3C Library Linked Data Incubator Group
22 October 2010

DRAFT SLIDES 2010-10-06
Early 2000s

DC straddling two mindsets

• **New “RDF graph” mindset**, reflected in:
  – Dublin Core as an RDF vocabulary declared in a formal RDF schema
  – Two patterns for DC in RDF/XML

• **Traditional IT “record format” mindset**
  – Element Set as slots in records
  – Plethora of random concrete syntaxes
    • HTML and XML very popular (e.g., OAI_DC format)
    • Most implementations made no reference to abstract syntax

• Interoperability among “Dublin Core” implementations problematic!
  – Problem of mapping/profiling DC with IEEE LOM
    • Not just about mapping elements
    • Also about underlying model…

• But RDF a hard sell
  – Researchy, “pie-in-the-sky”
  – Perception: “Just another flavor of XML…”
    • a (needlessly) complicated XML format…
Role of DCAM

• 2003-2007 motivation: to build a bridge...
  – Transition from mindset of trees and record formats to mindset of linked graphs
  – Harmonization of “native DC” model with RDF
• Today: quality and consistency of Linked Data?
  – Notion of **descriptive patterns** reflecting existing metadata practice
  – Notion of **bounded records** in support of provenance
  – Notion of **constraints** on those graphs supporting communities of practice
Resource URI

Property URI

Value URI

Property URI

dcam:memberOf

Value URI

"literal" @lang

Property URI

rdf:value

"literal" ^^SES URI

Resource URI

Property URI

"literal" @lang

Property URI

"literal" ^^SES URI
DC-RDF

Expressing Dublin Core metadata using the Resource Description Framework (RDF)

http://dublincore.org/documents/dc-rdf/
A common abstract syntax for diverse concrete syntaxes
A common abstract syntax for diverse concrete syntaxes
To provide a common interface for interoperating across syntaxes
DCAM “family” of specifications

**Model specifications**
- DCMI Abstract Model
- Description Set Profiles: A constraint language for Dublin Core Application Profiles

**Syntax expressions**
- Guidelines for implementing Dublin Core in XML (DC-XML) – last updated 2003 - not based on DCAM
- Expressing Dublin Core Description Sets using XML (DC-DS-XML)
- Expressing Dublin Core metadata using HTML/XHTML meta and link elements (DC-HTML)
- Expressing Dublin Core metadata using the Resource Description Framework (RDF)
- Expressing Dublin Core metadata using the DC-Text format

**User Guidance**
- Interoperability Levels for Dublin Core Metadata
  - [http://dublincore.org/documents/interoperability-levels/](http://dublincore.org/documents/interoperability-levels/)
- DCMI Usage Board Criteria for the Review of Application Profiles
- Singapore Framework for Dublin Core Application Profiles
- Guidelines for Dublin Core Application Profiles
Description Set Profile Constraint Language

Structural constraints on DCAM entities when used in records
A book:
  a literal title
  a creator, described separately
A creator
  a literal name

Source: Mikael Nilsson
Singapore Framework for
dublin core application profiles

Application Profile uses

Functional Requirements

Domain Model

Built on

Description Set Profile

Built on

Domain Standards

Community Domain Models

Metadata Vocabularies

Built on

DCMI Abstract Model

Built on

Foundation Standards

RDFS (RDF Vocabulary Description Language)

Built on

RDF

Built on

Data Format

Built on

DCMI Syntax Guidelines

Built on

Based on: http://dublincore.org/documents/singapore-framework/
Interoperability Levels for Dublin Core metadata

1: Informal interoperability
- Shared vocabularies defined in natural language

2: Semantic interoperability
- Shared formal-semantics model

3: Description Set syntactic interoperability
- Shared formal model for "records"

4: Description Set Profile interoperability
- Shared formal vocabularies and constraints in records

http://dublincore.org/documents/interoperability-levels/
Deployed base

Shared (natural-language) definitions

Shared formal-semantic model

Shared "records"

Shared constraints
Future DCAM Scenarios

1. **DCMI carries on developing DCAM as before** – incrementally improving the DCAM and Description Set Profile specifications, with a work plan for further concrete syntaxes based on DCAM.

2. **DCMI develops a “DCAM 2” specification** – simplified and better aligned with RDF
   - **Variant 2a.** The improved DCAM 2 specification is taken as the new basis for structural constraints (DSP) and for continued work on concrete syntaxes
   - **Variant 2b.** The improved DCAM 2 specification is treated as a clarification, for the benefit of the Dublin Core community, of how the DCAM abstract syntax relates to the RDF abstract syntax, to be deprecated over time in favor of the latter. No new concrete syntaxes are undertaken.

3. **DCMI deprecates DCAM** – a “product of its time” – and henceforth works on the basis of the RDF abstract syntax.

4. **DCMI does nothing** – either to develop DCAM and DSP further or to change or clarify their existing statuses. In effect, DCAM and DSP are “frozen” and de-emphasized, but with no particular explanation.
Future DCAM Scenarios

1. DCMI carries on developing DCAM

• Incremental improvements to DCAM
  – Structural constraints for application profiles, as before, on basis of DSP
  – Work plan for further concrete syntaxes based on DCAM (such as DC-DS-XML)

• Questions:
  – Is there a demonstrated interest?
  – Who would edit the specs?
  – How would review and testing be managed?
Future DCAM Scenarios

2a. “DCAM 2” as basis for new work

- DCMI would develop a “DCAM 2” specification – simplified and better aligned with RDF
- In Variant 2a, the improved DCAM 2 specification would be taken as the new basis:
  - for structural constraints of application profiles (DSP)
  - for a workplan to develop new and existing concrete syntaxes
- Questions:
  - Is there a demonstrated interest in “DCAM 2”?
  - Who would edit the specs?
  - How would review and testing be managed?
  - What would be the impact of “DCAM 2” on specifications in the existing “DCAM family”?
Future DCAM Scenarios

2b. “DCAM 2” to clarify, then deprecate

- DCMI would develop “DCAM 2” specification
  - Simplified and better aligned with RDF
- In Variant 2b, different goal for “DCAM 2”:
  - Clarification, for DC community, of how DCAM relates to RDF and Linked Data
  - Transitional specification, to be deprecated over time in favor of RDF
  - No new concrete syntaxes to be undertaken
- Questions
  - Is there clear interest in “DCAM 2” (for purposes of clarification and transition)?
  - Who would edit “DCAM 2”?
  - What should be done with the existing “DCAM family” of specifications?
Future DCAM Scenarios

3. Deprecate DCAM, embrace RDF

- DCMI deprecates DCAM
  - A “product of its time”
  - Henceforth promote RDF abstract syntax

- Questions
  - Are there users of DCAM that would be negatively impacted?
  - What should be done with the existing “DCAM family” of specifications?

- Status of each document
- Change of DCMI message

- What is an application profile, if not based on DCAM (Singapore Framework, DSP...)?
Future DCAM Scenarios

4. DCAM simply left untouched

• DCMI does nothing
  – No changes to DCAM or DSP or clarification of their statuses.
  – In effect, DCAM and DSP are “frozen” and de-emphasized, but with no particular explanation.

• Questions
  – Does DCMI really stand behind continued “recommendations”?
  – What cost in reputation and credibility?
Issue: DCAM abstract syntax

• DCAM abstract syntax vs RDF abstract syntax
  – Should DCAM dissolve into mainstream RDF?
  – Are Descriptions and Description Sets expressible as Named Graphs?
  – Significant differences between Vocabulary Encoding Schemes and SKOS Concept Schemes?

• DCAM-related modeling guidance
  – Use of rdf:value (or skos:prefLabel, rdfs:label, foaf:name, skos:notation, dcterms:title...)?
Issue: Application Profiles

• Does RDF need a notion of Application Profiles?
  – What are the requirements?
  – Do profiles need to express constraints?
• Representing patterns of constraints at level of RDF graph
  – Syntax pattern checks
    • Something like Description Set Profile constraint language?
    • Like DSP approach, checking patterns “in the graph” rather than “in the world”?
  – OWL applied with closed-world Assumptions?
• Singapore Framework split:
  – underlying vocabularies / data expressed using vocabularies
• At what level to express those constraints:
  – Wired into specification of underlying vocabulary?
  – Expressed as patterns matched to the data?