```
<PRE>
Title:                  Usage Board Meeting Agenda
Date of meeting:        27-28 September 2003
Venue:                  Seattle
Date Modified:          2003-09-17
Maintainer:             Thomas Baker, Usage Board chair <thomas.baker@bi.
fhg.de>
Description:            Revisions are posted to the Usage Board mailing
list.
                        On completion of a meeting, the archived agenda
can be
                        accessed through the Usage Board Meetings page.

UB member participants:

   Andy Powell            Rebecca Guenther        Traugott Koch
   Diane Hillmann         Roland Schwaenzl        Andrew Wilson
   Stuart Sutton

   Apologies: Haruki Nagata

Guests:

   Stu Weibel (Sunday only)
   Heike Neuroth (Sunday only)
   Jani Stenvall (Sunday only?)


TOPIC 01. Unfinished business from Ithaca (Tom) - Saturday

   Assuming I can participate by phone, we should talk through
   the Decisions from the Ithaca meeting and assess what items
   remain to be done (lots unfortunately, many of which are
   on my plate).

   Reading:
   -- Ithaca agenda
      http://www.gmd.de/People/Thomas.Baker/Agenda-ithaca.html
   -- Ithaca decisions
      http://www.gmd.de/People/Thomas.Baker/Decisions-ithaca.html

TOPIC 02. Abstract Model (Andy) - Saturday
```

```
   Andy will summarize the recent discussion on DC-Architecture
   on the Abstract Model.  We should consider folding this
   article together with the DCMI Grammatical Principles,
   with which it overlaps.  Tom has suggested incorporating
   some aspects of the D-Lib Magazine article "Grammar of
   Dublin Core" (metadata as "statements" of a particular
   structure; the notion of a "core vocabulary" or "pidgin"),
   so I will include this in the packet as well.

   Required reading:
   -- Andy's Abstract Model
      http://www.ukoln.ac.uk/metadata/dcmi/abstract-model/
   -- DCMI Grammatical Principles
      http://stage.dublincore.org/usage/documents/principles/
   -- D-Lib article "Grammar of Dublin Core"
      http://www.dlib.org/dlib/october00/baker/10baker.html

TOPIC 03. DCMICite and other DCSV proposals (Andy) - Sunday

    In light of the discussion on the Abstract Model, we
    need to decide whether to approve the DCMICite proposal.
    Whichever way we decide, we will need to justify our
    decision and consider what it implies for other such
    proposals in the pipeline (e.g., DCMI Agent Detail);
    for existing approved DCSVs (Point, Box, Period); and
    for the related syntax documents.

    Required reading:
    -- DCMICite: a Bibliographic Citation Dublin Core Structured
       Value (DCSV) Encoding Scheme
       http://epub.mimas.ac.uk/DC/citdcsv.html
    -- Background reading on DCMICite (required)
       http://epub.mimas.ac.uk/DC/citdcsvbackgd.html

   Background reading:
   -- DCMI Agent Detail, structured values for CCP elements (Rebecca
Guenther)
      http://www.jiscmail.ac.uk/cgi-bin/wa.exe?A2=ind0204&L=dc-
agents&F=&S=&P=169
   -- http://dublincore.org/documents/2000/07/28/dcmi-dcsv/
   -- http://dublincore.org/documents/1999/04/30/labelled-values-
```

```
syntax/
   -- http://dublincore.org/documents/2000/07/28/dcmi-point/
   -- http://dublincore.org/documents/2000/07/28/dcmi-period/
   -- http://dublincore.org/documents/2000/07/28/dcmi-box/

TOPIC 04. CEN "Guidelines for Dublin Core Application Profiles" (Tom)
- Saturday

    Assuming he is participating by phone, Tom will present
    the "Guidelines for Dublin Core Application Profiles"
    recently approved CEN and discuss possible sharing
    between CEN and DCMI of maintenance responsibilities
    for this document.  This document, drafted by Tom,
    Makx, Rachel Heery, and Thomas Fischer is based on an
    analysis of over twenty existing application profiles
    of Dublin Core. The guidelines aim at providing a
    sensible and coherent format that will meet the needs
    of most Application Profile creators; that is as simple
    as possible yet as detailed as is sometimes necessary;
    and that will lend itself in the future to transformation
    into machine-processable forms.

    Tom would like the Usage Board to "approve" these
    guidelines as something we can link from the UB Documents
    page and to consider sharing maintenance responsibility
    for the document with the CEN MMI-DC Workshop.  If we do
    this, we should clarify the dependency of this document
    on the Abstract Model and Principles.

    Required reading:
    -- CEN "Guidelines for Dublin Core Application Profiles"
       http://www.gmd.de/People/Thomas.Baker/mmidc076.pdf
    -- DCMI Usage Board Review of Application Profiles
       http://dublincore.org/usage/documents/profiles/

TOPIC 05. "Using Dublin Core" (Diane) - Saturday

    Diane will bring us up-to-date.

    Required reading:
    -- http://dublincore.org/documents/2003/08/26/usageguide/
    -- http://dublincore.org/documents/2003/08/26/usageguide/glossary.
```

shtml

```
TOPIC 06. Registration of Encoding Schemes (Traugott and Stu, as
guest)
          - Sunday

   In Seattle, Traugott should briefly review the open issues,
   from his perspective, in light of the discussion in
   Ithaca, and Stu will present a Directorate's perspective
   on the practicalities of offering a DCMI encoding scheme
   registration service.  Stu will also describe a related IETF
   activity of possible relevance to the idea of registering
   new terms.

   Required reading:
   -- Guidelines for Vocabulary and Encoding Scheme Qualifiers,
      http://dublincore.org/usage/documents/vocabulary-guidelines/

TOPIC 07. Libraries profile (Rebecca) - Sunday

   A brief report should identify any further actions that
   may need to be undertaken by Usage Board in support
   of the Library Application Profile.  In particular, the
   ISO8601 and AAT decisions from Ithaca need to be finalized.
   Rebecca can perhaps bring enough copies of the latest text
   of ISO8601 and AAT to pass around.

   In addition, we should perhaps consider, in light of the
   CEN Guidelines, how we might go about formally reviewing
   the Libraries profile -- possibly even at the next meeting.
   I have not included a copy of the Libraries profile in the
   packet because of its length; the text was included in the
   supplementary packet for Ithaca.

TOPIC 08. MARC Relator Terms as Refinements for Contributor (Rebecca)
          - Sunday

   In Florence, we agreed to work with the Library of Congress
   to have MARC relator terms declared as refinements of
   Contributor.  LC is currently implementing this decision
   and Rebecca can report in Seattle.  We should determine
   whether the Usage Board needs to follow through, and how,
   and identify any lessons learned.  After Seattle, we will
```

need to prepare a document explaining how MARC Relator terms can be used as refinements for Contributor.

The readings listed below may be a bit out of date with respect to Rebecca's latest work.

Required reading:
-- Assignment of URIs to metadata terms of MARC 21 - Action plan
   http://dublincore.org/usage/meetings/2003/06/MARC-URIs.pdf
-- Excerpts from the RDF schema declaring the MARC Relator terms
   http://dublincore.org/usage/meetings/2003/06/Relator-excerpts-rdfxml.html

TOPIC 09. AskDCMI (Stuart and Diane) - Saturday

Stuart and Diane will report on AskDCMI, which has been in operation for the past few weeks. We will discuss the role of Usage Board members in handling questions.

Required viewing:
-- http://askdcmi.askvrd.org/

Required reading:
-- http://dublincore.org/usage/meetings/2003/06/AskDCMI.html
-- http://askdcmi.askvrd.org/services/askdcmi/expert_tips.asp

TOPIC 10. Proposal from DCMI Type Working Group (Diane)

In Ithaca, we approved two related proposals to add terms to the DCMI Type Vocabulary -- "Moving Image" and "Still Image". Finalizing these decisions depends on adding to the Abstract Model and/or Principles a notion of "sub-class".

Diane should please bring the latest text for discussion.

TOPIC 11. Creative Commons proposal (Andrew and Stu, as guest)
          -- Sunday

Stu has been working with Eric Miller and with Andy on a proposal for handling references to Creative Commons licenses with dc:rights. Prior to the meeting he will circulate a text. The UB will not be in a position to vote on this, as it will not have been out for comment for the required four weeks, but we should be able to provide some useful feedback, especially in light of the Abstract Model discussion.

Andrew, who will shepherd this proposal at the next meeting, will report on Creative Commons's own attempts to encode basic rights information, as well as efforts underway in the METS context.

Stu to circulate a document before the meeting.

TOPIC 12. Usage Board recruitment (Stu)

Stu will say a few words.

```
Title:                  Usage Board Meeting Agenda
Date of meeting:        16-17 June 2003
Venue:                  Cornell University, Ithaca, New York
Date Modified:          2003-06-11
Maintainer:             Thomas Baker, Usage Board chair
Description:            Revisions are posted to the Usage Board mailing list.
                        On completion of a meeting, the archived agenda can
be
                        accessed through the Usage Board Meetings page.

##########################################################################
Note: This version of 2003-06-11 is "frozen".  All non-DCMI materials
(with some exceptions, e.g., JISCMAIL) have been archived on the DCMI
Web site, currently on http://stage.dublincore.org.  After the next build,
these archived documents will become available on the public Web site,
at which time this document will be edited to show http://dublincore.org
addresses and archived.  PDF packets have been prepared which hold _all_
of
the materials cited with URLs in this agenda.  This PDF packet, along with
a packet of "background materials" is available at:
    http://www.gmd.de/People/Thomas.Baker/Agenda-ithaca.pdf
    http://www.gmd.de/People/Thomas.Baker/Agenda-background.pdf
##########################################################################

Participants:
   Tom Baker, chair     Rebecca Guenther        Traugott Koch
   Andy Powell          Roland Schwaenzl        Andrew Wilson
   Diane Hillmann       Stuart Sutton           Makx Dekkers, guest

Apologies: Haruki Nagata

Venue:     http://stage.dublincore.org/usage/meetings/2003/06/Venue.html
Schedule:  http://stage.dublincore.org/usage/meetings/2003/06/Schedule.
html

TOPIC 01. Mission and principles (Tom)

   Some issues related to dumb-down need clarification --
   for example, does dumb-down mean "resolve all sub-property
   relationships" or "resolve to Elements (as opposed to
   Element Refinements)"?  Other issues relate to Encoding
   Schemes (how valid is the distinction between Vocabulary
   and Syntax encoding schemes?). We cannot resolve these
   issues without first discussing a draft on the list, so
   in Ithaca we should aim merely to identify and prioritize
   the open issues.

   The Mission document has been updated to remove redundant
   text about grammatical principles (which has since been
   moved to the the DCMI Grammatical Principles).

   Required reading:
   -- http://stage.dublincore.org/usage/meetings/2003/06/TOPIC01.html
   -- DCMI Grammatical Principles
      http://stage.dublincore.org/usage/documents/principles/
   -- Mission of the DCMI Usage Board
      http://stage.dublincore.org/usage/documents/2003/06/11/mission/
   -- Email digest on "vocabulary" vs "syntax" encoding schemes
      http://stage.dublincore.org/usage/meetings/2003/06/TOPIC01.types-of-
encoding-schemes.html

   Time needed: 30"

TOPIC 02. Usage Board Process (Stuart)

   A brief report on changes or clarifications made since
   the Florence meeting.

   Copies of the Process documents have been put into the
   "background readings" packet.

   Required reading:
   -- http://stage.dublincore.org/usage/meetings/2003/06/TOPIC02.html

   Time needed: 20"

TOPIC 03. DCMI Documentation (Tom)

   Brief report on the work-flow for generating Web pages
   and RDF schemas from the raw term declaration data in XML.

   There are small differences of wording between the DCMI
   Terms document and the newly published ISO standard.
   In Ithaca, we will vote on changing the DCMI wordings to
   bring them into line with the ISO standard.  Once this
   is done, we will need to inform John Kunze so that he can
   revise the RFC.

   Required reading:
```

```
   -- http://stage.dublincore.org/usage/meetings/2003/06/TOPIC03.html
   -- Differences between Usage Board documentation and ISO/NISO
      http://stage.dublincore.org/usage/meetings/2003/06/UB-ISO.
differences.html

   Time needed: 30"

TOPIC 04. "Using Dublin Core" (Diane)

    Diane has revised "Using Dublin Core" (e.g., with specific
    guidance for individual element refinements).  Since DCMI
    has no real process for approving the revision of this
    document (or at least none that really works), we should
    figure out how to bring the process of revision to a close.
    Time permitting, we could perhaps return to this document
    after other business is taken care of in order to discuss
    issues of content.

    The entire draft of "Using Dublin Core" is available in
    the packet of "background readings".

    Time needed: 30"

    Required reading:
    -- http://stage.dublincore.org/usage/meetings/2003/06/TOPIC04.html

TOPIC 05. Attributes of DCMI Terms (Tom)

   Put another way: What is the Application Profile used by
   the Usage Board to declare terms?  Do we need to clarify the
   details of this profile and declare it formally?

   Required reading:
   -- http://stage.dublincore.org/usage/meetings/2003/06/TOPIC05.html
   -- Usage Board Application Profile (draft)
      http://stage.dublincore.org/usage/meetings/2003/06/Usage-ap.html
   -- Attributes of DCMI metadata terms
      http://stage.dublincore.org/usage/meetings/2003/06/Rdf-xml-data-
attributes.pdf

   Time needed: 30"

TOPIC 06. RDF schemas of DCMI terms (Roland/Makx/Tom)

   Tom, Roland, Eric, Dave Beckett, Makx, and Stu held a
   round of discussions earlier this year with the goal of
   resolving modeling issues for declaring DCMI terms in
   RDF schemas.  Makx, Tom, and Roland will report on the
   status of these discussions and on future paths forward.
   We will discuss the role of the Usage Board as compared
   with DC-Architecture and other groups within DCMI with
   regard to the schemas.

   Time needed: 45"

   Required reading:
   -- http://stage.dublincore.org/usage/meetings/2003/06/TOPIC06.html

TOPIC 07. Registration of Encoding Schemes (Traugott)

   In Ithaca, we should evaluate the test period for Encoding
   Scheme registration and and clarify any problems that may
   have arisen.

   Required reading:
   -- http://stage.dublincore.org/usage/meetings/2003/06/TOPIC07.html
   -- Guidelines for Vocabulary and Encoding Scheme Qualifiers,
      http://dublincore.org/usage/documents/vocabulary-guidelines/

   Prototype:
   -- Vocabulary and Encoding Scheme Registration,
      http://wip.dublincore.org:8080/schemes/index.html
   -- Proposals pending as of 2003-06-11
      http://wip.dublincore.org/schemes/searchServlet?
reqType=summary&status=pen.

   Time needed: 90"

TOPIC 08. Structured Values, or "DCSVs" (Andy)

   Andy has developed a "Usage Board view of Structured Values"
   that we can discuss among ourselves in Ithaca and then
   with DC-AB and DC-Architecture.  Once UB principles are
   agreed we can go back to Citation WG and Libraries WG with
   advice about submitting proposals for structured values.

   Required reading:
   -- http://stage.dublincore.org/usage/meetings/2003/06/TOPIC08.html
   -- Andy's discussion paper (2003-06-10)
      http://stage.dublincore.org/usage/meetings/2003/06/Structured-
```

```
values.html

   Time needed: 90"

TOPIC 09. Libraries profile (Rebecca)

   A brief report should identify any further actions that
   may need to be undertaken by Usage Board in support of
   the Library Application Profile.  We should try to approve
   the nine proposed encoding schemes on a fast-track basis.
   With reference to the Usage Board review guidelines, we
   should discuss the feasibility of undertaking a formal
   review of the profile at the Seattle meeting.

   The entire Libraries Application Profile is included in
   the "background readings" packet.

   Required reading:
   -- http://stage.dublincore.org/usage/meetings/2003/06/TOPIC09.html
   -- Proposals for encoding schemes in DC-Lib Application Profile
      http://stage.dublincore.org/usage/meetings/2003/06/dclib-
encodingschemes.html
   -- DCMI Usage Board Review of Application Profiles
      http://dublincore.org/usage/documents/profiles/

   Time needed: 30"

TOPIC 10. MARC Relator Terms as Refinements for Contributor (Rebecca)

   In Florence, we agreed to work with the Library of Congress
   to have MARC relator terms declared as refinements of
   Contributor.  LC is currently implementing this decision
   and Rebecca can report in Ithaca.  We should determine
   whether the Usage Board needs to follow through, and how,
   and identify any lessons learned.  After Ithaca, we will
   need to prepare a document explaining how MARC Relator
   terms can be used as refinements for Contributor.

   Required reading:
   -- http://stage.dublincore.org/usage/meetings/2003/06/TOPIC10.html
   -- Assignment of URIs to metadata terms of MARC 21 - Action plan
      http://stage.dublincore.org/usage/meetings/2003/06/MARC-URIs.pdf
   -- Excerpts from the RDF schema declaring the MARC Relator terms
      http://stage.dublincore.org/usage/meetings/2003/06/Relator-excerpts-
rdfxml.html
```

```
   -- Comments from Roland
      http://stage.dublincore.org/usage/meetings/2003/06/Marc-relators-in-
rdf.html

   Time needed: 90"

TOPIC 11. AskDCMI (Stuart and Diane)

   Stuart and Diane will report on AskDCMI, which has been
   in operation for the past few weeks.  We will discuss the
   role of Usage Board members in handling questions.

   Required viewing:
   -- http://askdcmi.askvrd.org/

   Required reading:
   -- http://stage.dublincore.org/usage/meetings/2003/06/TOPIC11.html
   -- http://stage.dublincore.org/usage/meetings/2003/06/AskDCMI.html
   -- http://askdcmi.askvrd.org/services/askdcmi/expert_tips.asp

   Time needed: 30"

TOPIC 12. Proposal from DCMI Type Working Group (Diane)

   Two related proposals have been submitted to add terms
   to the DCMI Type Vocabulary -- "Moving Image" and "Still
   Image".  We need to vote on these.

   Required reading:
   -- http://stage.dublincore.org/usage/meetings/2003/06/TOPIC12.html
   -- http://stage.dublincore.org/usage/meetings/2003/06/StillImage.html
   -- http://stage.dublincore.org/usage/meetings/2003/06/MovingImage.html

   Time needed: 90"

TOPIC 13. Strategic role of Usage Board (Makx)

   As Managing Director of DCMI, Makx will lead a discussion
   of the strategic role of the Usage Board for DCMI in
   the medium term.  Some of this discussion can take place
   during breaks and at meals, but we should probably consider
   drafting a planning document over the next few months.

   Questions to consider include:
   - How does the UB relate to strategic planning of DCMI as an
```

```
      organisation?
    - Should the UB contribute more clearly to strategic objectives
      by soliciting proposals in certain areas?
    - Where will proposals come from in the future?

    Time needed: 45"

    Required reading:
    -- http://stage.dublincore.org/usage/meetings/2003/06/TOPIC13.html
    -- http://stage.dublincore.org/usage/meetings/2003/06/UB-Orientation.
pdf

TOPIC 14. Seattle meeting and timing (Tom)

     DC2003 is scheduled for Sunday 28 September through
     Thursday 2 October 2003.  In scheduling a Usage Board
     meeting we need to consider that Rosh-Hashanah is on
     September 27-28 and Yom Kippur on Monday, 7 October.
     These constraints seem to point towards holding a UB
     meeting after the conference, so we will need to discuss
     the exact timing.

     Required reading:
     -- http://stage.dublincore.org/usage/meetings/2003/06/TOPIC14.html

     Time needed: 15"

TOPIC 15. Dublin Core Application Profiles (Tom)

     Tom, Makx, Rachel, and Thomas Fischer have drafted
     Guidelines for Dublin Core Application Profiles.  Based on
     an analysis of over twenty existing application profiles
     of Dublin Core, the draft guidelines aim at providing
     a sensible and coherent format that will meet the needs
     of most Application Profile creators; that is as simple
     as possible yet as detailed as is sometimes necessary;
     and that will lend itself in the future to transformation
     into machine-processable forms.  Should the Usage Board
     approve of, promote, or use these guidelines?  Would the
     Usage Board need to discuss these guidelines in detail?

     Required reading:
     -- Guidelines for Dublin Core Application Profiles
        http://stage.dublincore.org/usage/meetings/2003/06/Guidelines.pdf
     -- DCMI Usage Board Review of Application Profiles
```

```
       http://dublincore.org/usage/documents/profiles/

TOPIC 16. Other issues (Tom)

    Here is a file where I have simply "dumped" issues which
    (at the time of dumping) seemed like they may some day need
    to be addressed.  Please browse this file at your leisure
    and point out if any issues have already been addressed
    or should move up in our list of priorities.

    Required reading:
    -- http://stage.dublincore.org/usage/meetings/2003/06/Issues.html
```

DCMI Usage Board Meeting -- Decisions
2003-06-16/17, Ithaca, New York

Andy Powell	Makx Dekkers (guest)
Diane Hillmann	Traugott Koch
Rebecca Guenther	Andrew Wilson
Roland Schwaenzl	Stuart Sutton
Tom Baker	

TOPIC 1: Mission and Principles

ACTION ITEM Tom: Soften distinction between syntax and vocabulary encoding schemes and post proposed form of words to UB list.

ACTION ITEM Tom: Add some words about resolution of sub-properties for 'dumb-down' purposes.

ACTION ITEM Tom: Suggest some additional text on UB list for Mission statement as decided at Florence UB meeting.

Proposal Roland: Remove from Grammatical Principles the sentence: "In DCMI practice, an Element Refinement refines just one parent property".

Yes: 1;
No: 6;
Abstain: 1.

TOPIC 2. Usage Board Process

ACTION ITEM Makx: Provide URL for new tables for communication
responsibilities to Diane and Stuart.

TOPIC 3. Documentation

ACTION ITEM Tom: Ask Harry to hide raw XML term data.

Decision: Change Usage Board documentation to align it with the wording in the NISO standard.

Decision: Add pointer in DCES document to the ISO standard.

Decision: No further hand editing of version 1.1 document will take place.

ACTION ITEM Makx: To talk with ISO & NISO re posting copies of standards on DCMI website.

Decision: Ask Directorate about explanation and justification for case change.

Decision: In Seattle UB to verify that Directorate explanation for case change has been posted to lists.

ACTION ITEM Makx: Investigate NISO/ISO procedures for making changes to standard and report at Seattle UB meeting.

TOPIC 4. "Using Dublin Core" Document

ACTION ITEM Diane: Add section about difference between simple and qualified DC.

ACTION ITEM Tom: Ask Harry if a script can be written (on basis of registry, or in XSLT) to generate the table of elements and qualifiers in the usage guide.

ACTION ITEM Diane: Send Tom unencumbered version of the elements and qualifiers table.

ACTION ITEM Diane: Add explanation for element categorisation.

ACTION ITEM Stuart: Change process for adding new terms to make the shepherd responsible for ensuring that generic examples are included with each proposal.

ACTION ITEM Andy + Diane + Stuart: Look at issue of how to store examples to allow output in multiple syntaxes.

ACTION ITEM Tom: Write up paragraph about notion of 'Recommended Resource' as a type of DC Document for submission to Directorate.

ACTION ITEM Diane: Draft proposal to fix UB process document to include procedure for updating 'Using Dublin Core' document regularly.

TOPIC 5: Attributes of DCMI terms

----

5 steps
1. Define the set of properties and encoding schemes for describing terms.
2. Understand how they relate to existing terms.
3. Ask DCMI Directorate for UB namespace.
4. Set up UB namespace and declare terms as necessary.
5. Define an application profile.

Timeframe: steps 1-3 for Seattle.

ACTION ITEM Tom: Submit something to the UB list by 4 July 2003.

TOPIC 7: Registration of Encoding Schemes

ACTION ITEM Diane and Stuart: Make necessary updates to the UB Process document.

ACTION ITEM Tom: Ask Directorate to advise UB on their position in regard to the legal issues surrounding encoding scheme registration.

Decision: In the interest of maintaining an audit trail all emails between UB/DCMI and the scheme owner/maintainer to be sent to a closed Jiscmail DC list for permanent retention.

ACTION ITEM Tom + Diane: Draft new document that explains such things as 'good neighbour' policy, what the process involves, the aims of the registration service, registration help, etc.

ACTION ITEM Traugott: List of priorities for

```
enhancements/changes to the scheme registration tool to be
submitted to Makx.

ACTION ITEM Tom: Document the XML output formats that are
wanted from the registration tool.

------------------------------------------------------------------------
TOPIC 8: Structured Values
------------------------------------------------------------------------

ACTION ITEM Andy: Write up result of discussions as abstract
model for DCMI and post to UB list for comment by end
July. Pass to Makx for posting to AB list before end August.

ACTION ITEM Andy + Andrew: Revise DCSV Specification and
various DCSV scheme specifications after approval of abstract
model recommendation.

ACTION ITEM Tom: Integrate Grammar document and abstract
model by end August.

ACTION ITEM Andy: Formulate reply to Citation working group
and Libraries working group reporting on progress with DCSV
model by end July.

------------------------------------------------------------------------
TOPIC 9: Libraries Profile
------------------------------------------------------------------------

Proposals for encoding schemes:

    1. ISO8601
        Yes      7
        Abstain  1
    2. AAT
        Yes      8
        Abstain  0

Issue: Correctness and consistency of names, labels, and
definitions of terms.

ACTION ITEM Tom: Namespace policy to have some provision for
any legal problems that might arise in regard to URIs for
schemes, dating of URIs to be discussed and approved before
```

```
any new namespace is announced by Directorate.

Decision: New terms to go in the terms namespace with one
or two exceptions (MARC, DOI?). Discussions and agreement on
new terms to be continued online in early to mid- July.

------------------------------------------------------------------------
TOPIC 10: MARC Relators as Refinements for Contributor
------------------------------------------------------------------------

ACTION ITEM Rebecca: Add wording to state that MARC:creator
equates to dc:creator.

Decision: Representation of equivalent terms to be dealt with
in description property.

------------------------------------------------------------------------
Topic 11: AskDCMI
------------------------------------------------------------------------

ACTION ITEM Diane: Add all current experts to all topics so
they see all questions.

------------------------------------------------------------------------
TOPIC 12: Proposal from DCMI Type Working Group
------------------------------------------------------------------------

Decision: UB to issue a clarification that, in the interests
of interoperability, proposals to extend the DCMI Type
vocabulary will not be accepted for the foreseeable future.
Instead, implementer communities are strongly encouraged to
develop their own vocabularies to give them the granularity
they need. Diane to incorporate a form of words expressing
this in her report of the decision on the image proposal.

Proposal:   Add still image as a sub-type of Image
    Approve:        6
    Not approve:    1
    Abstain:        1

Proposal:   Add moving image as a sub-type of Image
    Approve:        6
    Not approve:    1
    Abstain:        1
```

```
Proposal:   Deprecate Image
    Approve:        0
    Not approve:    8
    Abstain:        0
```

ACTION ITEM Tom: Add to the Application Profile for DCMI term declarations the notion of 'sub- class of'.

Decision: The new terms will be reflected in the documentation as sub-types. The documentation will include a human-readable statement that the new terms are "more specific than" the term Image. The raw UB data will add that the new image types are sub-classes of Image.

ACTION ITEM Stuart: The UB process document be amended to clarify that change requests from Working Groups should not combine a number of unrelated proposals.

ACTION ITEM Stuart: Add a clarification to the UB process document regarding future development of the DCMI Type vocabulary.

ACTION ITEM Diane + Stuart: Work on the DCMI Type vocabulary to move examples in definitions into a comment field.

TOPIC 13: Strategic Role of UB

Decision: UB is the body responsible for all DCMI namespace issues.

TOPIC 14: Seattle Meeting and Timing

Decision: Tentative dates Saturday 27 and Sunday 28 September.

Title:	**Dublin Core Abstract Model**
Creator:	Andy Powell UKOLN, University of Bath
Date Issued:	2003-08-11
Identifier:	http://dublincore.org/documents/2003/08/10/abstract-model/
Replaces:	
Is Replaced By:	Not applicable
Latest Version:	http://dublincore.org/documents/abstract-model/
Status of Document:	This is a DCMI Working Draft.
Description of Document:	This document describes an abstract model for Dublin Core metadata records.

1. Introduction

This document specifies an abstract model for Dublin Core (DC) metadata records [DCMI]. The primary purpose of this document is to provide a reference model against which particular DC encoding guidelines can be compared, in order to facilitate better mappings and translations between different syntaxes.

2. Terminology

This document uses the following terms:

resource

A *resource* is anything that has identity. Familiar examples include an electronic document, an image, a service (e.g., "today's weather report for Los Angeles"), and a collection of other *resources*. Not all *resources* are network "retrievable"; e.g., human beings, corporations, and bound books in a library can also be considered *resources*.

property
: A *property* is a specific aspect, characteristic, attribute, or relation used to describe *resources*.

element
: An *element* is a *property* of a *resource*.

element refinement
: An *element refinement* is a *property* of a *resource* that shares the meaning of a particular DCMI *element* but with narrower semantics. Since *element refinements* are *properties* of a *resource* (like *elements*), *element refinements* can be used in metadata *records* independently of the *properties* they refine. In DCMI practice, an *element refinement* refines just one parent DCMI *property*.

value
: A *value* results when an *element* or *element refinement* is used to describe a *resource*.

record
: A *record* is some structured metadata about a *resource*, comprising one or more *properties* and their associated *values*.

value URI
: A *value URI* is a URI that identifies the value of an *element* or *element refinement*.

value string
: A *value string* is a simple string that represents the *value* of an *element* or *element refinement*. In general, a *value string* does not contain any markup.

value string language
: The *value string language* indicates the language of the *value string*.

encoding scheme
: An *encoding scheme* provides contextual information or parsing rules that aid in the interpretation of a *value string*. Such contextual information may take the form of controlled vocabularies, formal notations, or parsing rules. If an *encoding scheme* is not understood by a client or agent, the *value string* may still be useful to a human reader. There are two types of *encoding scheme*: *vocabulary encoding scheme* and *syntax encoding scheme*.

encoding scheme URI
: An *encoding scheme URI* is a URI that identifies an *encoding scheme*. For all DCMI recommended *encoding schemes*, the URI is constructed by concatenating the name of the *encoding scheme* with the DCTerms namespace URI.

vocabulary encoding scheme
: A *vocabulary encoding scheme* indicates that the *value string* is a term from a controlled vocabulary, such as the value "China - History" from the Library of Congress Subject Headings.

syntax encoding scheme

A *syntax encoding scheme* indicates that the *value string* is formatted in accordance with a formal notation, such as "2000-01-01" as the standard expression of a date.

marked-up text
A string that contains HTML, XML or other markup (for example TeX) and that is associated with the *value* of an *element* or *element refinement*.

rich value
Some *marked-up text,* an image, a video, some audio, etc. (or some combination thereof) that is associated with the *value* of an *element* or *element refinement*.

related metadata
A metadata *record* that describes a *resource* that is related to the *resource* described by a DC metadata *record*.

qualifier
A *qualifier* is the generic heading traditionally used for terms now usually referred to specifically as *element refinements* or *encoding schemes*.

structured value
A *structured value* is one of the following:

- A *value string* that contains machine-parsable component parts (and which has an associated *syntax encoding scheme* that indicates how the component parts are encoded within the string).
- Some *marked-up text*.
- Some *related metadata*

3. Qualified DC model

The abstract DC model for qualified DC metadata records is as follows:

- A *qualified DC record* is made up of one or more *properties* and their associated *values*.
- Each *property* is an attribute of the *resource* being described.
- Each *property* must be one of the *elements* or *element refinements* recommended by the DCMI [DCTERMS].
- *Properties* may be repeated.
- Each *value* may be identified by a *value URI*.
- Each *value* may have a *value string*.
- Each *value string* may have an associated *encoding scheme*.
- Each *encoding scheme* is identified by an *encoding scheme URI*.
- Each *value string* may have an associated *value string language* (e.g. en-GB).
- Each *value* may have an associated *rich value* (some marked-up text, an image, a video, some audio, etc. or some combination thereof).
- Each *value* may have some associated *related metadata*.

The italicised terms used in this description are defined in the terminology section above.

Note that, in general, the value string should be a string that does not contain any markup.

It recognised that many real-world metadata applications will use additional properties beyond those indicated in the third bullet point above. While such usage does not fall strictly within the definition of 'qualified DC' provided here, such applications are strongly encouraged to conform to the DCMI abstract model in order to achieve maximum interoperability with other DC metadata records.

4. Simple DC model

The abstract model for simple DC metadata records is a simplification of the qualified DC model defined above:

- A *simple DC record* is made up of one or more *properties* and their associated *values*.
- Each *property* is an attribute of the *resource* being described.
- Each *property* must be one of the 15 DCMES elements [DCMES].
- *Properties* may be repeated.
- Each value has a *value string*.
- Each *value string* may have an associated *value string language* (e.g. en-GB).

4.1 Dumb-down

The process of translating a qualified DC metadata record into a simple DC metadata record is normally referred to as 'dumbing-down'. In terms of the abstract model, the dumb-down algorithm is as follows:

1. Remove any *related metadata* and *rich values*.
2. Remove any *encoding schemes* (and *encoding scheme URIs*).
3. If a *value string* is not present and a *value URI* is present, use the *value URI* as the *value string*.
4. Repeatedly resolve any *properties* to their 'parent' *property* (as indicated by the 'Refines' attribute in the DCMI Metadata Terms recommendation [DCTERMS]).
5. Remove any resulting *properties* that are not one of the 15 DCMES elements [DCMES].

Note that software should make use of the DCMI term declarations represented in RDF schema language [DC-RDFS] and the DC XML namespaces [DC-NAMESPACES] to automate steps 4 and 5.

5. Encoding guidelines

Particular encoding guidelines (HTML meta tags, XML, RDF/XML, etc.) [DCMI-ENCODINGS] do not need to encode all aspects of the abstract models described above. However, DCMI recommendations that provide encoding guidelines should refer to the models described this document and should indicate which parts of the models are encoded and which are not. In particular, encoding guidelines should indicate whether any rich values or related metadata records associated with a value are embedded within a DC record or are encoded separately and linked to it using a URI.

References

DCMI
Dublin Core Metadata Initiative
<http://dublincore.org/>

DCTERMS
DCMI Metadata Terms
<http://dublincore.org/documents/dcmi-terms/>

DCMES
Dublin Core Metadata Element Set, Version 1.1: Reference Description
<http://dublincore.org/documents/dces/>

DCMI-ENCODINGS
DCMI Encoding Guidelines
<http://dublincore.org/resources/expressions/>

DC-RDFS
DCMI term declarations represented in RDF schema language
<http://dublincore.org/schemas/rdfs/>

DC-NAMESPACES
Namespace Policy for the Dublin Core Metadata Initiative (DCMI)
<http://dublincore.org/documents/dcmi-namespace/>

Acknowledgements

Thanks to Pete Johnston and the members of the DC Usage Board for their comments on previous versions of this document.

Appendix A - A note about structured values

This appendix discusses 'structured values', as they are used in DC metadata applications at the time of writing.

Many existing applications of DC metadata have attempted to encode relatively complex descriptions (i. e. descriptions that contain more than simply a property and its string value). These attempts have been loosely referred to as 'structured values'. It is possible to identify a number of different kinds of structured values. Four are enumerated below. The first two of these are recommended by the DCMI, in the sense that there are a number of existing encoding schemes that define values that conform to these definitions of structured values. The latter two are not currently recommended, but it is likely that they are in fairly common usage across metadata applications worldwide.

Labelled strings

These are value strings that contain explicitly labelled components within the string. Examples of this kind of structured value include:

DCSV

and the various DCMI syntax encoding schemes built on it - Period, Point and Box. An example of the use of DCSV in Period is:

```
<meta name="dcterms:temporal"
      scheme="dcterms:Period"
      content="start=Cambrian period; scheme=Geological
timescale; name=Phanerozoic Eon;" />
```

vCard

for example:

```
<meta name="dc:creator"
      content="BEGIN:VCARD\nORG:University of Oxford\nEND:VCARD
\n" />
```

Note that vCard is not currently a DCMI recommended encoding scheme.

Unlabelled strings

These are value strings that contain implicit components within the string, i.e. the components are determined based solely on their position within the string. Examples of this kind of structured value include:

W3CDTF

the date-time format used within most DC metadata. For example:

```
<meta name="dc:date"
      scheme="dcterms:W3CDTF"
      content="2003-06-10" />
```

Marked-up text

These are strings containing 'presentational' or other markup, for example adding paragraph breaks, superscripts or chemical/mathematical markup to a dc:description. It is possible to characterise various kinds of markup as follows:

- Markup based on a version of HTML.
- Markup based on other XML-based languages such as CML and MathML.
- Non-XML markup languages such as TeX.

Related resource description (related metadata)

These are metadata records that describe a second resource (i.e. not the resource being described by the DC record). For example, a related metadata record associated with a dc:creator property could contain a complete description of the resource author (including birthday, eye-colour and favourite beverage if desired!).

In the past, 'related resource descriptions' have tended to be encoded using XML, vCard (see above) or by inventing multiple 'refinements' of DCMES properties (for example DC.Creator.Address). The RDF/XML encoding of DC (see below) provides us with a more thorough modelling of related metadata records through the use of multiple linked nodes in an RDF graph.

In DC metadata records, the following elements (and their element refinements) are used to provide the name or identifier of a second resource that is related to the resource being described:

- dc:creator
- dc:contributor
- dc:publisher
- dc:relation
- dc:source

In the case of the first three, this is typically done by providing the name (or in some cases the name and a small amount of additional information in order to better identify the person or organisation) of the related resource as the value string.

In the case of the last two, this is typically done by providing the URI (or some other identifier) of the related resource as the value URI. However, where no identifier is available, the name of the related resource can be provided instead (or as well) using the value string.

It should be noted that the value strings of these elements (and their element refinements) are not intended to be used to provide full descriptions of the related resource.

Summary

The categories outlined above are not watertight and there are certainly overlaps between them. For example, labelled strings can be viewed as a type of non-XML markup language. In addition, there will be cases where marked-up text (e.g. MathML) can be viewed as a related resource description.

Nevertheless, the purpose of the categorisation used here is to try and analyse existing usage of complex metadata structures within current DC metadata applications. In the context of the abstract model proposed here, all the types of structured values outlined above form part of the qualified DC model:

- A labelled string should typically be treated as *related metadata* (though it should be noted that DCSV and the various DCMI syntax encoding schemes built on it - Period, Point and Box - are currently treated as *value strings* with an appropriate *encoding scheme*).
- An unlabelled string should be treated as a *value string* with an appropriate *encoding scheme*.
- Marked-up text should typically be treated as a *rich value*.
- A related resource description should be treated as *related metadata*.

Appendix B - RDF and the abstract model

This appendix discusses the relationship between the DC abstract model and the Resource Description Framework (RDF).

RDF currently provides DCMI with the richest encoding environment of the available encoding syntaxes. It is therefore worth taking a brief look at how the abstract model described here compares with the RDF model.

Note that the intention here is not to provide a full and detailed description of how to encode DC metadata records in RDF. Instead, three simple examples of the use of DC in RDF are considered.

Example 1: dc:creator

Figure 1 shows a simple RDF graph (and the RDF/XML document that represents it). The graph shows a resource with a single property (dc:creator). The *value* of the property is a second (blank) node, representing the creator of the resource. This second blank node has several properties, used to describe the creator, and an rdfs:label property that is used to provide the *value string* for the dc:creator property.

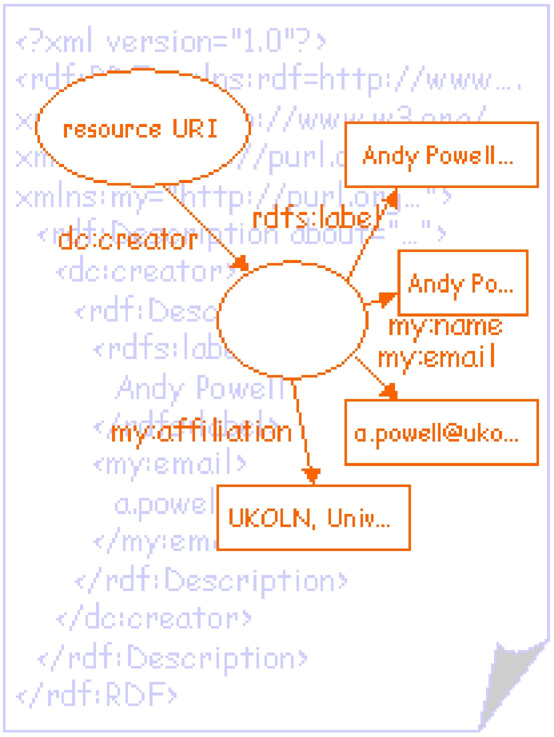


Figure 1

Figure 2 shows the same information separated into two graphs. In this case the *related metadata* that describes the creator has been more clearly separated from the description of the resource by moving it into a separate RDF/XML document. In order to do this, the node representing the *value* has been assigned a *value URI*, allowing the two nodes in the two RDF/XML documents to be treated as representing the same thing.

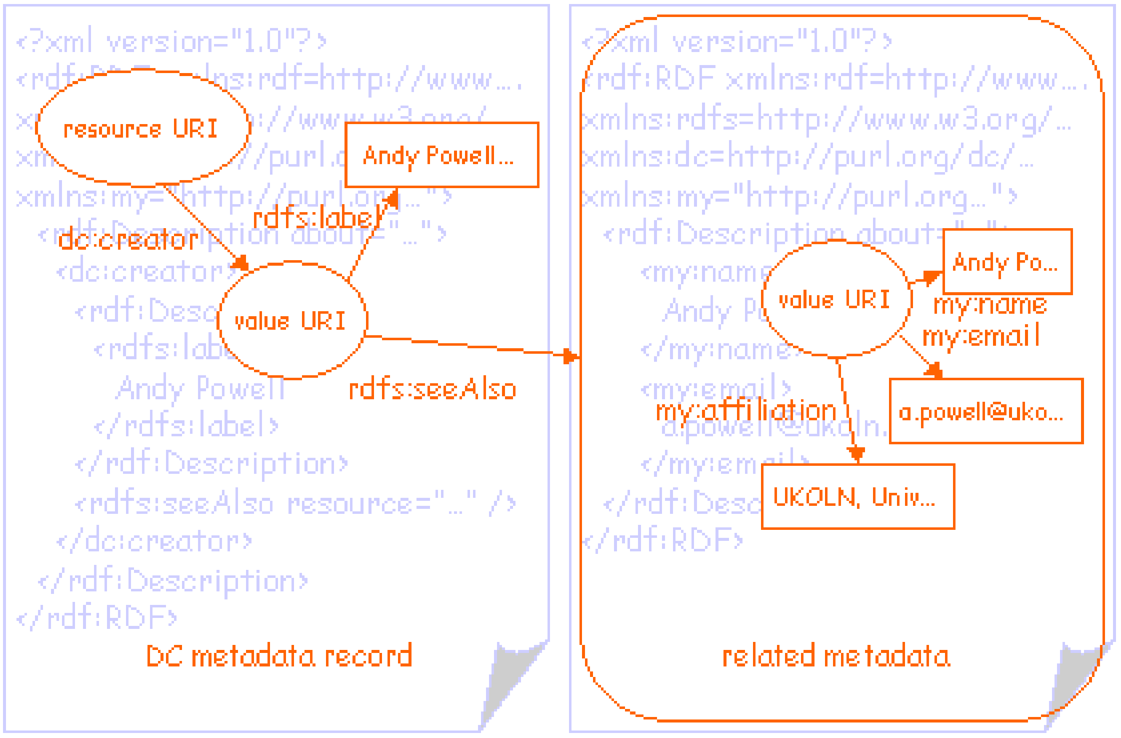


Figure 2

The *related metadata* in the second RDF/XML document is linked to the first using the rdfs:seeAlso property and the URI of the RDF/XML document. Note that it is not strictly necessary to separate the two graphs in this way; it is perfectly valid to represent the second graph as a sub-graph of the first, as shown in figure 1. However, for the purposes of this document, the two graphs have been separated in order to more clearly differentiate the DC metadata *record* from the *related metadata*. In some cases it will be good practice to facilitate this separation anyway. For example, in order to serve the second graph from a directory service of some kind.

Example 2: dc:subject

Figure 3 shows a second simple RDF graph (and the RDF/XML document that represents it). The graph shows a resource with a single property (dc:subject). The *value* of the property is a second (blank) node, representing the subject of the resource. This second blank node has an rdfs:

label property that is used to provide the *value string* for the dc:subject property, an rdf:value property that is used to provide the classification scheme notation and an rdf:type property to provide the encoding scheme URI.

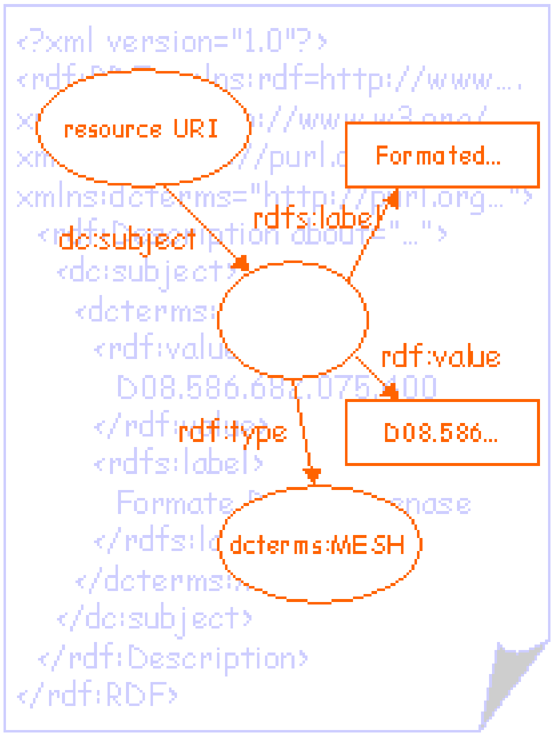


Figure 3

Figure 4 shows the same information separated into two graphs. In this case the *related metadata* that describes the subject has been more clearly separated from the description of the resource by moving it into a separate RDF/XML document. In order to do this, the node representing the *value* has been assigned a *value URI*, allowing the two nodes in the two RDF/XML documents to be treated as representing the same thing.

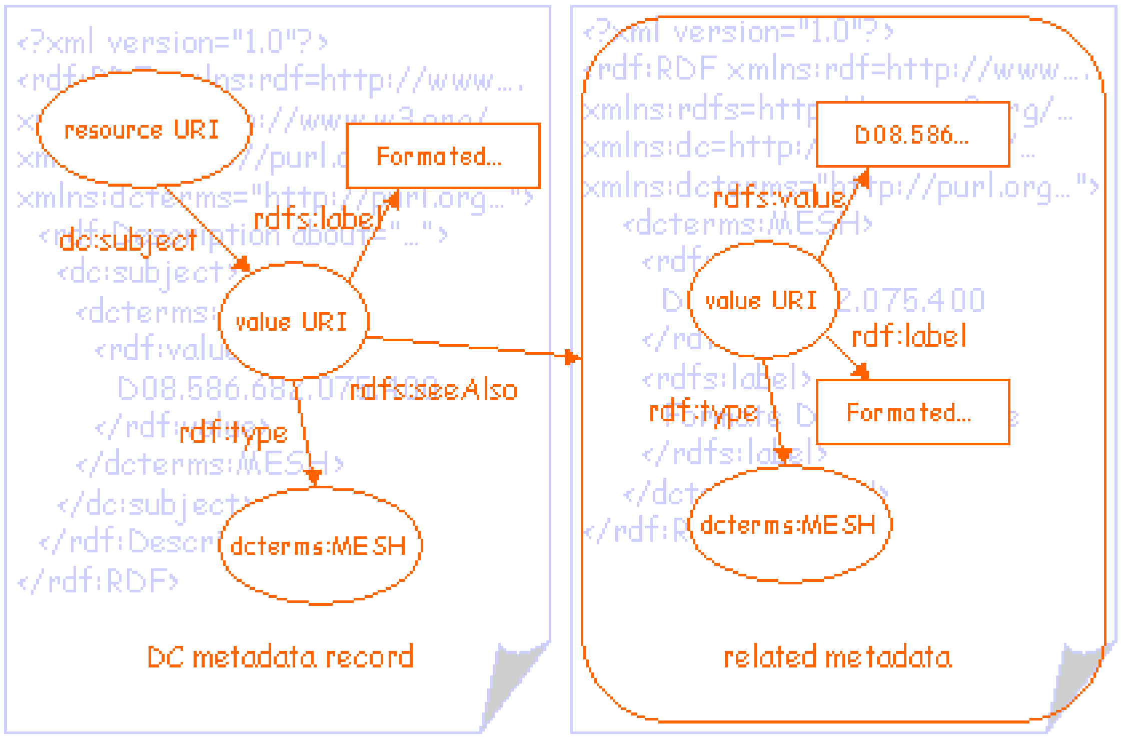


Figure 4

The *related metadata* in the second RDF/XML document is linked to the first using the rdfs:

seeAlso property and the URI of the RDF/XML document. Note that it is not strictly necessary to separate the two graphs in this way; it is perfectly valid to represent the second graph as a sub-graph of the first, as shown in figure 3. However, for the purposes of this document, the two graphs have been separated in order to more clearly differentiate the DC metadata *record* from the *related metadata*. In some cases it will be good practice to facilitate this separation anyway. For example, in order to serve the second graph from a terminology service of some kind.

Example 3: dc:description

Figure 5 shows a third simple RDF graph (and the RDF/XML document that represents it). The graph shows a resource with a single property (dc:description). The *value* of the property is a second (blank) node with an rdfs:label property that is used to provide the *value string* for the dc:description property. The second node also has an rdfs:

seeAlso property that links to a *rich value* - in this case some HTML marked-up text that provides a richer representation of the description.

Note that it is possible to embed the marked-up text within a single RDF graph (using rdf: parseType="Literal"). However, this is not shown here.

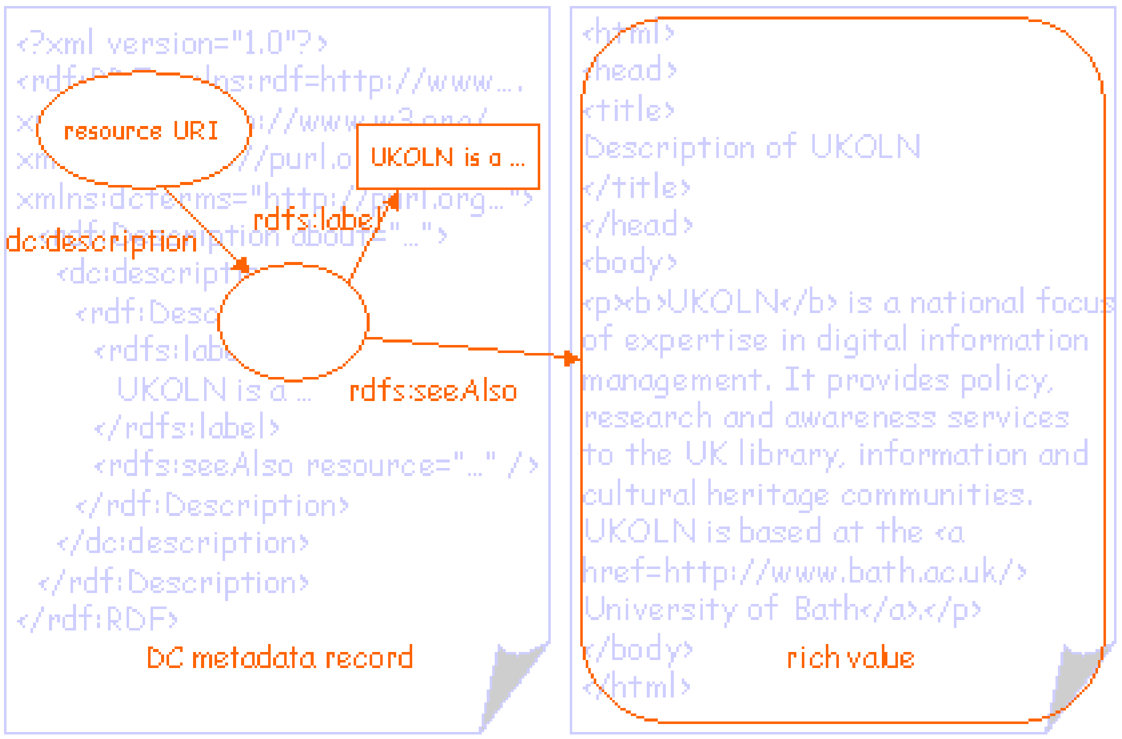


Figure 5

Summary

By re-visiting the second figure from example 2 (figure 4) it is possible to layer the terminology used in the abstract models above over the RDF graph.

A similar exercise could be undertaken for other encoding syntaxes (e.g. XML and HTML meta tags). As mentioned above, it may be the case that the full richness of the abstract model is not able to be offered in all encoding syntaxes.

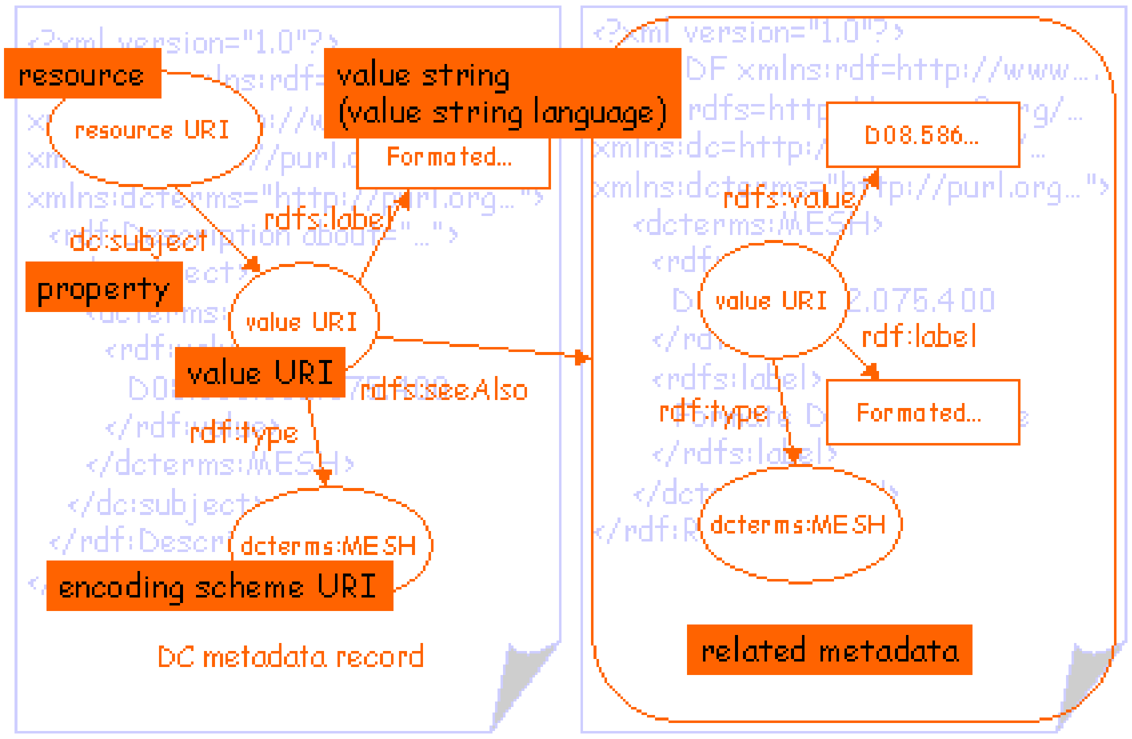


Figure 6

Metadata associated with this resource: http://dublincore.org/documents/abstract-model/index.shtml.rdf

```
Title:                  DCMI Grammatical Principles
Creator:                DCMI Usage Board
Identifier:             http://dublincore.org/usage/
documents/2003/02/07/principles/
Latest version:         http://dublincore.org/usage/documents/
principles/
Date modified:          2003-02-07
Description:            This document describes the grammatical
principles
                        that govern the decisions of the Usage Board as
the
                        maintenance body for DCMI metadata semantics.
See
                        also a related document, "DCMI Usage Board
                        Mission" [1], and the Dublin Core Metadata
Glossary
                        [8].
```

1. Scope of this grammar

This grammar presents the typology of DCMI metadata terms and describes the principles underlying their definition and use. As defined in the "Namespace Policy for the Dublin Core", a DCMI term is "a DCMI element, a DCMI qualifier or term from a DCMI-maintained controlled vocabulary." A DCMI namespace, in turn, is "a collection of DCMI terms" [2].

2. Elements and qualifiers

2.1. Elements

An Element is a property of a resource. As intended here, "properties" are attributes of resources -- characteristics that a resource may "have", such as a Title, Publisher, or Subject.

2.2. Qualifiers

"Qualifiers" is the generic heading traditionally used for terms now usually referred to specifically as Element Refinements or Encoding Schemes.

2.2.1. Element Refinements.

An Element Refinement is a property of a resource which shares the meaning of a particular DCMI Element but with narrower semantics. In some application environments (notably HTML-based encodings), Element Refinements are used together with Elements in the manner of natural-language "qualifiers" (i.e., adjectives) [3]. However, since Element Refinements are properties of a resource (like Elements), Element Refinements can alternatively be used in metadata records independently of the properties they refine [9]. In DCMI practice, an Element Refinement refines just one parent DCMI property.

2.2.2. Encoding Schemes.

An Encoding Scheme provides contextual information or parsing rules that aid in the interpretation of a term value. Such contextual information may take the form of controlled vocabularies, formal notations, or parsing rules. If an Encoding Scheme is not understood by a client or agent, the value may still be useful to a human reader. There are two types of Encoding Scheme:

2.2.2.1. Vocabulary Encoding Schemes

Vocabulary Encoding Schemes indicate that the value is a term from a controlled vocabulary, such as the value "China - History" from the Library of Congress Subject Headings.

2.2.2.2. Syntax Encoding Schemes

Syntax Encoding Schemes indicate that the value is a string formatted in accordance with a formal notation, such as "2000-01-01" as the standard expression of a date.

2.3. Dumb-down Principle

The qualification of Dublin Core Elements is guided by a rule known colloquially as the Dumb-Down Principle. According to this rule, a client should be able to ignore any qualifier and use the value as if it were unqualified. While this may result in some loss of specificity, the remaining term value (minus the qualifier) must continue to be generally correct and useful for discovery. Qualification is therefore supposed only to refine, not extend the semantic scope of an Element.

2.4. Appropriate values

Best practice for a particular Element or Qualifier may vary by context. Definitions may provide some guidance; other information may be found in the Usage Guide [6].

3. Vocabulary Terms

The Usage Board maintains the DCMI Type Vocabulary [7] -- a general, cross-domain list of recommended terms that may be used as values for the Resource Type element to identify the genre of a resource. The member terms of the DCMI Type Vocabulary are called Vocabulary Terms.

4. Application Profiles

In DCMI usage, an Application Profile is a declaration of which metadata terms an organization, information resource, application, or user community uses in its metadata [10].

REFERENCES

[1] http://dublincore.org/usage/documents/mission/
[2] http://dublincore.org/documents/2001/10/26/dcmi-namespace/
[3] http://www.ietf.org/rfc/rfc2731.txt
[4] http://www.ukoln.ac.uk/metadata/dcmi/dc-xml-guidelines/
[5] http://dublincore.org/documents/dcq-rdf-xml/
[6] http://dublincore.org/documents/usageguide/
[7] http://dublincore.org/usage/terms/dcmitype/
[8] http://dublincore.org/documents/2001/04/12/usageguide/glossary.shtml
[9] A shift from the former view to the latter is reflected in

```
    the names assigned by the Usage Board to Element
    Refinements, with a move away from adjective-like
    names such as "created" (approved in July 2000)
    towards noun-phrase-like names such as "dateCopyrighted"
    (approved in July 2002). One consequence of using Element
    Refinements independently of Elements is that information
    about relationships between them will reside outside of
    the metadata records in separate schemas that applications
    needing to perform operations such as dumb-down will need
    to consult.
[10] http://dublincore.org/usage/documents/profiles/
```

ARTICLES

D-Lib Magazine
October 2000

Volume 6 Number 10

ISSN 1082-9873

A Grammar of Dublin Core

Thomas Baker
GMD -- German National Research Center for Information Technology
Scientific Library and Publication Services
Schloss Birlinghoven
D-53754 Sankt Augustin, Germany
thomas.baker@gmd.de

"We begin with the sentence, because the sentence is the unit of discourse, because words can be classified only from their function in the sentence, and because the pupil should, from the outset, see that what determines the words in the sentence and the sentence itself is the thought."

Alonso Reed and Brainerd Kellogg, *Higher Lessons in English*, 1877

Dublin Core Made Simple

Dublin Core as a Language. Dublin Core is often presented as a modern form of *catalog card* -- a *set* of elements (and now qualifiers) that describe resources in a complete *package*. Sometimes it is proposed as an *exchange format* for sharing records among multiple collections. The founding principle that "every element is optional and repeatable" reinforces the notion that a *Dublin Core description* is to be taken as a whole. This paper, in contrast, is based on a much different premise: Dublin Core is a ***language***. More precisely, it is a small language for making a particular ***class of statements*** about resources. Like natural languages, it has a vocabulary of word-like terms, the two classes of which -- elements and qualifiers -- function within statements like nouns and adjectives [1]; and it has a syntax for arranging elements and qualifiers into statements according to a simple pattern [2].

A Pidgin for Digital Tourists. Whenever tourists order a meal or ask directions in an unfamiliar language, considerate native speakers will spontaneously limit themselves to basic words and simple sentence patterns along the lines of "***I am*** so-and-so" or "***This is*** such-and-such". Linguists call this pidginization. In such situations, a small phrase book or translated menu can be most helpful. By analogy, today's Web has been called an Internet Commons where users and information providers from a wide range of scientific, commercial, and social domains present their information in a variety of incompatible data models and description languages. In this context, Dublin Core presents itself as a metadata pidgin for digital tourists who must find their way in this linguistically diverse landscape. Its vocabulary is small enough to learn quickly, and its basic pattern is easily grasped. It is well-suited to serve as an auxiliary language for digital libraries.

This Grammar. This grammar starts by defining terms. It then follows a 200-year-old tradition of English grammar teaching by focusing on the structure of single statements (see the Reed and Kellogg quote above). It concludes by looking at the growing dictionary of Dublin Core vocabulary terms -- its *registry*, and at how statements can be used to build the metadata equivalent of paragraphs and compositions -- the *application profile*.

Elements and Qualifiers

Vocabulary terms in general. Strictly speaking, a Dublin Core element or qualifier is a unique identifier formed by a name (e.g., title) prefixed by the URI of the namespace in which it is defined, as in http://dublincore.org/2000/03/13-dces#title. In this context, a namespace is a vocabulary that has been formally published, usually on the Web; it describes elements and qualifiers with natural-language labels, definitions, and other relevant documentation. Currently there are two namespaces for Dublin Core: the Dublin Core element set and the Dublin Core Qualifiers, denoted here by the conventional abbreviations dc: and dcq: [DCMI 1999, DCMI 2000]. In this paper, as in many application environments, the elements and qualifiers are referred to in a machine-readable short form, such as dc:title.

Elements. The fifteen elements of the Dublin Core element set are the defining feature of Dublin Core as a language. In their short form, the elements are dc:title, dc:creator, dc:subject, dc:description, dc:publisher, dc:contributor, dc:date, dc:type, dc:format, dc:identifier, dc:source, dc:language, dc:relation, dc:coverage, and dc:rights. These correspond to fifteen broadly defined *properties* of resources that are generally useful for searching across repositories in multiple domains.

Qualifiers. Qualifiers modify the properties of Dublin Core statements by specifying, in the manner of natural-language adjectives, "what kind" of subject, date, or relation. Qualifiers currently fall into two classes. *Encoding schemes* are pointers to contextual information or parsing rules that aid in the interpretation of an element value. For example, dcq:lcsh qualifies dc:subject to specify that the keywords are a Library of Congress Subject Heading, and dcq:iso8601 qualifies dc:date to specify that the string "2000-06-13" is formatted according to an international standard. *Element refinements* make a property more specific without extending its meaning, such as dcq:revised as a modifier of dc:date (yielding "date revised"). In July 2000, a DCMI Usage Committee recommended a batch of fifty-two qualifiers to exemplify these principles of qualification, but the qualifiers for Dublin Core statements may also come from other namespaces, as in the example yans:author below (where yans: is a hypothetical YetAnotherNameSpace).

Elements and qualifiers defined in languages other than English. Strictly speaking, elements and qualifiers are represented by machine-readable tokens that stand for general concepts such as "title," "subject," and "date." The Dublin Core Metadata Initiative discusses and approves their definitions in English. In principle, however, they can be labelled and defined equally well in any other language, such as Dutch or Arabic or Thai. For example, dc:creator may be labeled "Creatore" in Italian, "Pencipta" in Bahasa Indonesian, or "Verfasser" in German [Baker 1998]. To date, the element set has been translated into twenty-six languages. Bear in mind as you read that although this grammar is written in English, a Japanese version could translate every English word here into Japanese -- all grammar terms and example sentences included -- *except* for the English-like names of the tokens themselves.

Dublin Core Statements

Diagramming statements. Since the 1870s, the grammar of sentences has been taught in (American) high schools using sentence diagrams [Reed and Kellogg 1886, House and Harman 1950, Warriner et al. 1973]. This style has a binary flavor -- the sentence baseline is intersected to divide the *subject*, "that of which something is said," from the *predicate*, that which is said of the subject. Within the predicate, a smaller line separates the *object* (in Dublin Core terms, the property) from the *objective complement* (in Dublin Core, the property value). This style is nicely expressive of Dublin Core because the qualifiers, hanging below the baseline on slanted lines, are visibly subordinate to the properties they modify (see Figures 1 and 2).

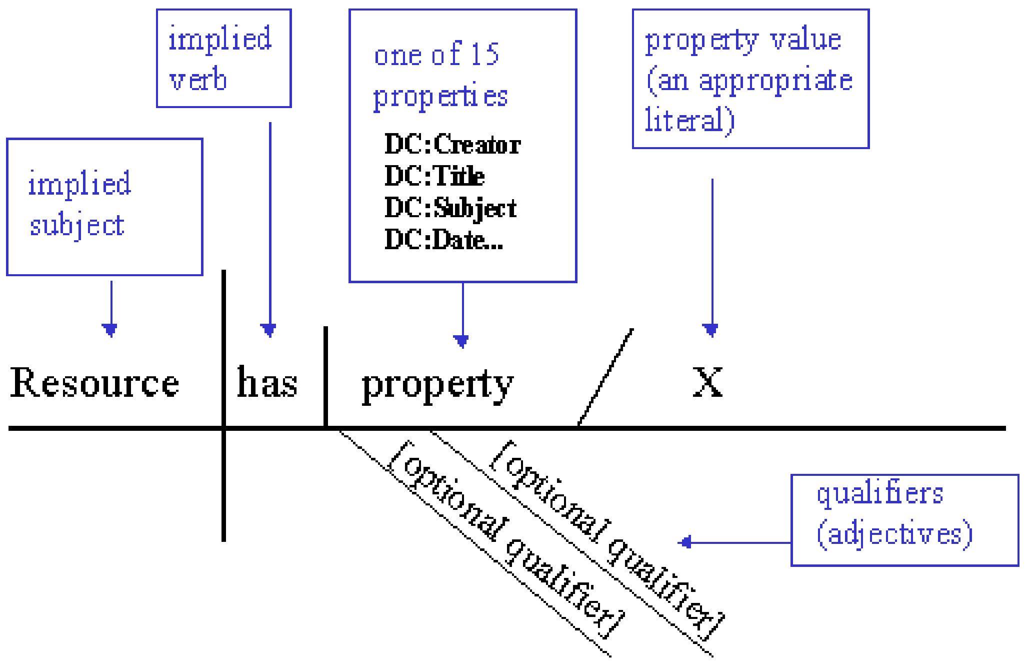


Figure 1: The general pattern

Parts of a Statement. Dublin Core is in effect a class of statements of the pattern "Resource has property X," where "resource" is the implied subject; followed by an implied verb ("has"); followed by one of fifteen properties from the Dublin Core element set; followed by a property value -- an appropriate literal such as a person's name, a date, some words, or a URL. For example: "Resource has dc:creator 'Tom Baker'," and "Resource has dc:date '2000-06-13'." Optional qualifiers may make the meaning of a property more definite, as in "Resource has dc:date dcq:revised '2000-06-13'."

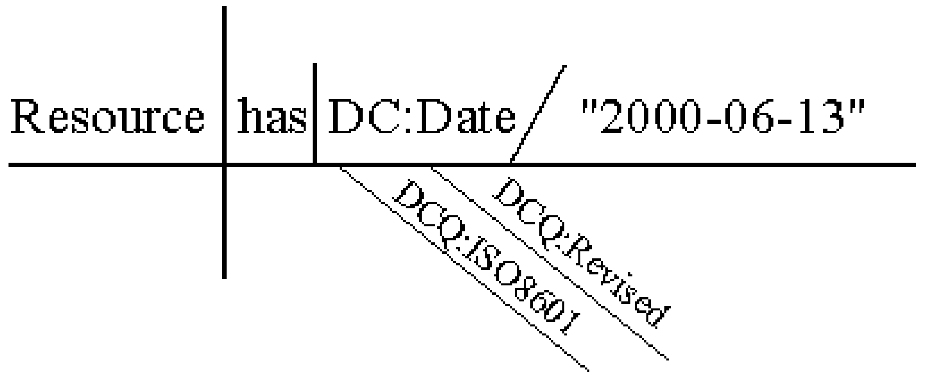
Principles of Qualification. The qualification of Dublin Core properties is guided by a rule known colloquially as the Dumb-Down Principle. According to this rule, "a client should be able to ignore any qualifier and use the description as if it were unqualified. While this may result in some loss of specificity, the remaining element value (minus the qualifier) must continue to be generally correct and useful for discovery" [DCMI 2000]. Qualification is therefore supposed only to refine, not extend the semantic scope of a property. In borderline cases, qualification should not result in a literal that could be misleading.

Appropriate Literals. Whether a property value is "useful for discovery" is at the heart of the notion of appropriateness. A property value should be a string of an expected type -- usually, for example, some sort of name for dc:creator, dc:contributor, dc:publisher, or dc:title; a URL for dc:relation, dc:identifier, or dc:source; full-text sentences for dc:description; short text strings or keywords for dc:subject, dc:type, dc:format, and dc:language; and a recognizable combination of years, months, and days for dc:date. Both in theory and in practice, the range of expected data types varies from property to property; which types are appropriate for a given property is open to interpretation and debate (see below).

Evaluating Statements. To test whether a Dublin Core statement is conceptually solid, cover the qualifiers with your hand ("dumbing down"), read the statement above the line, and ask:

- Does it make sense?
- Is it factually and logically correct?
- Is the literal "appropriate" for the given property?

Examples



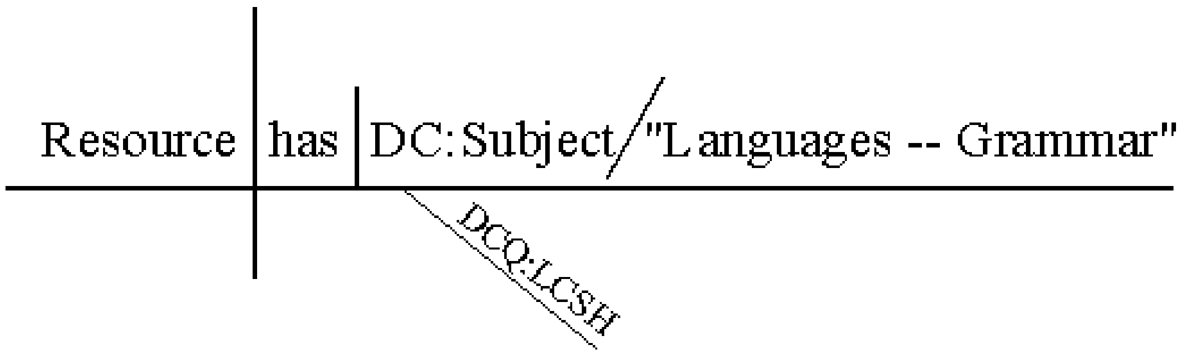


Figure 2: Some statements, diagrammed

Resource has dc:title 'A Grammar of Dublin Core.'

> Does it make sense? Yes. Is it correct? Yes. Is the literal "appropriate"? Yes, a sequence of words is normal and expected for the property dc:title.

Resource has dcq:iso8601 dcq:revised dc:date '2000-06-13.'

> This means that a resource was revised on 6 June 2000. The statement dumbs down to "Resource has dc: date '2000-06-13'," which means that the date 6 June 2000 has something to do with the life-cycle of the resource. This is less specific than the qualified statement, but still correct.

Resource has dcq:lcsh dc:subject 'Languages -- Grammar.'

> This says that the resource is about the subject "grammar of languages," and that these words are a controlled term from the Library of Congress Subject Headings. The statement dumbs down to "Resource has dc:subject 'Languages -- Grammar'," which makes sense even if we do not know that the term comes from the Library of Congress.

Resource has yans:cerif dc:subject 'H352 Grammar, semantics, semiotics, syntax.'

> This literal includes a language-independent abbreviation, "H352," which will be useful for applications that understand yans:cerif. The string "H352" may confuse some users, but otherwise does no harm.

Resource has yans:author dc:creator 'Tom Baker.'

> Users of the yans: namespace will recognize that yans:author is being used here as an adjective modifying dc:creator (just as *talk* is an adjective modifying *show* in the phrase *talk show*). It is an awkward but correct way to say that Tom Baker is "the author sort of creator" -- i.e., the author -- of the resource. This statement dumbs down to "Resource has dc:creator 'Tom Baker'," -- less specific but still correct.

Resource has dc:relation 'http://www.dlib.org/dlib/december98/12baker.html.'

> The statement asserts that the resource is somehow related to an article in *D-Lib Magazine* (a URL is appropriate as a literal for dc:relation).

Some not-so-good examples

Resource has dc:creator 'name.given:Thomas; name.family:Baker; employer:GMD; contact:Schloss Birlinghoven, D-53754 Sankt Augustin.'

> A reader can see what this *compound* or *structured value* is saying. But a search engine would need to know how to parse out the components and suppress the tags in order to index this cleanly, lest a search for creators named "Augustin" should yield false hits. Generally speaking, things like affiliations and addresses -- properties *of the creator* of a resource -- do not belong in Dublin Core statements about the resource itself. Metadata providers that used such compound values within specific usage communities could "speak Dublin Core" to the rest of the world by exporting just the name ("Resource has dc:creator 'Thomas Baker'").

Resource has yans:creator 'Tom Baker.'

> This statement is useful for applications that recognize the yans: namespace, but it is not a "Dublin Core" statement per se. If the yans: and dc: definitions of Creator were compatible, either the metadata provider or an indexing application could use a crosswalk to translate this into the Dublin Core statement "Resource has dc:creator 'Tom Baker'."

Ongoing Issues

Fifteen fuzzy buckets. The properties of Dublin Core are like fifteen big buckets, and the rules about which types of literals may be placed in those buckets are somewhat fuzzy. This fuzziness is intentional -- the Internet is a diverse and chaotic place where a more disciplined, top-down approach to standardization is unrealistic, especially for use across multiple domains and languages. If the rules of Dublin Core were more precise, people would inevitably bend them. In the jargon of computer science, then, Dublin Core is "weakly typed" as a language. A search engine may find a variety of information types in any given bucket -- from URLs to non-textual, alphanumeric strings to full text in any language.

The "appropriateness" of literals. Requirements for the appropriateness of a literal are in practice somewhat contradictory. Ideally, a literal should be useful for discovery, which means it should make sense "as is" to the average user. Yet it should also be processable in an expected way by search engines. Programmers need to be know, for example, when to index on strings separated by white space, minus punctuation and stop words, when to expect a URL, and when to expect an alphanumeric date string. Some elements are particularly ambiguous in this regard. Dc:rights, for example, can be free text or a URL. Dc:coverage can be a place name, the name of a time period, a numeric identifier for a place or time, or even a compound value -- in effect, a miniature schema with multiple sub-components separated by semicolons or XML tags. Where such a range of data types is permissible, should it be acceptable to relax the Dumb-Down Principle?

Should qualifiers in effect modify the expected data type of the literal? Or should the presence of, say, XML angle brackets be expected to trigger, automatically, a change in parsing algorithms? Would such a complexification of a property compensate for the corresponding loss of Dublin Core's overall simplicity? Or can the need for complex description be resolved in a broader framework, outside Dublin Core per se?

Application profiles. One broader framework for such a resolution is the *application profile*. As currently defined, application profiles are the metadata equivalent of regional idioms or creoles (complexified pidgins). Implementors who need an application language more expressive than a pidgin may combine elements and qualifiers from Dublin Core with elements from other namespaces into a richer vocabulary or embed them into a syntactically more sophisticated data model. Such linguistic innovation is considered by many people to be reasonable as long as implementors respect a distinction between namespaces, where elements and qualifiers are given standard definitions, and profiles, where elements from multiple namespaces are (only) reused, combined, adapted, and constrained [Heery 2000]. The profile, then, is the natural locus for full descriptions -- the catalog card or metadata package taken as a whole.

For example, the Collection Description Schema of the Research Support Libraries Programme (RSLP) in the UK uses dc:title -- officially defined as a "name given to the resource" -- but defines it more narrowly as a "name given *to the collection*." Alongside such Dublin Core elements, it uses elements from other namespaces, such as cld:accessConditions (for the hours of access and classes of permitted users) from a local "Collection Level Description" namespace. These elements are framed in a data model that specifies typical relationships between a collection, its individual items, a collector, an owner, a location, and the constituent parts of a collection -- each of which may be described with multiple attributes [RSLP]. An RSLP description does not talk just about information resources per se, but also about the people, organizations, and access frameworks related to those resources.

Developing profiles and coining new elements. Some working groups of DCMI are developing domain-specific profiles of Dublin Core, surveying the descriptive needs of domains such as education and government to determine an appropriate mix of Dublin Core elements and elements from other namespaces and perhaps to coin additional elements for concepts not covered in existing standards. These working groups need to consider that literals appropriate to domain specialists may not make much sense to general users, especially in statements that have been "dumbed down." As the example above makes clear, moreover, core elements are needed for classes of resources other than document-like objects, such as people and organizations (generically, *agents*) and spatially and temporally grounded *events*. Urgently required are data-model conventions for combining multiple entities within an application profile -- for example, to include an author's affiliation and address in the description for a resource -- and stable formats for the encoding of profiles as XML or RDF schemas.

Building a dictionary (registry) for Dublin Core. Historically, the standardization of national languages such as English has been helped by the compilation of dictionaries. Good dictionaries often strike a balance between *pre*scribing guidelines for good style and *de*scribing a living language with examples of actual usage. Metadata languages like Dublin Core have hitherto been developed prescriptively, in standards committees, as there have been no convenient ways to track local innovations in usage and feed them back into the standardization process. However, several related developments are now enabling the collective construction of metadata dictionaries, or registries. The new Resource Description Framework (RDF) Schemas standard of the World Wide Web Consortium provides a format for publishing schemas that can be harvested by metadata search engines [W3C 2000]. Eric Miller has developed an open-source software toolkit for indexing a distributed corpus of RDF schemas as one huge database, with an interface for following hyperlinked cross-references between related terms in namespaces and profiles -- in effect, a metadata schema browser [EOR]. The Dublin Core Metadata Initiative is using this toolkit to manage its namespace [Open Metadata Registry], and a working group is formulating technical and policy guidelines for its ongoing management [DC-Registry]. A European project, SCHEMAS, is promoting the use of RDF schemas to help harmonize metadata practice among EU-funded projects and is using RDF to build a layer of annotated pointers to namespaces, profiles, and metadata activities generally [SCHEMAS].

Does your application speak Dublin Core? Pidgins are inherently limited in what they can express, but they are easy to learn and enormously useful. In real life, we talk one way to our professional colleagues and another way to visitors from other cultures. Our digital library applications need to do this as well. Simplicity and complexity are both appropriate, depending on context. If Dublin Core is too simple or generic to use as the native idiom of a particular application, pidgin statements may be extracted or translated from richer idioms that exist for specialized domains. This output should also be filtered to keep the fifteen buckets clear of encoding debris and semantic silt. One should treat digital tourists with courtesy and hide from them the complexities of a local application vocabulary or grammar. However sophisticated its local idiom may be, an application might also speak a pidgin that general users and generic search engines will understand. Simple, semantically clean, computationally obvious values will help us negotiate our way through a splendidly diverse and heterogeneous Internet.

Appendix: Dublin Core and RDF grammar compared

Directed Labelled Graphs. The Resource Description Framework (RDF), a relatively new standard of the World Wide Web Consortium, is emerging as an information model and encoding format of choice for metadata and application profiles that use Dublin Core [W3C 1999, W3C 2000]. RDF is a grammar for expressing relationships among resources located or represented somewhere on the Internet. These relationships are depicted graphically with Directed Labelled Graphs (DLGs), which use arcs (predicates expressing properties) to establish a relationship between multiple nodes (resources). Nodes are seen as subjects or objects depending on the direction of the arrow.

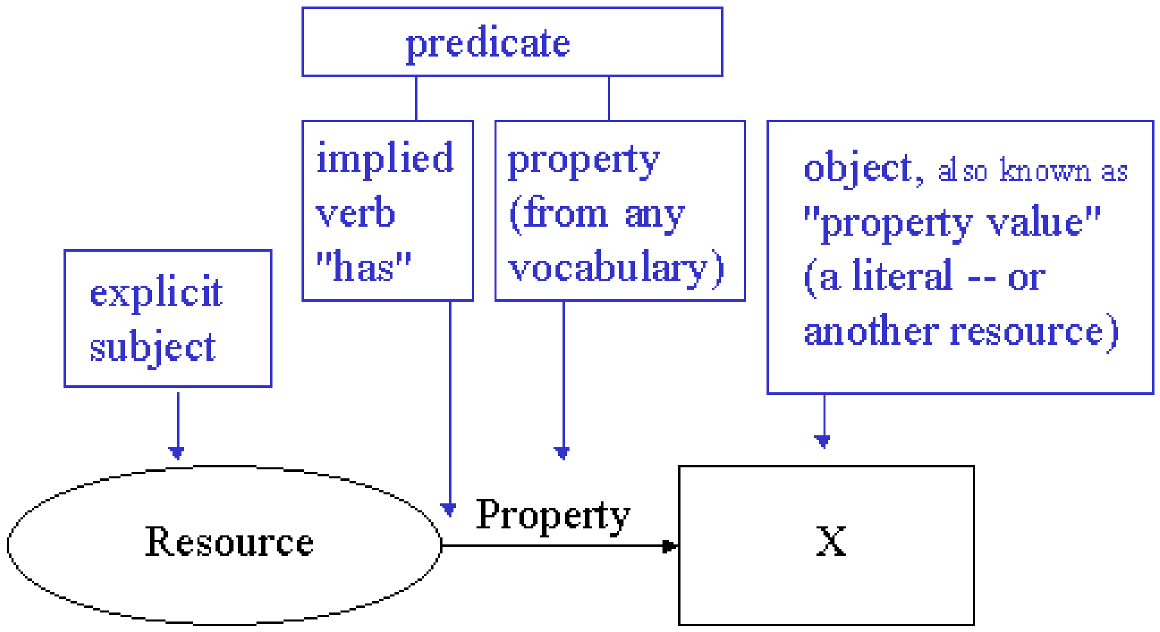

predicate
implied verb "has"
property (from any vocabulary)
explicit subject
object, also known as "property value" (a literal -- or another resource)
Resource
Property
X

Figure 3: The general pattern of RDF statements ("triples")

RDF triples. In contrast to the binary flavor of the pedagogical English grammars cited above, the model of RDF statements is a tripartite one of *subject*, *predicate*, and *object* (see Figure 3). RDF statements, accordingly, are called "triples":

> A *property* is a specific aspect, characteristic, attribute, or relation used to describe a resource. ... A specific resource together with the named property plus the value of that property for that resource is an RDF *statement*. These three individual parts of a statement are called, respectively, the *subject*, the *predicate*, and the *object*. The object of a statement (i.e., the property value) can be another resource or it can be a literal; i.e., a resource (specified by a URI) or a simple string or other primitive datatype defined by XML. In RDF terms, a *literal* may have content that is XML markup but is not further evaluated by the RDF processor [W3C 1999].

Figure 4 shows an RDF triple consisting of the subject "http://www.w3.org/Home/Lassila," predicate "has creator," and object "Ora Lassila."

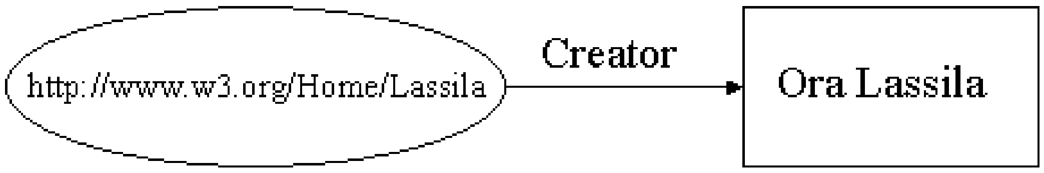


Figure 4: Directed Labelled Graph of a triple

Subject of an RDF statement. The subject of an RDF statement is anything that can be named by a Universal Resource Identifier (URI). In principle, this is anything from a Web page or museum artifact to an abstract concept or event. The subject of a Dublin Core statement, in contrast, is an anonymous "information resource," perhaps a "document-like object" [3]. In Dublin Core, the subject ("resource") is always implied, never named as in RDF. The RDF statement in Figure 4, then, would be expressed in Dublin Core as two statements: "Resource has dc:identifier 'http://www.w3.org/Home/Lassila'," and "Resource has dc:creator 'Ora Lassila'." As long as it is clear from the context that the two statements refer to the same resource -- for instance, they are both in the same record -- one can infer that Ora Lassila has created the Web page [4].

English	Dublin Core	RDF
words **(have classes)**	**tokens** **(have classes)**	**nodes and arcs**
→ nouns	→ elements	
→ adjectives	→ qualifiers	
sentences **(have parts)**	**statements** **(have parts)**	**statements** **(have parts)**
→ subject	→ implied subject	→ subject
→ predicate		→ predicate
→ → verb	→ → implied verb	→ → implied verb
→ → object	→ → property	→ → property
→ → objective complement	→ → property value	→ object (property value)
→ modifiers	→ → qualifiers	

Figure 5: Grammar Terms Compared

Predicate of an RDF statement. The predicate of an RDF statement is an implied verb plus a property. This is more restricted than the standard definition of a predicate in high-school English grammar, which includes (in effect) "everything to the right of the vertical intersector in a sentence diagram" -- that is, everything within a clause that is said about a subject. However, since most people have only the vaguest recollection of predicates from high school, this grammar avoids using the term at all.

Property value of RDF statements. In RDF, these are called *objects*. But again, there is risk of confusion with English grammar. In the sentence "This paper has the title 'A Grammar of Dublin Core'," some high-school English grammars see 'A Grammar of Dublin Core' as an *objective complement* with respect to the *object* of the sentence, "title" (see Figure 5). Compounding the confusion, computer scientists are oriented to *objects* of a much different sort, and the "resource" of a Dublin Core statement might even be a physical *object*. This grammar avoids the term *object* altogether.

The lack of qualifiers in RDF. In Dublin Core, qualifiers depend on and modify one of Dublin Core's fifteen elements (properties). The basic RDF model does not express this type of dependency. Properties may relate to another as narrower to broader terms (using the relation "SubPropertyOf"), as "Author" relates to "Creator." In native RDF, however, both "Author" and "Creator" are full properties in their own right.

The compatibility of Dublin Core and RDF. The differences between Dublin Core and RDF outlined above are largely terminological; at issue is what the parts of a statement such as 'Resource has Property X' should be called. The difficulty of choosing the right words for this grammar should not obscure the basic compatibility and complementarity of Dublin Core and RDF. RDF offers a general model for statements, while Dublin Core offers a particular type of pidgin-like statement about information resources and privileges a small set of special words. RDF offers a specific encoding in XML for expressing its conceptual model, while Dublin Core is by design independent of any particular encoding format. RDF, then, is just one of the possible information models that can use tokens from Dublin Core, while Dublin Core is just one of the languages expressible in RDF.

Footnotes

[1] It would be desirable to have a word for Dublin Core vocabulary terms in general -- elements and qualifiers as a whole -- just as natural-language nouns, verbs, adjectives, and conjunctions are all called *words*. Unfortunately, *entities*, *concepts*, and *symbols* are all too abstract and vague; *words* are associated too closely with natural language; and *lexemes* are too obscure. The first draft of this paper spoke generically of *elements* and distinguished between *core elements* and *qualifiers*, but veterans of Dublin Core found this confusing. The next draft introduced *tokens*, which conveys the notion that Dublin Core vocabulary terms stand for general concepts that are defined and labeled in many natural languages, but people also found this confusing. *Vocabulary terms* seems a bit cumbersome, so I avoid the term whenever possible and speak simply of *elements and qualifiers*.

[2] Early in the workshop series, before it was formally called a Metadata Initiative, the Dublin Core effort was declared to be primarily about *semantics* in contrast to *syntax*, and the latter was declared to be out of scope. In that context, however, *syntax* referred to the encoding of metadata in HTML, database, and (later) XML or RDF formats; it involved questions such as which tags to place where, within what angle brackets or punctuation, and how to group or nest related elements. This grammar, in contrast, presents the *syntax* of Dublin Core statements in a linguistic sense, as the rules governing how the words of a sentence are related to each other -- which words modify other words, and which words are of central importance in the statement.

[3] In practice, the reasoning behind this is circular: Dublin Core properties are appropriate for any entity that has such properties. This circularity has spared us a perhaps futile attempt to seek philosophical consensus on a universal ontology of entity classes and allowed us to get on with the task of describing whatever it is we are describing.

[4] Dan Brickley points out that software tools could exploit additional information about entities and vocabularies to translate sequences of RDF-encoded Dublin Core statements into a natural-language style that flows more elegantly and reads less like a pidgin.

Acknowledgements

I have discussed this paper with many colleagues from both DCMI and other communities. Special thanks to Bill Arms, Jose Borbinha, Dan Brickley, David Bearman, Rachel Heery, Linda Hill, Diane Hillmann, Erik Jul, Carl Lagoze, Sigfrid Lundberg, Eric Miller, Paul Miller, Andy Powell, Shigeo Sugimoto, and Stuart Weibel for valuable comments on the drafts. Many thanks to Michael Kasper for advice regarding the literature on English grammar.

References

[Baker 1998] Thomas Baker, "Languages for Dublin Core," *D-Lib Magazine*, December 1998, < http://www.dlib.org/dlib/december98/12baker.html>.

[Berners-Lee 1998] Tim Berners-Lee, "Why RDF model is different from the XML model," < http://www.w3.org/DesignIssues/RDF-XML.html>.

[DCMI 1999] Dublin Core Metadata Initiative, "Dublin Core Metadata Element Set, Version 1.1," < http://purl.org/dc/documents/rec-dces-19990702.htm>.

[DCMI 2000] Dublin Core Metadata Initiative, "Dublin Core Qualifiers," < http://purl.org/dc/documents/dcmi-qualifiers>.

[DC-Registry] Dublin Core Registry Working Group, < http://www.mailbase.ac.uk/lists/dc-registry/>.

[EOR] EOR Tookit: Web-based search interface tools for RDF RDMS systems,
< http://eor.dublincore.org/index.html>.

[Heery 2000] Rachel Heery, "Application profiles: mixing and matching metadata schemas,"
< http://www.ariadne.ac.uk/issue25/app-profiles>.

[House and Harman 1950] Homer C. House and Susan Emolyn Harman, *Descriptive English Grammar*, *Second Edition*, Englewood Cliffs, N.J.: Prentice-Hall, Inc.

[Open Metadata Registry] The Open Metadata Registry,
<http://wip.dublincore.org/registry/OpenRegistry>.

[Reed and Kellogg 1886] Alonso Reed and Brainerd Kellogg, *Higher lessons in English*, New York: Clark and Maynard, 1886 [1877]; Delmar (New York): Scholars' Facsimiles and Reprints, 1987.

[RSLP] RSLP Collection Description Schema,
< http://www.ukoln.ac.uk/metadata/rslp/schema/>.

[SCHEMAS] SCHEMAS: A Forum for Metadata Schema Implementers,
< http://www.schemas-forum.org>.

[W3C 1999] Ora Lassila and Ralph Swick, eds., "Resource Description Framework (RDF) Model and Syntax Specification," [W3C Recommendation],
< http://www.w3.org/TR/REC-rdf-syntax>.

[W3C 2000] Dan Brickley and R. V. Guha, eds., "Resource Description Framework (RDF) Schema Specification 1.0," [W3C Candidate Recommendation],
< http://www.w3.org/TR/WD-rdf-schema>.

[Warriner et al. 1973] John E. Warriner et al., *English Grammar and Composition, Third course*, New York: Harcourt Brace Jovanovich, Inc.

The following four changes and corrections were made at the request of the author, Thomas Baker, on 10/17/00): 1.) In the paragraph "Does your application speak Dublin Core?" one phrase was changed from: "its elements can be embedded in a richer local vocabulary from which pidgin statements can be generated or extracted as needed." to read "pidgin statements may be extracted or translated from richer idioms that exist for specialized domains." 2.) The name Linda Hill was added to the "Acknowledgements" section. 3.) In the paragraph "Vocabulary terms in general", "(e.g., Title)" was changed to "(e.g., title)." 4.) In the commentary of the first of the "Not-so-good examples," the sentence "Resource has dc:creator 'Tom Baker'" was changed to: "Resource has dc:creator 'Thomas Baker'".

Top | **Contents**
Search | **Author Index** | **Title Index** | **Monthly Issues**
Previous Article | **Next Article**
Home | **E-mail the Editor**

DOI: 10.1045/october2000-baker

DCMICite: a Bibliographic Citation Dublin Core Structured Value (DCSV) Encoding Scheme

A Proposal to the DCMI Usage Board from the DCMI Citation Working Group

21 August 2003

This version: <http://epub.mimas.ac.uk/DC/citdcsv-20030821.html>
Previous version: This is the first version for review
Latest version: <http://epub.mimas.ac.uk/DC/citdcsv.html>

Creator: Ann Apps <ann.apps@man.ac.uk>
MIMAS, University of Manchester, UK

Status of this document: *Under Review*

Description: This document contains a proposal from the DCMI Citation Working Group to the DCMI Usage Board for a DCMICite Dublin Core Structured Value (DCSV) Encoding Scheme for *dcterms:bibliographicCitation*. Use of this encoding scheme will provide a means to record, within a string value, the bibliographic citation information that identifies a journal article, within its own metadata, in a form that is both human readable and machine parsable.

Name:	DCMICite
Label:	DCMI Cite
Definition:	DCMICite identifies a journal article by its bibliographic citation information encoded within a Dublin Core Structured Value (DCSV) [1]. It is a 'semi-colon separated' string of 'key=value' pairs and includes the following keys, which are optional and repeatable where appropriate: **journalTitle** : The title of a journal, ie. a serial publication. This may include the sub-title, but not extra information such as its affiliation with a society. **journalAbbreviatedTitle** : The abbreviated title of a journal, such as it may appear in a reference list. It may follow some recommended (but unidentified) scheme such as ISO4, Chemical Abstracts, Index Medicus, Vancouver, World List. Note that this may not be unique because different schemes may use the same

	abbreviation for different journal titles. **issn** : An ISSN global standard identifier of a journal. **eissn** : An ISSN global standard identifier of an electronic journal where this differs fom the print journal ISSN. **journalVolume** : A number of a journal volume, in arabic or roman form, as it appears on the print journal cover or electronic journal home page. **journalIssueNumber** : The issue, part or number that denotes a particular issue of a journal, as it appears on the print journal cover or electronic journal home page. In many cases this indicates a part of a journal volume. **journalIssueDate** : The formal date of a particular issue of a journal, as it appears on the print journal cover or electronic journal home page. **pagination** : The inclusive page range of an article within a particular journal issue, from first page to last page (if known).
Condition:	Sufficient bibliographic citation information should be included to identify the article unambiguously, and it must not contain conflicting information. The minimum identification requirement is: Journal identification (ie. at least one of: journalTitle, issn, eissn) Sufficient volume and issue or date identification Start page.
Condition:	It is important to note that a DCMICite must *identify* an article. It should not be used with the purpose of recording related metadata such as the bibliographic information about the containing journal or issue, which would be recorded in a dcterms: isPartOf relation.

Examples:	journalTitle=Library and Information Science Research; journalAbbreviatedTitle=LISR; journalVolume=22; journalIssueNumber=3; journalIssueDate=2000; pagination=311-338;
	
	issn=1359-6462; journalVolume=48; journalIssueNumber=5; pagination=475

Type of term:	Encoding Scheme
Term qualified:	dcterms:bibliographicCitation

Why needed:	Dublin Core has become recognised as the core standard for metadata interoperability in many application domains. But it is very apparent that there is currently no Dublin Core recommended method to encode the bibliographic citation information for a journal article within its own metadata in a consistent and machine parsable way. There are continual requests from people in the digital library and publishing communities for advice on how to encode this information in Dublin Core. The lack of any Dublin Core recommendation is causing ad hoc solutions that have a negative impact on interoperability.
Proposed status:	Conforming
Related DCMI terms:	DCMICite could be used informally as the value of *dc:identifier* in simple DC.
Related non-DCMI terms:	The proposed ANSI/NISO OpenURL Framework Standard (Z39.88-2003) [2] includes a *journal* metadata format that can be encoded within a string. This would provide a different encoding format and the DCMI Citation Working Group intends to provide guidelines on its use. However, it would be an alternative to DCMICite rather than a replacement for it: OpenURL encoding is too cryptic for human readability, values being URL-encoded for transport over HTTP; because of its main application as 'appropriate copy' linking data, of which the journal metadata format is but a part, it requires the inclusion of further cryptic key/value pairs; and it potentially includes metadata items that would be recorded in other DC properties within a DC record.
Impact on applications:	There should be no impact because DCMICite does not change any existing recommendation. However it may have an impact on applications that are using their own ad hoc methods for recording this information, in particular where they are recording this information in different DC properties.
Further information	Background information about the proposal.
About the proposers:	DCMI Citation Working Group
References:	[1] Cox, S. and Ianella, R. (2000) DCMI DCSV: A syntax for writing a list of labelled in a text string. http://dublincore.org/documents/dcmi-dcsv/ [2]NISO Committee AX: Development of an OpenURL Standard. http://library.caltech.edu/openurl/

Electronic Publishing

DCMICite Backgrond Information

DCMI Citation Working Group

21 August 2003

This version: <http://epub.mimas.ac.uk/DC/citdcsvbackgd-20030821.html>
Previous version: This is the first version for review
Latest version: <http://epub.mimas.ac.uk/DC/citdcsvbackgd.html>

Creator: Ann Apps <ann.apps@man.ac.uk>
MIMAS, University of Manchester, UK

Status of this document: *Background information for proposal under review*

Description: This document contains background information to a proposal from the DCMI Citation Working Group to the DCMI Usage Board for a DCMICite Dublin Core Structured Value (DCSV) Encoding Scheme for *dcterms:bibliographicCitation*.

---

1. Bibliographic Citations

It should be noted that the bibliographic information described here is the complete bibliographic record for the resource itself. This DCMICite proposal is not concerned with capturing citation linking data for a related resource.

1.1 Journal Article Bibliographic Citation Properties

The properties generally used to capture the bibliographic citation of a journal article may be identified at three distinct levels: the **Journal** level; the journal **Issue** level (which may also include a journal Volume level); and the individual **Article** level. The following table indicates these properties according to this hierarchical level, and where appropriate which Dublin Core element is already available to record the information.

Level	Property	DC Element
Journal	Journal Title	
	Journal Abbreviated Title	

	Journal Identifier	
Issue	Volume	
	Number	
	Chronology	
Article	Article Title	dc:title
	Author	dc:creator
	Publisher	dc:publisher
	Publication Year	dc:date
	Publication Date	dc:date
	Identifier	dc:identifier
	Pagination	

Notes:

- Journal Abbreviated Title may adhere to a published scheme, eg. ISO4, Chemical Abstracts, Index Medicus, Vancouver, World List. Often it is an abbreviated journal title which is included in a reference to an article rather than the full title of the journal. Note that this may not be unique because different schemes may use the same abbreviation for different journal titles.
- A journal identifier would usually be an ISSN. A journal may have more than one ISSN to differentiate the print journal and the electronic journal.
- Issue Numbers are denoted differently in different journals, eg. `part'. Some journals are arranged by year, eg. 12/1999, in which case the year is effectively the volume.
- Chronology is the `cover date' as it appears on the cover of a printed journal. This may be different from the actual date of publication of the issue, which would be encoded in *dcterms:issued*.
- Chronology could be denoted by a season or quarter.
- An article identifier could be: a URL to the actual article; a DOI; an article-level SICI; etc. Note that the only scheme for *dc:identifier* ratified by Dublin Core is URI.
- Other information about an article may be recorded within other elements of the DC record. For instance, its abstract could be captured in *dc:description*, its language in *dc: language* and keywords or classification information in *dc:subject*.
- Only the most common bibliographic information about an article has been included here. A Journal Issue could also have an identifier, such as an issue-level SICI, but this is less commonly used.

It is apparent from this table that there is currently no method within Dublin Core to capture the bibliographic citation of a journal article, except by recording the information in an ad hoc way within a *dc:description* element, or by capturing the metadata in a hierarchical manner.

2. Proposed Encoding for Journal Articles

The proposed recommended method for capturing bibliographic citation information about journal articles in Dublin Core is as follows:

- The bibliographic citation information for a journal article should be captured within a *dcterms:bibliographicCitation* element encoded using a **DCMI Cite** DCSV.
- This information may include:
  - Journal Title, Journal Abbreviated Title, Journal Identifier
  - Volume, Issue Number, Chronology
  - Pagination
- In addition a Journal Identifier or a Journal Issue Identifier may be included in a *dcterms:isPartOf dc:relation* element refinement
- All other information about an article, such as its title and authors, should be included as appropriate within the 15 elements of the basic Dublin Core Metadata Element Set

It should be noted that this recommendation is based on the print model of journal publishing. At the present time, recording the position of an article within a printed journal is the generally used model and a requirement for reference linking. Most journals are still published according to this model, and many cited articles appear in older journals which were originally published in print. Where articles published in electronic-only journals are cited the pagination and possible the issue numbering information is irrelevant, though it would by necessity be replaced by some other numbering.

3 Example of a Journal Article Metadata Record

This is an example of a Dublin Core record for a journal article including its bibliographic citation information. This example is independent of any syntax recommendation apart from the DCMI Cite DCSV. Any other syntax and any line breaks used in this example are for clarification purposes only. Note that this example also includes alternative identifiers for the article

```
dc:title = Studying E-journal User Behavior Using Log Files
dc:creator = Lu, Y
dc:creator = Apps, A
dc:subject(scheme=DDC) = 020
dc:description = Statistical methods for analysing e-
journal user behaviour.
dc:publisher = Pergamon
dcterms:issued(scheme=W3CDTF) = 2000
dc:type(scheme=DCMIType) = text
dcterms:medium(scheme=IMT) = application/pdf
```

```
dc:identifier(scheme=URI) = doi:10.1060/xyz.abc
dc:identifier(scheme=URI) = urn:sici:07408188(200010)
22:3<311:SEUB>2.0.CO;2-X
dcterms:bibliographicCitation(scheme=DCMICite) =
  journalTitle=Library and Information Science Research;
  journalAbbreviatedTitle=LISR;
  journalVolume=22;
  journalIssueNumber=3;
  journalIssueDate=October 2000;
  pagination=311-338;
dc:language(scheme=RFC1766) = en
dcterms:isPartOf(scheme=URI) = urn:issn:0740-8188
dc:rights = © Elsevier, 2000
```

4. Other Possible Solutions

Past discussions of the DCMI Citation Working Group have looked at alternative methods and properties for capturing bibliographic citation information about a journal article.

4.1 Using *dc:description*

The advantages of the *dc:description* solution would be to retain the essential simplicity of Dublin Core, and that the information would be presented to someone discovering the metadata in a human-readable way. The latter point is important, and any solution should provide data to a human end-user in a readily understandable form - they may wish to find the article on a library shelf. The disadvantage of using *dc:description* is that it becomes difficult to perform further machine processing on the discovered metadata, which may be required for discovery of the location of the article.

4.2 A Hierarchical Solution

It would be possible to partially record the information in a hierarchical way, using a *dc:relation* element to point from an article record to a record for its containing issue, and similarly from the issue record to that for the containing journal. There are two drawbacks to this solution:

- There is no obvious DC element in which to encode the pagination information, which pertains to the article itself, not to the containing journal issue.
- It may be that there is insufficient knowledge of the containing journal and issue to allow for discovery of the article from such hierarchical metadata. An end-user would expect to receive all the information about an article following a search in one piece.

It could be thought that including information about a journal in the metadata for an article breaks the `one-to-one' rule. However the objective here is not to provide information about the journal, but rather to provide a bibliographic citation of an article, which effectively identifies it.

4.3 An Application Profile

Another possible method for including in a metadata record information which doesn't fit obviously into one of the 15 elements of the Dublin Core element set (DCMES), or the ratified qualifiers, is to introduce application specific elements and qualifiers within an application profile[4]. Thus an option would be to define a `citation' profile and include new elements such as `journal title' within it. However, capturing bibliographic citation information seems to be a generic, cross-domain problem. The bibliographic citation of a journal article is fairly fundamental information, required within many subject areas, at least for academia and researchers. It is information which is becoming increasingly significant with the implementation of linking technologies. Therefore it would seem sensible to have a best practice convention for capturing journal article citation information within Dublin Core metadata using existing elements, rather than a proliferation of application profiles attempting to solve the same problem in different ways with new application specific elements.

4.4 OpenURL

The emerging OpenURL [5] standard may provide an alternative encoding scheme for capturing journal article bibliographic citation information within a single string. When OpenURL becomes a NISO/ANSI standard, the DCMI Citation Working Group will produce guidelines for its use as a possible value for a *citation* identifier. But it would be an alternative to the proposed *DCMI Cite* DCSV rather than a replacement for it. However, there are reasons for not recommending OpenURL as a citation encoding scheme as a replacement for DCMICite:

- The proposed *DCMI Cite* DCSV is relatively human-readable, whereas an OpenURL is more cryptic, values being URL-encoded for transport over HTTP. OpenURL is suitable only for situations where the citation is for machine processing.
- Because of its main application as 'appropriate copy' linking data, of which the journal metadata format is but a part, it requires the inclusion of further cryptic key/value pairs
- It potentially includes metadata items that would be recorded in other DC properties within a DC metadata record
- In situations where bibliographic citation information is to be keyed manually, it would be difficult to use an OpenURL, whereas using *DCMI Cite* would be straightforward. It is possible that this keyed information would also be for machine processing, so a plain text citation would be unsuitable.

4.5 Previous Work

How to record a bibliographic record for a journal article has previously been discussed by an earlier Dublin Core Citation Working Group, whose recommendation made after the DC7 Workshop was 'Citation Qualifier Proposal - 2000' [6], which also includes details of a vote by the general DC community at the plenary session at DC8 Workshop.

5. References

[1] DCMI Citation Working Group. http://www.dublincore.org/groups/citation

[2] A *citation* Element Refinement for *dc:identifier* http://epub.mimas.ac.uk/DC/citproposal.html

[3] Cox, S. and Ianella, R. (2000) DCMI DCSV: A syntax for writing a list of labelled in a text string. http://dublincore.org/documents/dcmi-dcsv/

[4] Heery, R. and Patel, M. (2000) Application profiles: mixing and matching metadata schemas. *Ariadne* **25**, September 2000. http://www.ariadne.ac.uk/issue25/app-profiles

[5]NISO Committee AX: Development of an OpenURL Standard. http://library.caltech.edu/openurl/

[6] Citation Qualifier Proposal - 2000. http://www.dublincore.org/groups/citation/citqualifier2000.html

---

MIMAS Electronic Publishing

Dublin Core Used Here

W3C XHTML 1.0

```
Date:         Fri, 5 Apr 2002 13:01:25 +0100
Reply-To:     Electronic discussion list to support the efforts of the Agent
              working grou <DC-AGENTS@JISCMAIL.AC.UK>
Sender:       Electronic discussion list to support the efforts of the Agent
              working grou <DC-AGENTS@JISCMAIL.AC.UK>
From:         "Clayphan, Robina" <Robina.Clayphan@BL.UK>
Subject:      Re: Qualifiers for Creator, Contributor, Publisher
Comments: To: Dublin Core Libraries Application Profile
          <DC-LIBRARIES-AP@JISCMAIL.AC.UK>,
          "DC-LIBRARIES@JISCMAIL. AC. UK (E-mail)" <DC-LIBRARIES@JISCMAIL.AC.UK>
Content-Type: text/plain; charset="iso-8859-1"
```

Dear Colleagues,

Apologies for multiple copies.

This draft document is one of two proposals answering the third milestone
defined for the Agents Working Group at the DC 2001 conference [1] and
referred to in Rebecca's earlier email.

Milestone: Develop a list of CCP (Creator / Contributor / Publisher)
qualifiers

Description: Develop a list of CCP (Creator / Contributor / Publisher)
qualifiers. This would likely use the MARC relator codes as a point of
departure, and define such things as agent refinements, classes, types and
roles

Introduction

This proposal has been discussed on the DC Libraries Application Profile
list but is not intended solely for that domain. Comment is invited from
other domains.

As a result of discussion it was decided to separate the issue of agent
Roles from other agent-related data and offer two proposals. One will
describe a means of recording a Role for an agent and another will suggest a
means to encompass other information about the resource that is also related
to the agent. This document is the second of these and describes a Dublin
Core Structured Value (DCSV) [2] for use as an encoding scheme with the
Creator, Contributor or Publisher elements.

It should be noted that unlike previous agent proposals, this DCSV is NOT an
attempt to describe the agent - that work is being progressed elsewhere in
the Agents Working Group. This DCSV aims to enhance resource discovery in
two ways: by adding further markup and therefore specificity to CCP values
(e.g. parts of names); by including further agent-related information about
the resource (e.g. affiliation at the time the resource was created).

Previous discussion of agent information had identified four Types of Agent:
personal, corporate, instrument (variously called entity, service, automata)
and event. Although each of these was assumed to have different components,
further consideration indicated that, given some latitude in the
definitions, the same group of components could serve for all types. All
the components in the DCSV are optional and those not relevant to any
particular type of agent can be ignored.

Proposed DCMI Agent Detail Encoding Scheme

The proposal requires a new encoding scheme, DCMI Agent Detail, which
follows the Dublin Core Structured Value syntax [2] for encoding a list of
labelled values within a text string. The Dublin Core Structured Value
definition has the status of a DCMI recommendation.

All components are optional. Different domains are free to specify which
components are desirable for the types of agent identified as valuable
within that domain. For example, the Library Application Profile would
provide guidance as to which components to use for agents appropriate to
library resources. Examples are given after the description of the
components.

DCMI Agent Detail
Name: DCMIAgentDetail
Label: DCMI Agent Detail
Definition: DCMI Agent Detail contains additional information about the
agent in its capacity as Creator, Contributor or Publisher of the resource
being described. It is not intended to describe the agent.

The proposed labels which may be encoded within DCMI Agent Detail are:

Family Name
Label: familyName
Definition: The family name or surname of the agent.

Note: Components for parts of names are given to avoid dependence on
parsing the value based on punctuation and the difficulties that can arise
due to different cultural norms. Specific markup of values gives greater
flexibility to users of the data for filing uses and enhanced resource
discovery.

Given Names
Label: givenNames
Definition: The personal name or names of the agent.

Name
Label: name
Definition: The full name of the agent, personal, corporate or of another
type, expressed in the manner recommended by any particular domain.
Note: An undifferentiated name component is included to accommodate names
that do not require the family name/given name sub-elements or applications
that choose not to use the sub-elements. In the latter case different
domains are free to specify their own guidance as to how the value should
be entered.

Affiliation Location
Label: affiliationLocation
Definition: The affiliation or location of the agent at the time they made
their contribution to the described resource. This can be the name of an
institution in the case of a personal agent or an address or geographic
name in the case of a publisher or corporate agent.
Note: It is not envisaged that this would be updated, the rationale being
that e.g. the affiliation of the author at the time of the creation of the
resource is an attribute of some significance to the resource and remains
the same for the resource even if the agent subsequently moves on.

Description
Label: description
Definition: To specify the type or nature of the agent. For example to
specify an event or instrument/automata/service. (See examples)

Date Time
Label: dateTime
Definition: The date and time at which the agent executed its role in the
life of the resource.
Note: This is regarded as important for event-type agents where the date
the agent executed the given role may differ from the date the resource was
created or issued.

Identifier
Label: identifier
Definition: An identifier for the agent.
Note: This component is included to allow for the inclusion of an agent
identifier and/or to enable linking to another record describing the agent
held separately from the record for the resource.

Role was considered as a component of the DCSV. A separate proposal is being submitted to include role values as element refinements for the Agent elements. This approach will be re-evaluated if necessary.

Jurisdiction has been proposed as an additional component of the DCSV. The following definition is put forward here for discussion and clarification.
Label: jurisdiction
Definition: The local, provincial, regional, national or supranational authority to which a corporate body belongs.
Note: For example, Jurisdiction may be "Australia" when Name is "Department of Finance and Administration".

DCSV Examples
These examples assume that Role values have been approved as CCP element refinements. Encoding schemes for the Identifier component within the DCSV are shown using the namespace convention which may or may not be an acceptable usage within a component. Line breaks are for the sake of clarity only.

Example 1 (HTML). Showing a person as creator, an organisation as sponsor and a publisher.

```
<meta name="DC.Creator.author"
scheme="DCMIAgentDetail"
content="familyName=Clayphan;
givenName=Robina;
affiliationLocation=The British Library;
identifier=AAAA:xxxx">
<meta name="DC.Contributor.sponsor"
scheme="DCMIAgentDetail"
content="name=Some Organisation;
affiliationLocation=London, England;
identifier=BBBB:yyyy">
<meta name="DC.Publisher" scheme="DCMIAgentDetail"
content="name=Some Publisher;
affiliationLocation=London, England;
identifier=CCCC:zzzz">
```

The following examples are guesses at how other domains would use the DCSV for the types of agent that have been mentioned in past discussions - principally the instrument/automata/service agent and event agent, which equates to conferences in current library usage. Role qualifiers have not been entered in either example. Guidance and examples are sought from other domains.

Example 2: showing an instrument/automata/service agent.
(This is for a sound recording of a steam engine.)

```
<meta name="dc:Contributor"
scheme="DCMIAgentDetail"
content="name=The Flying Scotsman;
```

description=Steam Engine;
identifier=GNER:4472">

Example 3: showing an event agent as contributor.
<meta name="dc:Contributor"
scheme="DCMIAgentDetail"
content="name=Glastonbury Festival 2000;
affiliationLocation=Glastonbury, UK;
description=Music and performing arts festival;
dateTime=2000-06">

Regards,
Robina Clayphan

1 http://dublincore.org/groups/agents/
2 http://dublincore.org/documents/2000/07/28/dcmi-dcsv/

Robina Clayphan, Metadata Development Analyst,
National Bibliographic service, The British Library
Tel: +44 (0)1937 546969
Fax: +44 (0)1937 546586
email: robina.clayphan@bl.uk

Back to: Top of message | Previous page | Main DC-AGENTS page

Title:	**DCMI DCSV: A syntax for writing a list of labelled values in a text string**
Creator:	Simon Cox
Creator:	Renato Iannella
Date Issued:	2000-07-28
Identifier:	http://dublincore.org/documents/2000/07/28/dcmi-dcsv/
Replaces:	http://dublincore.org/documents/2000/07/11/dcmi-dcsv/
Is Replaced By:	Not Applicable
Latest version:	http://dublincore.org/documents/dcmi-dcsv/
Status of document:	This is a DCMI Recommendation.
Description of document:	We describe a method for recording lists of labelled values in a text string, called Dublin Core Structured Values, with the label DCSV. The notation is intended for structured information within attribute values in markup-languages such as HTML and XML. This is likely to be useful in recording complex element values in metadata systems based on the qualified Dublin Core model.
NOTICE TO IMPLEMENTORS:	The syntax examples included in this document are provisional, and are currently under review as part of the DCMI work on recommending coordinated syntax recommendations for HTML, XML, and RDF. These recommendations and minor editorial changes in this document can be expected to take place in the near future. **Note that the use of "=" as a separator is a change from earlier versions of this specification which used ":" in the same position.** This change was considered desirable because the ":" character occurs frequently within strings which are likely to be used as names and values. Using "=" as a separator reduces the need to escape characters in the data.

Table of Contents

Introduction

It is highly desirable to be able to encode or *serialise* structured values within a plain-text string. Some generic methods are in common use. Inheriting conventions from natural languages, commas (,) and semi-colons (;) are frequently used as list separators. Similarly, comma-separated-values (CSV) and tab-separated-values (TSV) are common export formats from spreadsheet and database software, with *line-feeds* separating rows or tuples. Dots (.) and dashes (-) are sometimes used to imply hierarchies, particularly in thesaurus applications. The eXtensible Markup Language [XML] provides a general solution, using tags contained within angle brackets (<, >) to indicate the structure.

A number of named encoding schemes use punctuation characters within a text string to indicate specific components. For example, a colon (:) terminates the protocol label, and slashes (/), question-marks (?), ampersands (&) and hashes (#) are used to separate other fields in identifiers coded as URI's [URI]. Colons (:) separate specified labels from values within a field, and semi-colons (;) separate fields within a personal description according to a common implementation of vCard [vCard]. Hyphens are used to separate fields in a date according with the W3C profile of ISO8601 [W3C-DTF]. For some schemes - vCard and W3C-DTF, for example - the punctuation indicates a very formal structure to the value, and is expected to be parsed automatically.

Element *attributes* in markup languages, such as HTML [HTML4] and XML [XML], provide a position for recording data. For some "empty" elements - such as the <IMG > and <META > elements in HTML - attributes are the only place to hold data. In other cases there may be good reasons to store data in element attributes rather than element content. For example, fragments of XML can be included in the <HEAD> of a HTML document, and will be safely ignored by most client software (eg browsers) *provided the elements have no content*. This syntax trick can be used to embed XML-RDF encoded data safely in current versions of HTML [RDF-in-HTML].

Future versions of HTML are expected to overcome these limitations by allowing general XML documents to be included [XHTML]. Nevertheless, there is strong interest in using HTML <META > elements to record data with more structure than normally implied by a plain-text string, in particular to record metadata according to the qualified Dublin Core model [Q-DC-HTML].

However, the use of element attributes for storing data has some technical limitations:

1. attributes may occur no more than once
2. values are constrained to a set of types which restricts the permissible character-strings [HTML4] in some contexts. Use of XML's angle-bracket delimiters (<, >) and various other punctuation characters is only valid in certain cases (i.e. when the content type is CDATA), and is only generally reliable using escape-mechanisms (i.e. as *character entities*). In general, strings containing these characters are prone to misinterpretation by some user-agents (e.g. browsers).

Note that there is no intrinsic way to indicate structure within the values of attributes of HTML elements.

Our intention in this recommendation is to define a compact human-readable data-structuring method for HTML attribute values of content type CDATA, avoiding certain punctuation characters which are prone to cause difficulties in some encoding environments. The notation should normally be used only when no other suitable scheme is available. It is based on methods used and found successful elsewhere, but is more generalised than the preceding standards. It may be used as the basis of profiles designed to encode particular data types [Profiles].

Structured Values - the DCSV scheme

To allow the recording of generic **Structured Values**, we introduce the Dublin Core Structured Values (**DCSV**) scheme.

We distinguish between two types of substring - *labels* and *values*, where a label is the name of the type of a value, and a value is the data itself. Furthermore, we allow a complete value to be disaggregated into set of *components*, each of which has its own label and value. A value that is comprised of components in this way is called a *structured value*.

Punctuation characters are used in recording a structured value as follows:

- equals-signs (=) separate plain-text labels of structured value-components from the values themselves
- semi-colons (;) separate (optionally labelled) value-components within a list
- dots (.) indicate hierachical structure in labels, if required.

The labels and the component values themselves each consist of a text-string. The intention is that the label will be a word or code corresponding to the name of the value-component. Labels may be absent, in which case the entire sub-string delimited by semi-colons (;) or the end of the string comprise a component value.

The following patterns show how structured values may be recorded in strings using DCSV:

```
"u1; u2; u3"
"cA=v1"
"cA=v1; cB.part1=v2; cB.part2=v3"
"cA=v1; u2; u3"
```

where u1, u2 and u3 are unlabelled components, cA and cB are the labels of Structured Value components, part1 and part2 are sub-components of cB, and v1, v2 and v3 are values of the components.

The use of specific punctuation characters in DCSV coded values means that care must be exercised if these characters are to be used directly within strings which comprise the content (either labels or values) of the components. For DCSV, therefore, when an equals-sign (=), or a semi-colon (;) is required within the value, the characters are escaped using a backslash, appearing as \= \;. There should be no ambiguity regarding the dot, full-stop or period (.) within strings: when it is part of a label, a dot indicates some hierachy; when part of a value, it has the conventional meaning for the context. This method of escaping special characters largely preserves readability and the ability to enter DCSV coded metadata values easily using a text-editor if required. Software written to process DCSV coded values must make the necessary substitutions.

Note that in HTML the double-quote (") character can be used directly within a CDATA attribute value if the full string is delimited by single-quotes ('), but in XML the double-quote must be encoded as a character entity in element attributes.

As there is no explicit grouping mechanism, DCSV can only be used to record a list. DCSV is only intended to be used for relatively simple structured values, probably as an interim approach, pending more general support for syntaxes such as XML which allow recording of more complex hierarchical structures. However, it is more compact than the XML equivalent, and is more easily read and constructed in some common contexts, such as within HTML <meta > elements.

Parsing DCSV

A simple method can be used to parse metadata values recorded according to the DCSV scheme. For a single value recorded using the DCSV scheme:

1. split the text-string into a list of substrings on any unescaped semi-colons (;);
   if no semi-colon is present, there is a single substring
2. split each substring into its (label,value) on any unescaped equals-signs (=);
   if no = is present, the label is empty
3. within each value replace the escaped characters with the actual character required.

A short Perl program which performs this parsing operation is included at the end of this recommendation.

Examples

"name.given=Renato; name.family=Iannella; employer=DSTC; Contact=Level 7, Gehrmann Labs, The University of Queensland, Qld. 4072, Australia"
"rows=200; cols=450"

The DCSV scheme provides useful support for the representation of complex values for metadata elements in HTML, while remaining fully compatible with all commonally used tools (browsers, editors, metadata harvesters). When used in this way "DCMIDCSV" or the name of one of its derivatives can be noted as the value of the SCHEME attribute of the HTML <META> element as shown in the following examples of qualified Dublin Core metadata:

```
<META NAME="DC.Contributor" SCHEME="DCMIDCSV"
CONTENT="name.given=Eric; name.family=Miller; employer=OCLC; height=170
cm">
<META NAME="DC.Format" SCHEME="DCMIDCSV" CONTENT="rows=200;
cols=450">
<META NAME="DC.Coverage.spatial" SCHEME="DCMIBOX"
CONTENT="name=Western Australia; northlimit=-13.5; southlimit=-35.5;
westlimit=112.5; eastlimit=129">
<META NAME="DC.Coverage.spatial" SCHEME="DCMIPOINT"
CONTENT="name=Bridgnorth, Shropshire, U.K.; east=372000; north=293000;
units=m; projection=U.K. National Grid">
<META NAME="DC.Date" SCHEME="DCMIPERIOD" CONTENT="name=Perth
International Arts Festival, 2000; start=2000-01-26; end=2000-02-20;">
```

Sample Code for parsing DCSV coded values

The following Perl program reads a DCSV coded string entered on stdin, and prints a formatted version of the structured result. This code is provided for demonstration purposes only and contains no error-checking.

```
#!/usr/local/bin/perl

use strict

print "Enter string to be parsed:\n";
```

```
my $string = join('',<STDIN>);

print "\nString to be parsed is [$string]\n";

# First escape % characters
$string =~ s/%/"%".unpack('C',"%")."%"/eg;

# Next change \ escaped characters to %d% where d is the character's ascii code
$string =~ s/\\(.)/"%".unpack('C',$1)."%"/eg;

print "\nEscaped string is [$string]\n";

# Now split the string into components
my @components = split(/;/, $string);

print "\nComponents:\n";
foreach $component (@components) {

        my ($label, $value) = split(/=/, $component, 2);

        # if there is no = copy contents of $label into $value and empty $label
        if (!$value) {
        $value = $label;
        $name = '';
        }

        # strip whitespace from name string
        $label =~ s/^\s*(\S+)\s*$/$1/;

        # convert % escaped characters back in label string
        $label =~ s/%(\d+)%/pack('C',$1)/eg;

        #convert % escaped characters back in value string
        $value =~s/%(\d+)%/pack('C',$1)/eg;

        print "Label [$label] has value [$value]\n";

}
```

Acknowledgments

John Kunze encouraged us to write up this proposal formally. Kim Covil wrote the perl code. Eric Miller nagged regarding the overlap with XML. Steve Tolkin convinced us to switch to =.

References

[**DCMI**]
Dublin Core Metadata Initiative, OCLC, Dublin Ohio.
http://dublincore.org/

[**HTML4**]
Dave Raggett, Arnaud Le Hors, Ian Jacobs, 1999, *HTML 4.01 Specification*
http://www.w3.org/TR/html4/

[Profiles]
DCMI Box - specification of the spatial limits of a place, and methods for encoding this in a text string
http://dublincore.org/documents/dcmi-box/

DCMI Point - a point location in space, and methods for encoding this in a text string
http://dublincore.org/documents/dcmi-point/

DCMI Period - specification of the limits of a time interval, and methods for encoding this in a text string http://dublincore.org/documents/dcmi-period/

[**Q-DC-HTML**]
S. Cox, 2000, *Recording qualified Dublin Core metadata in HTML*
http://dublincore.org/documents/dcq-html/

[**RDF-in-HTML**]
This uses the most compact form of XML-RDF [RDF-syntax], in which all the data occurs as attribute values. In this form several important capabilities are not available, such as multiple (repeated) values. For an example, see Figure 5 in S.J.D. Cox and K.D. Covil, *"A web-based geological information system using metadata"*, Proc. 3rd IEEE META-DATA Conference, http://computer.org/conferen/proceed/meta/1999/papers/7/cox_covil.html

[URI]
T. Berners-Lee, R. Fielding, L Masinter, 1998 Uniform Resource Identifiers (URI): Generic Syntax RFC2396
http://www.ietf.org/rfc/rfc2396.txt

T. Berners-Lee, L. Masinter, and M. McCahill, 1994 Uniform Resource Locators, RFC1738
http://www.ietf.org/rfc/rfc1738.txt

T. Berners-Lee, 1994 Universal Resource Identifiers in WWW: A Unifying Syntax for the Expression of Names and Addresses of Objects on the Network as used in the World-Wide Web, RFC1630
http://www.ietf.org/rfc/rfc1630.txt

[**vCard**]
F. Dawson, T. Howes, vCard MIME Directory Profile RFC2426
http://www.ietf.org/rfc/rfc2426.txt.

[**W3C-DTF**]
M. Wolf, C. Wicksteed, 1997, Date and Time Formats

http://www.w3.org/TR/NOTE-datetime

[XHTML]
Steven Pemberton and many others, 1999 XHTML 1.0: The Extensible HyperText Markup Language
http://www.w3.org/TR/xhtml1/
See also Dave Raggett, HyperText Markup Language Activity Statement
http://www.w3.org/MarkUp/Activity.html

[XML]
Extensible Markup Language
http://www.w3.org/XML/

Title:	# A Syntax for Writing a List of Labelled Values in a Text String
Creator:	Simon Cox
Creator:	Renato Iannella
Date Issued:	1999-04-30
Identifier:	http://dublincore.org/documents/1999/04/30/labelled-values-syntax/
Replaces:	Not Applicable
Is Replaced By:	Not Applicable
Latest version:	http://dublincore.org/documents/labelled-values-syntax/
Status of document:	This is a DCMI Note.
Description of document:	A method for recording lists of labelled values in a text string, called Dublin Core Structured Values, with the label DCSV, is described. The notation is intended for structured information within attribute value strings in markup-languages such as HTML and XML. This is likely to be useful in recording complex element values in metadata systems based on the qualified Dublin Core model.

1. Introduction

Element *attributes* in markup languages, such as HTML [HTML4] and XML [XML], provide an alternative position to the element *content* for recording data. For some "empty" elements - such as the `<IMG >`and `<META >`elements in HTML - attributes are the only place to hold data. In other cases there may be good reasons to prefer element attributes to element content for data. For example, fragments of XML can be included in the `<HEAD>`of a HTML document, and will be safely ignored by most client software (eg browsers) provided the elements have no content. This syntax trick can be used to embed XML-RDF encoded data safely in current versions of HTML [RDF-in-HTML].

Future versions of HTML are expected to overcome these limitations by allowing general XML documents to be included [XHTML]. Nevertheless, there is strong interest in using HTML `<META >`elements to record data with more structure than normally implied by a plain-text string, in particular to record metadata according to the qualified Dublin Core model [Q-DC-HTML].

However, the use of element attributes for storing data has a number of technical limitations:

1. attributes may occur no more than once
2. the data must consist of a text-string including no double-quotes (")

These features mean that there is no built-in way to indicate the structure of the data.

Nevertheless, in certain applications, it is highly desirable to be able to encode structured values within a plain-text string. Some generic methods are in common use. Inheriting conventions from natural languages, commas (,) and semi-colons (;) are frequently used as list separators. Similarly, comma-separated-values (CSV) and tab-separated-values (TSV) are common export formats from spreadsheet and database software. Dots (.) and dashes (-) are sometimes used to imply hierarchies, particularly in thesaurus applications.

A number of specific encoding schemes use punctuation characters within the text string to indicate structure. For example, colons (:) terminate protocol labels, and double slashes (//) act as separators for identifiers coded as URIs [URI]. Colons (:) separate specified labels from values within a field, and semi-colons (;) separate fields within a personal description according to vCard [vCard]. Hyphens are one of the many characters used to separate fields in a date according with ISO8601 [ISO8601]. For some schemes - vCard and ISO8601, for example - the punctuation indicates a very formal structure to the value, and is expected to be parsed automatically.

Our intention in this note is to define a generic, self-describing data-structuring method for text-strings, to be used when no other suitable scheme is available. This is based on methods used and found successful elsewhere, vCard in particular, but is more generalised than the preceding standards.

2. Structured Values - the DCSV scheme

To allow the recording of generic **Structured Values**, we introduce the Dublin Core Structured Values (**DCSV**) scheme.

We distinguish between two types of substring - *labels* and *values*, where a label is the name of the type of a value, and a value is the data itself. Furthermore, we allow a complete value to be disaggregated into set of *components*, each of which has its own label and value. A value that is comprised of components in this way is called a *structured value*.

Punctuation characters are used in recording a structured value as follows:

- colons (:) separate plain-text labels of structured value-components from the values themselves
- semi-colons (;) separate (optionally labelled) value-components within a list
- dots (.) indicate hierachical structure in labels, if required.

The labels and the component values themselves each consist of a text-string. The intention is that the label will be a word or code corresponding to the name of the value-component. Labels may be absent, in which case the entire sub-string delimited by semi-colons (;) or the end of the string comprise the component value.

The following patterns show how structured values may be recorded in strings using DCSV:

```
"u1; u2; u3"
"cA:v1"
"cA:v1; cB.part1:v2; cB.part2:v3"
"cA:v1; u2; u3"
```

where `u1`, `u2`and `u3`are unlabelled components, `cA`and `cB`are the labels of Structured Value components, `part1`and `part2`are sub-components of `cB`, and `v1`, `v2`and `v3`are values of the components.

The use of the specific punctuation characters in DCSV coded values means that these characters cannot be used directly within strings which comprise the content (either labels or values) of the components. For DCSV, therefore, when a period, full-stop or dot (.) a colon (:), or a semi-colon (;) is required within the value, the characters are escaped using a backslash, appearing as **\. \: \;**, and the backslash itself is escaped similarly \\ . This method of escaping special characters largely preserves readability and the ability to enter DCSV coded metadata values easily using a text-editor if required. Software written to process DCSV coded values must make the necessary substitutions.

Note that, the double-quote (") character is a generic case that cannot be used directly within HTML or

XML element attributes.

3. Parsing DCSV

A simple method can be used to parse metadata values recorded according to the DCSV scheme. For a single value recorded using the DCSV scheme:

1. split the text-string into a list of substrings on any unescaped semi-colons (;);
   if no semi-colon is present, there is a single substring
2. split each substring into its (label,value) on any unescaped colons (:);
   if no colon is present, the label is empty
3. within each value replace the escaped characters with the actual character required.

A short Perl program which performs this parsing operation is included at the end of this note.

4. Examples

```
"name.given:Renato; name.family:Iannella; employer:DSTC;
Contact:Level 7, Gehrmann Labs, The University of Queensland,
Qld\. 4072, Australia"
"rows:200; cols:450"
```

The DCSV scheme adds most of the components required for the representation of the qualified DC model in HTML [Q-DC-HTML][Q-DC-RDF], while remaining fully compatible with the HTML-4 [HTML] standard. It thus supports a recording method for qualified Dublin Core, compatible with tools which rely on HTML (browsers, metadata harvesters), but with a clear route for migrating relatively rich information into fully structured notations when appropriate. In this context, DCSV is noted as the value of the

```
SCHEME
```

attribute of the HTML

```
<META >
```

element as shown in the foloowing examples:

```
<META NAME="DC.Creator" SCHEME="DCSV"
```

```
         CONTENT="name.given:Simon; name.family:Cox;
employer:CSIRO; height:177 cm">
<META NAME="DC.Language" SCHEME="RFC1766" CONTENT="en-AU">
<META NAME="DC.Contributor" SCHEME="vCard" CONTENT="fn:Simon
Cox; org:CSIRO">
<META NAME="DC.Date" SCHEME="ISO8601" CONTENT="1999-04-30">
<META NAME="DC.Relation" SCHEME="URL"
CONTENT="http://www.foo.bar/explication.html">
<META NAME="DC.Format.media" SCHEME="IMT" CONTENT="image/gif">
<META NAME="DC.Format.size" CONTENT="14 kB">
<META NAME="DC.Format.size" SCHEME="DCSV" CONTENT="rows:200;
cols:450">
```

5. Sample Code for parsing DCSV coded values

The following Perl program reads a DCSV coded string entered on stdin, and prints a formatted version of the structured result. This code is provided for demonstration purposes only and contains no error-checking.

```
#!/usr/local/bin/perl

print "Enter string to be parsed:\n";

my $string = join('',<STDIN>);

print "\nString to be parsed is [$string]\n";

# First escape % characters
$string =~ s/%/"%".unpack('C',"%")."%"/eg;

# Next change \ escaped characters to %d% where d is the
character's ascii code
$string =~ s/\\(.)/"%".unpack('C',$1)."%"/eg;

print "\nEscaped string is [$string]\n";

# Now split the string into components
my @components = split(/;/, $string);

print "\nComponents:\n";
foreach $component (@components) {
    my ($name, $value) = split(/:/, $component, 2);
    # if there is no : $value is empty so copy $name
into $value and empty $name
    if (!$value) {
        $value = $name;
        $name = '';
    }

    # strip whitespace from name string
    $name =~ s/^\s*(\S+)\s*$/$1/;

    # convert % escaped characters back in value string
    $value =~ s/%(\d+)%/pack('C',$1)/eg;

    print "Name [$name] has value [$value]\n";
}
```

6. Acknowledgments

John Kunze encouraged us to write up this proposal formally. Kim Covil wrote the perl code.

7. References

[HTML4]
Dave Raggett, Arnaud Le Hors, Ian Jacobs, 1998, *HTML 4.0 Specification* http://www.w3.org/TR/REC-html40/

[ISO8601]
M. Wolf and C. Wicksteed, 1997, *Date and Time Formats*, http://www.w3.org/TR/NOTE-datetime

[Q-DC-HTML]
S. Cox 1999 *Recording qualified Dublin Core metadata in HTML* http://www.agcrc.csiro.au/projects/3018CO/metadata/qdchtml/NOTE-QDCHTML-19991103.html

[Q-DC-RDF]
E. Miller, P. Miller, D. Brickley, 1999. *Guidance on expressing the Dublin Core within the Resource Description Framework (RDF)* http://www.ukoln.ac.uk/interop-focus/activities/dc/

datamodel/

[RDF-in-HTML]

This uses the most compact form of XML-RDF [RDF-syntax], in which all the data occurs as attribute values. In this form several important capabilities are not available, such as multiple (repeated) values. For an example, see Figure 5 in S.J.D. Cox and K.D. Covil, "A web-based geological information system using metadata", *Proc. 3rd IEEE META-DATA Conference*, http://computer.org/conferen/proceed/meta/1999/papers/7/cox_covil.html

[URI]

T. Berners-Lee, R. Fielding, L Masinter, 1998 *Uniform Resource Identifiers (URI): Generic Syntax* RFC2396 http://info.internet.isi.edu/in-notes/rfc/files/rfc2396.txt

T. Berners-Lee, L. Masinter, and M. McCahill, 1994 *Uniform Resource Locators*, RFC1738 http://info.internet.isi.edu/in-notes/rfc/files/rfc1738.txt.

T. Berners-Lee, 1994 *Universal Resource Identifiers in WWW: A Unifying Syntax for the Expression of Names and Addresses of Objects on the Network as used in the World-Wide Web*, RFC1630 http://info.internet.isi.edu/in-notes/rfc/files/rfc1630.txt.

[vCard]

F. Dawson, T. Howes, 1998 *vCard MIME Directory Profile* RFC2426 http://info.internet.isi.edu/in-notes/rfc/files/rfc2426.txt

[XHTML]

Steven Pemberton and many others, 1999 *XHTML 1.0: The Extensible HyperText Markup Language* http://www.w3.org/TR/WD-html-in-xml/

See also Dave Raggett, *HyperText Markup Language Activity Statement* http://www.w3.org/MarkUp/Activity.html

[XML]

Extensible Markup Language http://www.w3.org/XML/

Metadata associated with this resource: http://dublincore.org/documents/1999/04/30/labelled-values-syntax/index.shtml.rdf

Title:	**DCMI Point Encoding Scheme: a point location in space, and methods for encoding this in a text string**
Creator:	Simon Cox
Date Issued:	2000-07-28
Identifier:	http://dublincore.org/documents/2000/07/28/dcmi-point/
Replaces:	http://dublincore.org/documents/2000/07/11/dcmi-point/
Is Replaced By:	Not Applicable
Latest version:	http://dublincore.org/documents/dcmi-point/
Status of document:	This is a DCMI Recommendation.
Description of document:	We introduce DCMI Point for identifying a point in space using its geographic coordinates. Components of the value correspond to the location coordinates in north and east directions, plus optionally elevation, and also allow the coordinate system and units to be specified, and a name if desired. We describe a method for encoding DCMI Point in a text-string, as a profile of DCSV. This notation is intended for recording the value of the DCES element **Coverage**, particularly when using HTML meta elements. We also show an alternative encoding for DCMI Point using XML.
NOTICE TO IMPLEMENTORS:	The syntax examples included in this document are provisional, and are currently under review as part of the DCMI work on recommending coordinated syntax recommendations for HTML, XML, and RDF. These recommendations and minor editorial changes in this document can be expected to take place in the near future. **Note that the use of "=" as a separator in the DCMI-DCSV encoding is a change from earlier versions of this specification which used ":" in the same position.**

Table of Contents

---

1. Introduction

Several methods are available to indicate a *place*. These include, but are not limited to:

- a **name**, normally defined in an identifiable enumeration such as a gazetteer or list of jurisdictional localities
- a unique **geocode**, such as a postal code
- the coordinates of a **point**, using geographic values or some well-defined projection and units
- a set of arcs or faces describing the **polygon** or **polyhedron** comprising the perimeter of the place
- the **limits** of a regular shaped container which encompasses the place, typically a rectangular **box** in two or three dimensions, using geographic values or some well-defined projection and units

The Dublin Core Metadata Element Set [DCMES] includes an element, **Coverage**, the value of which may contain an identifier for a place. If a name or geocode is used then the scheme from which that is selected determines valid values. However, there are no simple, commonly used, notations for the identifiers which use coordinates. Here we define DCMI Point, an identifier which specifies the coordinates of the point location of a place, and describe methods for encoding DCMI Point, as a profile of DCSV [DCSV], and using a fragment of XML [XML]. If an identifier corresponding to an extensive *region* is required, then DCMI Box [BOX] is available for rectangular regions.

2. Identifying a place - the DCMI Point scheme

We identify a place using a point location, described using coordinates in an identified cartesian system. The point may correspond to some place within an extensive region, such as the areal or volumetric centroid, but we do not specify the nature of the relationship in this document.

We define the following components to describe the point:

Component	**Definition**	**Default**[1]
east	The value of the coordinate of the location measured in the east direction[2]	+/- INF[3]
north	The value of the coordinate of the location measured in the north direction[2]	+/- INF[3]
elevation	The value of the coordinate of the location measured in the vertical direction[2]	+/- INF[3]
units	The units applying to unlabelled numeric values of north, east	signed decimal degrees
zunits	The units applying to unlabelled numeric values of elevation	metres
projection	The name of the projection used with any parameters required, such as ellipsoid parameters, datum, standard parallels and meridians, zone, etc	geographic coordinates on Earth for north, east; height above mean-sea-level for elevation.
name	A name for the place[4]	-

1 All components are optional.

2 Values are expressed as a text-string representing a number. Units should be included using conventional (SI) notation, unless the relevant units *or* zunits *component is present. However, if units are given as part of any value, then for this component these override those given by* units *or* zunits*.*

3 If this component is absent then the value is undefined. Processors performing numeric comparisons are recommended to set values corresponding to maximally inclusive matching, i.e. the location is a line if one coordinate is missing, and a plane if two are missing.

4 In this context the name is non-normative. In the case of a conflict, the place identified by the coordinate values takes precedence. The name is provided for user convenience only.

3. Encoding DCMI Point

The components of a DCMI Point identifier have no meaning when disaggregated, since in any particular instance it is the complete set which acts as the *identifier*. Thus, use of DCMI Point to identify a place requires that the components are linked together. For systems in which data is encoded using a limited character set, this is conveniently accomplished by packaging the components into a single text-string. Various serialisation syntaxes are available, including DCSV [DCSV] and XML [XML].

In normal usage, the unadorned token "DCMI Point" should be taken to refer to the encoding using DCSV.

3.1 DCSV encoding

Writing DCMI Point using DCSV notation is straightforward, using the component names defined above. A DCMI Point value appears as follows:

```
east=v1; north=v2; elevation=v3; units=v4; zunits=v5; projection=v6;
name=v7
```

where `v1` - `v7` are values as defined in the table above.

All components are optional but may not be repeated, and the ordering is not significant.

3.2 XML encoding

DCMI Point may be written in XML. Given the flexibility of XML many alternative notations are possible. One form looks like this:

```
<Point projection="v6" name="v7">

  <east units="v4a">v1</east>

  <north units="v4b">v2</north>

  <elevation zunits="v5">v3</elevation>

</Point>


defined by the DTD fragment:


<!ELEMENT   Point     (east?,north?,elevation?)>
```

```
<!ATTLIST   Point

      projection  CDATA    "geographic, height relative to mean-sea-
level"

      name        CDATA    #IMPLIED >

<!ELEMENT   east       (#PCDATA)>

<!ATTLIST   east       units    CDATA "signed decimal degrees">

<!ELEMENT   north      (#PCDATA)>

<!ATTLIST   north      units    CDATA "signed decimal degrees">

<!ELEMENT   elevation  (#PCDATA)>

<!ATTLIST   elevation  zunits   CDATA "m">
```

The values here are equivalent to the values in the DCSV profile. Note that:

1. We have defined an XML *element* Point. Instances of this would occur within a complete XML *document*.
2. The content model for Point is a conventionally ordered (x,y,z) sequence of (optional) coordinate elements. This is a cleaner representation of the information required to specify the "point" structure than is possible in DCSV. All other components of DCMI Box occur as *attributes*
3. `units` and `zunits` are recorded in an XML *attribute*. Since these are associated directly with the local coordinate element, it is possible to express different components in different units if desired.

4. Examples

Perth, Western Australia:

```
name=Perth, W.A.; east=115.85717; north=-31.95301
```

```
<Point name="Perth, W.A.">

        <east>115.85717</east>
        <north>-31.95301</north>

</Point>
```

Bridgnorth, Shropshire, U.K.:

```
east=372000; north=293000; units=m; projection=U.K. National Grid
```

```
<Point projection="U.K. National Grid" name="Bridgnorth">

        <east units="m">372000</east>

        <north units="m">293000</north>

</Point>
```

The Greenwich Meridian:

```
east=0;
```

```
<Point>

        <east>0</east>

</Point>
```

The highest point in Australia, illustrating the use of 3-D coordinates (and how flat Australia is):

```
east=148.26218; north=-36.45746; elevation=2228; name=Mt. Kosciusko
```

```
<Point name="Mt. Kosciusko">

        <east>148.26218</east>

        <north>-36.45746</north>

        <elevation>2228</elevation>
```

```
</Point>
```

5. References

[BOX]
S. Cox, 2000. DCMI Box Encoding Scheme- specification of the spatial limits of a place, and methods for encoding this in a text string http://dublincore.org/documents/dcmi-box/

[DCMES]
1999. Dublin Core Metadata Element Set, Version 1.1: Reference Description http://dublincore.org/documents/dces/

[DCMI]
Dublin Core Metadata Initiative, OCLC, Dublin Ohio.
http://dublincore.org/

[DCSV]
S. Cox, R. Iannella, 2000. A syntax for writing a list of labelled values in a text string http://dublincore.org/documents/dcmi-dcsv/

[XML]
Extensible Markup Language
http://www.w3.org/XML/

Metadata associated with this resource: http://dublincore.org/documents/2000/07/28/dcmi-point/index.shtml.rdf

Title:	**DCMI Period Encoding Scheme: specification of the limits of a time interval, and methods for encoding this in a text string**
Creator:	Simon Cox
Date Issued:	2000-07-28
Identifier:	http://dublincore.org/documents/2000/07/28/dcmi-period/
Replaces:	http://dublincore.org/documents/2000/07/11/dcmi-period/
Is Replaced By:	Not Applicable
Latest version:	http://dublincore.org/documents/dcmi-period/
Status of document:	This is a DCMI Recommendation.
Description of document:	We introduce DCMI Period for identifying a single time interval using its limits. Components of the value correspond to the start and end of the interval, either of which may be ommitted in the case of a single-ended interval. We describe a method for encoding DCMI Period in a text-string, as a profile of DCSV. This notation is intended for recording the value of the DCMES elements **Coverage** and **Date**, particularly when using HTML meta elements. We also show an alternative encoding for DCMI Period using XML.
NOTICE TO IMPLEMENTORS:	The syntax examples included in this document are provisional, and are currently under review as part of the DCMI work on recommending co-ordinated syntax recommendations for HTML, XML, and RDF. These recommendations and minor editorial changes in this document can be expected to take place in the near future. **Note that the use of "=" as a separator in the DCMI-DCSV encoding is a change from earlier versions of this specification which used ":" in the same position.**

Table of Contents

1. Introduction

Several methods are available to indicate a time interval. These include, but are not limited to:

- a **name**, normally defined in an enumeration such as a list of artistic, cultural, historical, archaeological, geological or cosmological eras or periods, a list of ruler's names, families or dynasties, etc.
- the **limits** of the interval, using either numeric or named values, the latter optionally including qualifiers such as **start of, end of, middle of**, etc.

The Dublin Core Metadata Element Set [DCMES] includes two elements, **Coverage** and **Date**, the values of which may contain an identifier for a time interval.

If a name is used then the scheme from which it is selected determines the meaning.

The W3C profile of the ISO8601 standard for dates and times [W3C-DTF] is generally useful for identifying time instants, and also includes a method for specifying complete intervals by joining two instants with a "/" character. However, there is a need for a richer model for use in some cases, for three reasons:

1. open intervals, i.e. those with only a start or an end, are not included in the specification
2. the syntax cannot be adapted for use with other spatio-temporal dimensions, which would be desirable for consistency of use with the DCMES Coverage element
3. the identification of the start and end of the interval is implicit - relying in the position within a string - and therefore error-prone, rather than explicitly labelled.

The W3C recommendation *Mathematical Markup Language* [MathML] includes an XML binding for intervals which permits quite general intervals to be described, using both numeric and non-numeric bounds. This is a syntax-specific notation which, in order to be consistent with other parts of the bigger specification of which it is a part, includes features which are relatively obscure for the simple goal here. It is also unclear how to specify this extract from a much larger standard, and since it is presented as an XML notation, other serialisations would need to be specified separately in any case.

Here we define DCMI Period, an identifier which uses a simple model to specify the limits of a time interval, and describe methods for encoding DCMI Period, as a profile of DCSV [DCSV], and using a fragment of XML [XML]. DCMI Period has been designed to be similar to DCMI Box [BOX] used for identifying a place, and thus allows consistent encoding of spatio-temporal information in the DCMES element **Coverage**, as well as consistency between **Coverage** and **Date**. The components of DCMI Period re-use the W3C-DTF syntax where possible.

DCMI Period identifies a single time *interval*. If an identifier corresponding to a time instant is required, then W3C-DTF [W3C-DTF] is available. For multiple disjoint intervals, repeated instances of DCMI Period may be used. DCMI Period is unsuited for identification of recurring and periodic time intervals.

2. Identifying a time interval - the DCMI Period scheme

We identify a time interval by specifying the start and end of the interval.

We define the following components to describe the interval:

Component	**Definition**	**Default**[1]
start	The instant corresponding to the commencement of the time interval	-INF[2]
end	The instant corresponding to the termination of the time interval	INF[2]
scheme	The encoding used for the representation of the time-instants in the start and end components[3]	W3C-DTF
name	A name for the time interval[4]	-

1 *All components are optional.*

2 *If either start or end is absent, then this implies an interval unbounded on that side. Thus, a DCMI Period with a single component start="2000-01-26" would identify the interval starting at the beginning of Australia Day in the year 2000 C.E. and continuing from that time.*

3 *If a non-numeric encoding is used then matching is maximally inclusive: i.e. if a start component is expressed as a named era then the interval being identified starts at the beginning of the era, and conversely for an end component the interval ends at the end of the named era.*

4 *In this context the name is non-normative. In the case of a conflict, the interval identified by the start and end values takes precedence. The name is provided for user convenience only.*

3. Encoding DCMI Period

The components of a DCMI Period identifier have no meaning when disaggregated, since in any particular instance it is the complete set which acts as the *identifier*. Thus, use of DCMI Period to identify a time interval requires that the components are linked together. For systems in which data is encoded using a limited character set, this is conveniently accomplished by packaging the components into a single text-string. Various serialisation syntaxes are available, including DCSV [DCSV] and XML [XML].

In normal usage, the unadorned token "DCMI Period" should be taken to refer to the encoding using DCSV.

3.1 DCSV encoding

Writing DCMI Period using DCSV notation is straightforward, using the component names defined above. A DCMI Period value appears as follows:

```
start=v1; end=v2; scheme=v3; name=v4;


    where v1 - v4 are values as defined in the table above.
```

All components are optional but may not be repeated. The ordering is not significant.

3.2 XML encoding

DCMI Period may be written in XML. Given the flexibility of XML many alternative notations are possible. One form looks like this:

```
<Period name="v4">

  <start scheme="v3a">v1</start>

  <end scheme="v3b">v2</end>

</Period>
```

defined by the DTD fragment:

```
<!ELEMENT   Period       (start?,end?)>

<!ATTLIST   Period

      name        CDATA #IMPLIED >

<!ELEMENT   start  (#PCDATA)>

<!ATTLIST   start  scheme    CDATA "W3C-DTF">

<!ELEMENT   end    (#PCDATA)>

<!ATTLIST   end    scheme    CDATA "W3C-DTF">
```

The values here are equivalent to the values in the DCSV profile. Note that:

1. We have defined an XML *element* Period. Instances of this would occur within a complete XML *document*.
2. The content model for Period is an ordered pair of elements (start,end), either of which may be omitted. All other components of Period occur as *attributes*
3. The scheme used to represent the component time-instants is recorded in an XML *attribute*. Since these are associated directly with either the start or end element, it is possible to express different components using different notations if desired.

4. Examples

The Great Depression:

name=The Great Depression; start=1929; end=1939;

```
<Period name="The Great Depression">

        <start>1929</start>

        <end>1939</end>

</Period>
```

Perth International Arts Festival, 2000:

```
name=Perth International Arts Festival, 2000; start=2000-01-26;
end=2000-02-20;
```

```
<Period name="Perth International Arts Festival 2000">

        <start>2000-01-26</start>

        <end>2000-02-20</end>

</Period>
```

1999 AFL Grand Final:

```
start=1999-09-25T14:20+10:00; end=1999-09-25T16:40+10:00; scheme=W3C-
DTF;
```

```
<Period name="1999 AFL Grand Final">

        <start scheme="W3C-DTF">1999-09-25T14:20+10:00</start>

        <end scheme="W3C-DTF">1999-09-25T16:40+10:00</end>

</Period>
```

The Phanerozoic Eon:

```
start=Cambrian period; scheme=Geological timescale; name=Phanerozoic
Eon;
```

```
<Period name="Phanerozoic Eon">

        <start scheme="Geological timescale">Cambrian period</start>

</Period>
```

---

5. References

[BOX]
S. Cox, 2000, DCMI Box - specification of the spatial limits of a place, and methods for encoding this in a text string http://dublincore.org/documents/dcmi-box/
[DCMES]
1999. Dublin Core Metadata Element Set, Version 1.1: Reference Description
http://dublincore.org/documents/dces/
[DCMI]
Dublin Core Metadata Initiative, OCLC, Dublin Ohio. http://dublincore.org/
[DCSV]
S. Cox, R. Iannella, 2000. A syntax for writing a list of labelled values in a text string
http://dublincore.org/documents/dcmi-dcsv/
[MathML]
Mathematical Markup Language (MathML) 1.01 Specification
http://www.w3.org/TR/REC-MathML/
[W3C-DTF]
M. Wolf, C. Wicksteed, 1997, Date and Time Formats
http://www.w3.org/TR/NOTE-datetime
[XML]
Extensible Markup Language
http://www.w3.org/XML/

Title:	**DCMI Box Encoding Scheme: specification of the spatial limits of a place, and methods for encoding this in a text string**
Creator:	Simon Cox
Date Issued:	2000-07-28
Identifier:	http://dublincore.org/documents/2000/07/28/dcmi-box/
Replaces:	http://dublincore.org/documents/2000/07/11/dcmi-box/
Is Replaced By:	Not Applicable
Latest version:	http://dublincore.org/documents/dcmi-box/
Status of document:	This is a DCMI Recommendation.
Description of document:	The DCMI Box encoding scheme is a method for identifying a region of space using its geographic limits. Components of the value correspond to the bounding coordinates in north, south, east and west directions, plus optionally up and down, and also allow the coordinate system and units to be specified, and a name if desired. A method for encoding DCMI Box in a text-string, as a profile of DCSV is described. This notation is intended for recording the value of the DCMES element **Coverage**, particularly when using HTML meta elements. An alternative encoding for DCMI Box using XML is also shown.
NOTICE TO IMPLEMENTORS:	The syntax examples included in this document are provisional, and are currently under review as part of the DCMI work on recommending coordinated syntax recommendations for HTML, XML, and RDF. These recommendations and minor editorial changes in this document can be expected to take place in the near future. **Note that the use of "=" as a separator in the DCMI-DCSV encoding is a change from earlier versions of this specification which used ":" in the same position.**

Table of Contents

---

1. Introduction

Several methods are available to indicate a place. These include, but are not limited to:

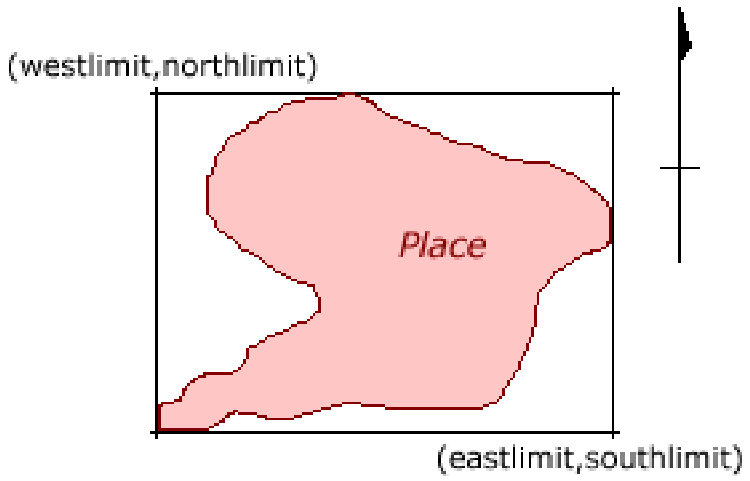
- a **name**, normally defined in an identifiable enumeration such as a gazetteer or list of jurisdictional localities
- a unique **geocode**, such as a postal code
- the coordinates of a **point**, using geographic values or some well-defined projection and units
- a set of arcs or faces describing the **polygon** or **polyhedron** comprising the perimeter of the place
- the **limits** of a regular shaped container which encompasses the place, typically a rectangular **box** in two or three dimensions, using geographic values or some well-defined projection and units

The Dublin Core Metadata Element Set [DCMES] includes an element, **Coverage**, the value of which may contain an identifier for a place. If a name or geocode is used then the scheme from which that is selected determines valid values. However, there are no simple, commonly used, notations for the identifiers which use coordinates. Here we define DCMI Box, an identifier which specifies the geographic limits of a place, and describe methods for encoding DCMI Box, as a profile of DCSV [DCSV], and using a fragment of XML [XML].

In the simplest usage DCMI Box approximates the extent of a place using a container with a regular shape. For a more precise representation of an irregular shape it is possible to use the approach of "tiling" the place with a set of simple regions defined using DCMI Box. Alternatively, another notation describing a polygon or polyhedron may be used. If an identifier corresponding to a *point* is required, then DCMI Point [POINT] is available.

2. Identifying a place - the DCMI Box scheme

We identify a place by considering the minimal rectangular box which fully encloses the place, whose faces are aligned parallel with the axes of an identified cartesian coordinate system [Figure].



We define the following components to describe the box:

Component	**Definition**	**Default**[1]
northlimit	The value of the constant coordinate for the northernmost face or edge[2]	INF[3]
eastlimit	The value of the constant coordinate for the easternmost face or edge[2]	INF[3]
southlimit	The value of the constant coordinate for the southernmost face or edge[2]	-INF[3]
westlimit	The value of the constant coordinate for the westernmost face or edge[2]	-INF[3]
uplimit	The value of the constant coordinate for the uppermost face or edge[2]	INF[3]
downlimit	The value of the constant coordinate for the lowermost face or edge[2]	-INF[3]

units	The units applying to unlabelled numeric values of northlimit, eastlimit, southlimit, westlimit	signed decimal degrees
zunits	The units applying to unlabelled numeric values of uplimit, downlimit	metres
projection	The name of the projection used with any parameters required, such as ellipsoid parameters, datum, standard parallels and meridians, zone, etc	geographic coordinates on Earth for northlimit, eastlimit, southlimit, westlimit; height above mean-sea-level for uplimit, downlimit.
name	A name for the place[4]	-

1*All components are optional. If any *limit component is absent, then this implies an interval unbounded on that side. Thus, a DCMI Box with a single component northlimit="0" would identify the entire southern hemisphere.*

2*Values are expressed as a text-string representing a number. Units should be included using conventional (SI) notation, unless the relevant units or zunits component is present. However, if units are given as part of any value, then for this component these override those given by units or zunits.*

3*If this component is absent then the value is undefined. Processors performing numeric comparisons are recommended to set values corresponding to maximally inclusive matching.*

4*In this context the name is non-normative. In the case of a conflict, the place identified by the coordinate values takes precedence. The name is provided for user convenience only.*

3. Encoding DCMI Box

The components of a DCMI Box identifier have no meaning when disaggregated, since in any particular instance it is the complete set which acts as the *identifier*. Thus, use of DCMI Box to identify a place requires that the components are linked together. For systems in which data is encoded using a limited character set, this is conveniently accomplished by packaging the components into a single text-string. Various serialisation syntaxes are available, including DCSV [DCSV] and XML [XML].

In normal usage, the unadorned token "DCMI Box" should be taken to refer to the encoding using DCSV.

3.1 DCSV encoding

Writing DCMI Box using DCSV notation is straightforward, using the component names defined above. A DCMI Box value appears as follows:

```
northlimit=v1; eastlimit=v2; southlimit=v3; westlimit=v4;
uplimit=v5;
downlimit=v6; units=v7; zunits=v8; projection=v9; name=v10
```

where `v1` - `v10` are values as defined in the table above.

All components are optional but may not be repeated, and the ordering is not significant.

3.2 XML encoding

DCMI Box may be written in XML. Given the flexibility of XML many alternative notations are possible. One form looks like this:

```
<Box projection="v9" name="v10">

<northlimit units="v7a">v1</northlimit>

<eastlimit units="v7b">v2</eastlimit>

<southlimit units="v7c">v3</southlimit>

<westlimit units="v7d">v4</westlimit>

<uplimit zunits="v8a">v3</uplimit>

<downlimit zunits="v8b">v4</downlimit>

</Box>
```

defined by the DTD fragment:

```
<!ELEMENT   Box  (northlimit?,eastlimit?,southlimit?,westlimit?,
uplimit?,downlimit?)>

<!ATTLIST   Box

      projection  CDATA "geographic, height relative to mean-sea-
```

```
level"

      name        CDATA #IMPLIED >

<!ELEMENT   northlimit  (#PCDATA)>

<!ATTLIST   northlimit  units    CDATA "signed decimal degrees">

<!ELEMENT   eastlimit   (#PCDATA)>

<!ATTLIST   eastlimit   units    CDATA "signed decimal degrees">

<!ELEMENT   southlimit  (#PCDATA)>

<!ATTLIST   southlimit  units    CDATA "signed decimal degrees">

<!ELEMENT   westlimit   (#PCDATA)>

<!ATTLIST   westlimit   units    CDATA "signed decimal degrees">

<!ELEMENT   uplimit     (#PCDATA)>

<!ATTLIST   uplimit     zunits   CDATA "m">

<!ELEMENT   downlimit   (#PCDATA)>

<!ATTLIST   downlimit   zunits   CDATA "m">
```

The values here are equivalent to the values in the DCSV profile. Note that:

1. We have defined an XML *element* Box. Instances of this would occur within a complete XML *document*.
2. The content model for Box is a (clockwise) sequence of (optional) *limit elements. This is a cleaner representation of the information required to specify the "box" structure than is possible in DCSV. All other components of Box occur as *attributes*
3. units and zunits are recorded in an XML *attribute*. Since these are associated directly with the local *limit element, it is possible to express different components in different units if desired.

4. Examples

Western Australia:

```
name=Western Australia; northlimit=-13.5; southlimit=-35.5;

westlimit=112.5; eastlimit=129



<Box name="Western Australia">

<northlimit>-13.5</northlimit>

<eastlimit>129</eastlimit>

<southlimit>-35.5</southlimit>

<westlimit>112.5</westlimit>

</Box>
```

Lake Jindabyne:

```
northlimit=5980000; westlimit=644000; eastlimit=647000;
southlimit=5966000;

units=m; projection=UTM zone 55 south

<Box projection="UTM zone 55 south" name="Lake Jindabyne">

<northlimit units="m">5980000</northlimit>

<eastlimit units="m">647000</eastlimit>

<southlimit units="m">5966000</southlimit>

<westlimit units="m">644000</westlimit>

</Box>
```

The Western Hemisphere:

```
westlimit=180; eastlimit=0
```

```
<Box>
<eastlimit>0</eastlimit>
<westlimit>180</westlimit>
</Box>
```

The Tropics:

```
northlimit=23.5; southlimit=-23.5
```

```
<Box>
<northlimit>23.5</northlimit>
<southlimit>-23.5</southlimit>
</Box>
```

A mine, illustrating the use of 3-D coordinates:

```
northlimit=-21.3; southlimit=-21.4; westlimit=139.8; eastlimit=139.9;
uplimit=400; downlimit=-100; name=Duchess copper mine
```

```
<Box name="Duchess copper mine">
<northlimit>-21.3</northlimit>
<eastlimit>139.9</eastlimit>
<southlimit>-21.4</southlimit>
<westlimit>139.8</westlimit>
```

```
<uplimit>400</uplimit>

<downlimit>-100</downlimit>

</Box>
```

---

5. References

[DCMES]
1999. Dublin Core Metadata Element Set, Version 1.1: Reference Description
http://dublincore.org/documents/dces/
[DCMI]
Dublin Core Metadata Initiative, OCLC, Dublin Ohio. http://dublincore.org
[POINT]
S. Cox, 2000. DCMI Point - a point location in space and methods for encoding this in a text string,
http://dublincore.org/documents/dcmi-point/
[DCSV]
S. Cox, R. Iannella, 2000. A syntax for writing a list of labelled values in a text string
http://dublincore.org/documents/dcmi-dcsv/
[XML]
Extensible Markup Language http://www.w3.org/XML/

ISSS/WS-MMI-DC/076

Dublin Core Application Profile Guidelines

Final draft CWA

September 2003

Contributor:	CEN/ISSS MMI-DC Project Team on Metadata – Dublin Core - Application Profiles
Contributor:	Thomas Baker
Contributor:	Makx Dekkers
Contributor:	Thomas Fischer
Contributor:	Rachel Heery
Modified:	2003-09-01
Description:	These guidelines specify the structure and content of Dublin Core Application Profiles, a form for documenting which terms a given application uses in its metadata, with what extensions or adaptations, and specifying how those terms relate both to formal standards such as Dublin Core as well as to less formally defined element sets and vocabularies. This draft will be submitted to the Metadata for Multimedia Information Dublin Core Workshop of the European Committee for Standardization (CEN) in Brussels for approval as a CEN Workshop Agreement in September 2003.

Contents

Foreword

Normally the Foreword is drafted by the CEN/ISSS secretariat.

Introduction

A Dublin Core Application Profile (DCAP) is a declaration specifying which metadata terms an organization, information provider, or user community uses in its metadata. By definition, a DCAP identifies the source of metadata terms used – whether they have been defined in formally maintained standards such as Dublin Core, in less formally defined element sets and vocabularies, or by the creator of the DCAP itself for local use in an application. Optionally, a DCAP may provide additional documentation on how the terms are constrained, encoded, or interpreted for application-specific purposes.

A DCAP is designed to promote interoperability within the constraints of the Dublin Core model and to encourage harmonization of usage and convergence on “emerging semantics" around its edges. Historically, application profiles have emerged out of a need to share local domain- or application-specific refinements of or extensions to Dublin Core within particular application communities without necessarily seeking an extension of the core standard maintained by the Dublin Core Metadata Initiative (DCMI). Application profiles document how implementers use elements from Dublin Core along with elements from other vocabularies, customizing standard definitions and usage guidelines for local requirements [HEERY].

In practice, application profiles are created for a wide range of purposes: to document the semantics and constraints used for a set of metadata records (“instance metadata”); to help communities of implementers harmonize metadata practice among themselves; to identify emerging semantics as possible candidates for formal standardization; as guides for semantic crosswalks and format conversions; as specifications for formal encoding structures such as Document Type Definitions (DTDs); for interpreting or presenting legacy or proprietary metadata in terms of widely-understood standards; or for documenting the rules and criteria according to which a set of metadata was created. Application profiles often represent “work in progress”, providing foci for ongoing efforts to incrementally improve and clarify a body of shared metadata semantics within a particular user community.

In the absence of guidelines, creators of application profiles have hitherto invented a wide range of presentation formats. The present document distils the salient features of many existing profiles into a format that is as concise and simple as possible, yet as precise and detailed as is sometimes necessary to support the various uses identified above.

Semantic interoperability – the ultimate purpose of documents such as DCAPs – is a longer-term goal to be pursued as metadata vocabularies and related enabling technologies mature over time. In their current form, DCAPs are designed to document metadata usage in a normalized form that will lend itself to translation into common models, such as RDF, that can be processed by machines to automate such interoperability.

Machine-understandable representations will achieve this goal to the extent that metadata terms can be referenced using stable, well-documented identifiers. As discussed below, the practice of identifying metadata terms with Uniform Resource Identifiers (URIs) is currently gaining momentum. Maintaining a DCAP over time, then, may involve improving its precision incrementally by identifying its terms with URIs as the URIs become available; this is referred to here as the Principle of Appropriate Identification.

In the meantime, these guidelines aim at the more modest aim of providing system developers and information specialists with a normalized and readable view of Dublin-Core-based metadata models. A DCAP should include enough information to be of optimal usefulness for its intended audience – a Principle of Readability – even if this entails the redundant inclusion of information, which, in a formal system of machine-processable schemas, might otherwise be fetched dynamically from external sources.

Given the flexibility of presentation required by the Principle of Readability, no assumption is made that DCAPs will be convertible into future machine-understandable forms without the use of ad-hoc heuristics or manual intervention. Creators of DCAPs should bear in mind that a normalized form of documentation cannot itself address the deeper problems of interoperability in a world with a diversity of underlying metadata models – problems which will continue to challenge the metadata community as a whole, and the Dublin Core Metadata Initiative in particular, for the foreseeable future.

1 Scope

The present document gives guidance on how information should be structured and presented in Dublin Core Application Profiles. Principles and concepts underlying DCAPs as declarative metadata constructs are defined and explained.

The guidelines do not mandate a particular document format for DCAPs. DCAPs may be presented as plain text files or as Web pages, word-processing files, PowerPoint, or indeed as ink on paper. By providing a consistent presentation structure for such documents, these guidelines aim at making it easier for people to understand what others are doing in their metadata. The guidelines mandate enough structure to ensure that DCAPs will be convertible as straightforwardly as possible into expressions that use schema languages, such as RDF, for automatic processing by machines. In this sense, a normalized documentational form for DCAPs is a first step towards the more ambitious and long-term goal of automating semantic interoperability across a broad diversity of information sources.

2 Definitions

Dublin Core Application Profile (DCAP): A DCAP is a declaration specifying which metadata terms an organization, information provider, or user community uses in its metadata and how those terms have been customized or adapted to a particular application. By definition, a DCAP is based in part on Dublin Core and follows DCMI Grammatical Principles [DCMI-PRINCIPLES]. A DCAP consists of a Descriptive Header and one or more Term Usages.

DCMI Grammatical Principles: As maintained by the Dublin Core Metadata Initiative, DCMI grammatical principles specify a typology of metadata terms – Elements, Element Refinements, Encoding Schemes, and Vocabulary Terms – along with their interrelationships and functions [DCMI-PRINCIPLES]. A DCAP is based on the simple model of a resource described with a flat set of properties. This is consistent with DCMI grammatical principles, which do not themselves specify more elaborate models.

Descriptive Header: A Descriptive Header places the DCAP into an interpretive context by specifying, at a minimum, a Title, Creator, Date, Identifier, and Description for the DCAP. An optional Preamble may comment on any technical or stylistic conventions followed in the DCAP.

Term Usage: A Term Usage is a description of a metadata term, which, at a minimum, identifies a metadata term in accordance with the Principle of Appropriate Identification by using one or more identifying attributes – Term URI, Defined By, Name, Label – as described in Section 3. Optionally, a Term Usage may also describe or annotate a term in more detail by providing additional definitional attributes, relational attributes, or constraints, as described in Section 4.

Principle of Appropriate Identification: The Principle of Appropriate Identification dictates that metadata terms be identified as precisely as possible. As established in the so-called CORES Resolution of December 2002, the preferred method for identifying a metadata term is to cite its Uniform Resource Identifier (URI). All DCMI metadata terms are identified with URIs, and URIs are currently being assigned to the terms of other major semantic standards such as MARC21 and IEEE/LOM [CORES-RESOLUTION]. Whenever such URIs are available they should be cited as an attribute of a Term Usage (its "Term URI"). Terms to which URIs have not (or not yet) been assigned should be identified using other attributes as appropriate, as described in Section 3.

Principle of Readability: The Principle of Readability dictates that a DCAP should include enough information in Term Usages to be of optimal usefulness for the intended audience of the DCAP – even if this entails the redundant inclusion of information which, in a formal system of machine-processable schemas, might otherwise be fetched dynamically from external sources. Conversely, the Principle of Readability allows unused attributes simply to be omitted from display.

3 Identifying terms with appropriate precision

Application profiles serve to clarify who is declaring and maintaining the metadata semantics that a group wants to share. This section describes how a metadata term used in a Term Usage can be identified with appropriate precision (the Principle of Appropriate Identification).

At present, the preferred method for identifying a metadata term is to cite its Uniform Resource Identifier (URI) if such is available. A URI is "a compact string of characters for identifying an abstract or physical resource" constructed according to a generic and flexible syntax [URI]. The World Wide Web Consortium has promoted the notion that “All important resources should be identified by a URI” [WEBARCH] and has specifically promoted the use of URIs for identifying metadata elements. In the CORES Resolution of December 2002, the maintainers of seven leading metadata standards – Dublin Core, IEEE/LOM, DOI, CERIF, MARC21, ONIX, and GILS -- pledged to assign URIs to their elements and to articulate policies for the persistence of those URIs [CORES-RESOLUTION]. (Note that a URI, when used to identify a metadata term, often functions as a Web address for accessing information about that term, such as a Web page or machine-processable schema. However, the CORES Resolution does not require that such identifiers resolve to such resources, and URIs that result in “file not found” messages are not necessarily “broken” as identifiers.)

For metadata terms to which a URI has been officially assigned – for example, by DCMI or by another signatory of the CORES Resolution – that URI should be cited in the field “Term URI”. For example, the Dublin Core element “Audience” should be cited as “http://purl.org/dc/terms/audience”. As this form of identification is precise and sufficient on its own, other identifying fields may be left blank:

```
Term URI        http://purl.org/dc/terms/audience
Name            -
Label           -
Defined By      -
```

In accordance with the Principle of Readability, other identifying attributes such as Name and Label could be added here to make the DCAP more “reader-friendly”. If the DCAP is intended as a guide for processing metadata records, it may indeed be necessary to provide a Name (i.e., the string actually used in the metadata records). If the Name or Label of a term are considered more “reader-friendly” as captions for a Term Usage than the Term URI, the order of these attributes may be changed to put these first. See Sections 5.4 (“Attributes copied from external sources”) and 5.2 (“Readability of Term Usages”) for further discussion.

A term that has been declared or documented somewhere but not assigned a URI (as far as one knows) should be identified as precisely as possible by providing its name and pointing to a declarative document or schema in which it has been defined. The declarative document or schema should be cited with URI, Web address, or bibliographic reference in the field "Defined By". The term itself can be cited using either a string identifier or token (in the field “Name”, which by default is assumed to be case-sensitive) or a natural-language label (in the field "Label"), or both, taken from the declarative document or schema:

```
Term URI        -
Name            AttendancePattern
Label           Attendance Pattern
Defined By      http://someones-project.org/schema.html
```

For a term that has not already been defined in any other declarative document, the field Defined By should simply cite the URI of the DCAP itself (as assigned with Identifier in the DCAP Descriptive Header). For example, in a DCAP with the URI "http://my-project.org/profile.html", a new local term called Star Ratings could be defined as follows:

```
Term URI        -
Name            StarRatings
Label           Star Ratings
Defined By      http://my-project.org/profile.html
```

A creator of a DCAP wishing to declare locally coined terms in a way that makes them citable with precision, and thus re-usable by others, may undertake the additional step of assigning them URIs. At present, the technical conventions and "Web etiquette" for naming metadata terms with URIs have yet to establish themselves in common practice, though at a minimum it seems both polite and sensible not to promote new URIs unless it is expected they will be maintained. For the purposes of DCAPs, DCMI itself provides models of practice, and further options are likely to emerge as the CORES Resolution is implemented [DCMI-NAMESPACE, DCMI-TERMS, DCMI-SCHEMAS]. Note that the CORES Resolution itself addresses the use of URIs as identifiers only and is silent on whether the URIs should resolve to informational Web pages or schemas [CORES-RESOLUTION].

4 Attributes of a Term Usage

Attributes for describing the metadata terms "used" in a DCAP are listed below. Note that they are called "attributes" here simply to avoid confusingly recursive formulations such as "terms for describing terms".

Use of Identifying Attributes in Term Usages (see Section 4.1) is governed by the Principle of Appropriate Identification. According to this principle, a Term Usage should use one or more of the four Identifying Attributes to identify a term as precisely as appropriate – i.e., with a formally assigned URI if available, or alternatively by citing a name or label for the term along with a reference to a document, schema, or Web page in which that term is defined.

All of the other attributes of Term Usages are optional and should be used as local needs may dictate. As discussed in Section 5.4, “local” and “source” attributes may be distinguished as necessary.

4.1 Identifying attributes

Term URI	A Uniform Resource Identifier used to identify the term.
Name	A unique token assigned to the term.
Label	A human-readable label assigned to the term.
Defined By	An identifier of a namespace, pointer to a schema, or bibliographic reference for a document within which the term is defined.

4.2 Definitional attributes

Definition	A statement that represents the concept and essential nature of the term.
Comments	Additional information about the term or its application.
Type of term	A grammatical category of the term (e.g., "Element", "Element Refinement", or "Encoding Scheme").

4.3 Relational attributes

Refines	The described term semantically refines the referenced term.
Refined By	The described term is semantically refined by the referenced term.
Encoding Scheme For	The described term, an Encoding Scheme, qualifies the referenced term.
Has Encoding Scheme	The described term is qualified by the referenced Encoding Scheme.
Similar To	The described term has a meaning the same as, or similar to, that of the referenced term.

4.4 Constraints

Obligation	Indicates whether the element is required to always or sometimes be present (i.e., contain a value). Examples include "Mandatory", "Conditional", and "Optional".)
Condition	Describes the condition or conditions according to which a value shall be present.
Datatype	Indicates the type of data that can be represented in the value of the element.
Occurrence	Indicates any limit to the repeatability of the element.

5 Discussion

5.1 Descriptive Headers

By definition, a DCAP consists of a Descriptive Header and one or more Term Usages (see Section 2). The Description Header should include the following:

- A brief description of the DCAP based on Dublin Core. At a minimum, the description should specify a Title, Contributor, Date, Identifier, and Description, as explained in more detail in Annex A. Ideally, the description of the DCAP will elaborate on the context in which the DCAP is intended to be used.
- Optionally, a Preamble for the DCAP should describe any technical or formatting conventions used in the DCAP. For example, if namespace prefixes are used in the Name field (see Section 5.6), these prefixes should be documented here. The Preamble can also cite Web pages or schemas in which the terms used in the DCAP are documented and defined so that such information does not need to be repeated in the "Defined By" field of each individual Term Usage; see the example in Section 6.1.1.

5.2 Readability of Term Usages

By default, each term cited in a DCAP should be described with its own Term Usage – a table with a full set of attributes on the left and attribute values on the right. In accordance with the Principle of Readability, however, the intended use of a DCAP may dictate a different presentational style: while DCAPs intended for use by software developers will need to be explicit and detailed, DCAPs intended primarily as informational documents for human consumption can (and often should) be much terser. The following are several ways in which Term Usages may be formatted for readability:

- Instead of creating a separate Term Usage for every Element Refinement and Encoding Scheme used in an application, such terms may simply be cited in the attributes Refined By or Has Encoding Scheme of the Term Usages of the Elements to which they refer (see Section 5.3). Note that this terser style does not support the addition of usage notes, local definitions, or annotations, for which a full Term Usage must be used.
- Attributes not needed for Term Usages can simply be omitted. At one extreme, a Term Usage might legitimately consist of just a Name and a Term URI; see the example in Section 6.1.2).
- The order of attributes presented in Section 4 is significant only for the usability of DCAPs as documents – not for future machine-processable representations of DCAPs. Authors of DCAPs may therefore change the order of attributes in the interest of readability, though they should bear in mind that any such changes may make it more difficult for people to compare two DCAPs visually. For examples of how Name (in boldface) is placed before Term URI, see Section 6.1.2 and DCMI-TERMS.
- In the interest of readability, it might make sense to describe Elements, Element Refinements, and Encoding Schemes with different subsets of relevant attributes. Indeed, these different types of terms might be grouped under separate sections of the DCAP document. (For example, see how Elements and Element Refinements are separated from Encoding Schemes in the document "DCMI Metadata Terms" [DCMI-TERMS].)
- In the interest of readability and of future machine-parsability, attributes should be repeated when necessary (as opposed to listing multiple values for a single attribute).

5.3 “Using” Element Refinements, Encoding Schemes, and Vocabulary Terms

5.3.1 Using Vocabulary Terms

According to DCMI Grammatical Principles, a Vocabulary Term is a member of a controlled vocabulary of values, and a controlled vocabulary of values (as a whole) is named by an Encoding Scheme [DCMI-PRINCIPLES].

In general, it is not the role of application profiles to declare controlled vocabularies of values, either in the sense of creating lists of potential values or in the sense of giving that list (as a whole) a name and URI. Sets of Vocabulary Terms are most appropriately declared in separately citable documents external to a DCAP.

However, if the creator of a DCAP merely wishes to specify a short list of possible values (e.g., "Animal, Vegetable, or Mineral"), these can be simply listed in a "Comment" field.

5.3.2 Using Encoding Schemes

There are three ways to cite Encoding Schemes in a DCAP:

- The most concise way is to use the attribute Has Encoding Scheme (or Comment) for a blanket reference to a set of encoding schemes documented elsewhere. The DCAP for Resource Discovery Network, for example, simply cites “RDN Subject Encoding Schemes” and gives a URL where the list of those encoding schemes may be found; see the example in Section 6.1.2.
- A more precise way is to use the attribute Has Encoding Scheme, repeated as necessary, to cite each Encoding Scheme by its URI (or by Name and URI); for example, see the example in Section 6.3.2.
- In addition to citing Encoding Schemes in the Has Encoding Scheme attribute of Elements, creators of DCAPs may want to describe Encoding Schemes in stand-alone Term Usages in order to annotate their usage, for example by specifying a Datatype, Occurrence, or Local Definition. The attribute Encoding Scheme For points back to the Element or Element Refinement qualified.

5.3.3 Using Element Refinements

The options for Element Refinements are analogous to those for Encoding Schemes:

- Statements such as “all terms in Vocabulary D can be used as element refinements for Contributor” can be simply recorded in a Refined By attribute (or as a Comment).
- Element Refinements can be cited one-by-one using the attribute Refined By; see the example in Section 6.3.2.
- Element Refinements can additionally be described in separate Term Usages.

5.4 Attributes copied from external sources

Ideally, application profiles would be dynamically up-dated with information on the terms they use directly from schemas on the Web and this information would be integrated with local annotations into a “one-stop” document for the convenience of users. The use of machine-understandable DCAPs may some day make this possible.

In the meantime, however, creators of DCAPs who wish to include definitions or other such information from original source documents in their Term Usages have no choice but to copy that information from the source. While the Principle of Readability specifically permits this, authors of DCAPs should bear in mind that copied information, if not maintained, can go out of alignment with the official source.

Where information copied from external sources is supplied, this fact should be reported in the Preamble as described in Section 5.1. Where it is necessary to distinguish in a DCAP between attributes defined locally and attributes copied from an external source, the DCAP should establish its own document-internal convention, such as distinguishing between a Local Definition and a Source Definition; see the example in Section 6.3.2.

5.5 Types of Comments

Past creators of application profiles for Dublin Core have invented many types of annotation, the most popular of which have been Notes, Best Practice, Usage, Scope, Open Questions, Examples, Purpose, Guidelines, and Don't Confuse With. While the present guidelines lump all of the above into a generically named Comments field, creators of DCAPs may wish to repeat this field with different labels as needed. The needs of future machine processing do not now seem to dictate tighter uniformity in this area.

5.6 Term URIs versus Qualified Names

In the sense intended here, Qualified Names are names of metadata terms that are “qualified” with a prefix standing for a namespace with which the terms are associated (a “namespace prefix”). For example, the Dublin Core element “Title” is sometimes referenced in metadata records and usage documentation using a namespace prefix such as “DC.” or “dc:” as in “DC.Title” or “dc:title”. As straightforward as this citation method may seem, it is based on assumptions about the nature of “namespace” that cannot be assumed to hold across different application environments (e.g., HTML versus RDF versus relational databases) or metadata communities (e.g., for citing elements from standards other than Dublin Core), and at any rate it presupposes an additional mechanism or declaration for associating prefixes with the proper namespaces.

For such reasons, it is far better to cite an element with a full URI – indeed, this is the only method supported by the CORES Resolution and by DCMI policy [CORES-RESOLUTION, DCMI-NAMESPACE]. According to the Principle of Appropriate Identification followed in these guidelines, a Term URI must be cited when available.

On the other hand, long strings such as “http://purl.org/dc/elements/1.1/title” are not very readable and may be misunderstood by the average reader of a DCAP. In accordance with the Principle of Readability, therefore, the author of a DCAP may choose to use qualified names (e.g., “dc:title”) in the “Name” field – as long as any prefixes used are explained in the Preamble of the DCAP, and as long as any available Term URIs are cited as well.

5.7 Declaring new elements

There is nothing to restrain the creator of a DCAP from creating new URIs as identifiers for locally coined metadata terms. For reasons discussed above in Section 3, one should perhaps pause for reflection before taking this step, and if URIs are declared, this step should perhaps be documented separately and not embedded “in passing” into a DCAP full of Term Usages. Any URIs declared for use in a DCAP might best be formed by following the DCMI algorithm and concatenating the URL of the DCAP (e.g., "http://myproject.org/profile/") and the Name of the term (e.g., "starRatings") into a single string (e.g., "http://myproject.org/profile/starRatings") [DCMI-NAMESPACE]. Other models for forming URIs as identifiers for metadata elements are emerging with the implementation of the CORES Resolution [CORES-RESOLUTION].

5.8 Documenting grouped or nested metadata elements

In order to be usable across a diversity of application environments, Dublin Core was designed as a flat set of attributes for describing a resource. In implementation practice, however, Dublin Core elements may be embedded in more elaborate models that group or nest the elements in locally specific ways.

In the absence of a clear and widely accepted data model beyond that of the flat set of attributes, however, applications for integrating metadata from many different sources may be able only to extract and interpret the metadata in terms of Simple Dublin Core, losing any application-specific modelling context. An application designer wishing to document nesting or grouping constructs in a DCAP will need to extend the guidelines described here in order to do so and should bear in mind that documenting such constructs will not in itself guarantee that they will be understood or correctly processed by other applications.

5.9 Documenting unorthodox practices

For reasons both of history and of expedience, a significant number of applications have metadata based on interpretations of the Dublin Core model that are unsound from the standpoint of today's grammatical principles. For example, an application may use CreatorDateOfBirth – an element representing the birth date of a creator of a resource that does not, however, semantically "refine" Creator as its name may imply.

Rather than incorrectly asserting "CreatorDateOfBirth" to be an Element Refinement refining http://purl.org/dc/elements/1.1/creator, the Term Usage in the DCAP should simply record the local name of the element and identify the URI of the DCAP itself as its source. For example, if the DCAP itself is identified by "http://myproject.org/profile/2003/03/17/", the Term Usage should declare the following, leaving empty any fields (such as "Term URI" and "Refines") that would make incorrect assertions about the element:

```
Term URI        -
Local Name      CreatorDateOfBirth
Defined By      http://my-project.org/profile.html
Refines         -
```

Whether "errors" such as "CreatorDateOfBirth" will be of negative consequence for interoperability will depend on how they are interpreted and used in the context of particular applications. The analytical effort involved in creating a DCAP is in effect an important first step towards putting such applications onto a more interoperable foundation.

6 Examples

6.1 UK Resource Discovery Network OAI Application Profile

6.1.1 Descriptive Header

Title	RDN OAI Application Profile
Contributor	Andy Powell
Date	2003-03-23
Identifier	URL for this document – to be assigned
Description	This document expresses the application profile established by the Resource Discovery Network (RDN) to be used by RDN partners for harvesting of records using the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH). The Application Profile is expressed according to guidelines published by the CEN/ISSS [Reference]. Full user documentation for the Application Profile, together with associated XML schemas, is available at http://www.rdn.ac.uk/oai/rdn_dc/. All Dublin Core terms are fully documented at http://www.dublincore.org/documents/dcmi-terms/.

6.1.2 A Term Usage

Name	Subject
Term URI	http://purl.org/dc/elements/1.1/subject
Has Encoding Scheme	DC Subject Encoding Schemes
Has Encoding Scheme	RDN Subject Encoding Schemes
Comment	RDN Subject Encoding Schemes are available from http://www.rdn.ac.uk/publications/cat-guide/subject-schemes/
Obligation	Recommended

6.1.3 Commentary

The DCAP for the UK Resource Discovery Network is formatted in the tersest possible style [RDN]. Note in particular the following:

- The Descriptive Header puts the DCAP into a specific usage context.
- Only the attributes actually used in a given Term Usage are shown. Indeed, most of the Term Usages in this DCAP consist of just a Name and a Term URI.
- Instead of listing all of the DCMI- and RDN-maintained Encoding Schemes as separate Has Encoding Scheme entries (or as separate Term Usages), this DCAP uses shorthand references to “DC Subject Encoding Schemes” and “RDN Subject Encoding Schemes”. Pointers to documentation are given in the Descriptive Header (for the former) and in a Comment field of the Term Usage (for the latter).
- This DCAP increases the readability of Term Usages by listing the Name, in boldface, before the Term URI. This option is discussed in Section 5.2.

6.2 Renardus Application Profile

6.2.1 Descriptive Header

Title	Renardus Application Profile
Contributor	Metadata Working Group SUB Göttingen
Date	18-04-2002
Identifier	http://renardus.sub.uni-goettingen.de/renap/renap.html
Description	Cross search and cross browse European quality controlled subject gateways.

6.2.2 A Term Usage

Term URI	http://purl.org/dc/elements/1.1/language
Name	Language
Label	Language
Defined By	-
Definition	-
Comments	Renardus: The language code is the ISO 639-2, three-letter code. SUB will provide a mapping between the two letter and three letter language code but this will also be found on the LoC site – ISO 639-2: http://lcweb.loc.gov/standards/iso639-2/englangn.html
Type of term	Element
Refines	-
Refined By	-
Encoding Scheme For	-
Has Encoding Scheme	http://purl.org/dc/terms/ISO639-2
Similar To	-
Obligation	Mandatory
Condition	-
Datatype	String
Occurrence	Repeatable

6.2.3 Commentary

The DCAP for the Renardus Project has been formatted in a somewhat more verbose style [RENARDUS]. Note in particular:

- The DCAP uses its own URL as an identifier.

6.3 UK e-Government Metadata Standard Application Profile

6.3.1 Descriptive Header

Addressee	Metadata Working Group, Interoperability Working Group
Contributor	Drafted by Interoperability and Metadata Analyst, Office of the e-Envoy, Cabinet Office, UK farah.ahmed@e-envoy.gsi.gov.uk
Contributor	Metadata Working Group
Coverage.spatial	UK
Creator	Senior Policy Advisor, Interoperability and Metadata, Office of the e-Envoy, Cabinet Office, UK
Date.issued	2003-08-05
Description	The elements and refinements that provide the structure for metadata used by the UK public sector, designed to complement the e-GMS.
Format	Text/MS Word 2003
Identifier	http://purl.oclc.org/NET/e-GMS-AP_v1
Language	Eng
Publisher	Office of the e-Envoy, Cabinet Office, UK. govtalk@e-envoy.gsi.gov.uk
Rights.copyright	http://www.hmso.gov.uk/docs/copynote.htm Crown Copyright
Source	http://purl.oclc.org/NET/e-GMS_v2
Status	Version 1.0 For publication
Subject	Metadata
Subject.category	Information management
Title	UK e-government metadata standard application profile version 1

6.3.2 A Term Usage

Term URI	http://purl.org/dc/elements/1.1/date
Defined by	http://purl.org/dc/elements/1.1/
Name	Date
Label	Date
Source Definition	A date associated with an event in the life cycle of the resource.
Source Comment	-
Local Definition	-
Local Comment: Purpose	To enable the user to find the resource by limiting the number of search hits according to a date, e.g. the date the resource was made available.
Local Comment: Notes	Dates need to appear in a format that is recognisable to people all over the world, and that can be interpreted by computer software. The W3C format allows accurate searching, and makes it clear which is the year, month or day. The format is ‘ccyy-mm-dd’, where ‘ccyy’ is the year, ‘mm’ is the month and ‘dd’ the day. When the time is also needed, add ‘hh:mm’, where ‘hh’ is the hour (using the 24 hour clock), ‘mm’ is minutes. More about this notation can be found at http://www.w3.org/TR/NOTE-datetime.

Local Comment: Not to be confused with	Coverage – Date refers to dates relevant to the information resource itself, not the information held within the resource. For example, for a document about the civil service in the 18th century, put ‘18th century’ in Coverage and put the date published in Date.
Type of term	Element
Refines	-
Refined by	http://www.govtalk.gov.uk/terms/dateAcquired
Refined by	http://purl.org/dc/terms/dateAccepted
Refined by	http://purl.org/dc/terms/Available
Refined by	http://purl.org/dc/terms/dateCopyrighted
Refined by	http://purl.org/dc/terms/created
Refined by	http://purl.org/dc/terms/issued
Refined by	http://purl.org/dc/terms/dateSubmitted
Refined by	http://purl.org/dc/terms/valid
Refined by	http://purl.org/dc/terms/modified
Refined by	http://www.govtalk.gov.uk/terms/cutOffDate
Refined by	http://www.govtalk.gov.uk/terms/dateDeclared
Refined by	http://www.govtalk.gov.uk/terms/dateClosed
Refined by	http://www.govtalk.gov.uk/terms/nextVersionDue
Refined by	http://www.govtalk.gov.uk/terms/updatingFrequency
Encoding Scheme For	-
Has Encoding Scheme	http://purl.org/dc/terms/W3CDTF
Has Encoding Scheme	http://purl.org/dc/terms/Point
Similar To	-
Constraints	The value must always be taken from the specified encoding scheme, with the exception of the ‘updatingFrequency’ refinement.
Obligation	Mandatory
Condition	A value must be given either for the unqualified date or at least one date refinement
Datatype	-

6.3.3 Commentary

The DCAP for the UK e-Government Metadata Standard is formatted in the most detailed and specific possible style [EGMS]. While this results in a significantly longer document than the DCAPs for RDN and Renardus, such specificity may be helpful to developers of applications that need to create or process metadata based on the DCAP. Note in particular the following:

- Encoding Schemes and Element Refinements are listed using repeated fields in the Term Usage of the Element to which they refer. In addition, each Encoding Scheme and Element Refinement is also described in its own Term Usage, which allows information about each of them, such as Definition and Constraints, to be recorded in the DCAP as well.

- The Term Usage marks information coming from outside sources: the “Source Definition” copies the definition of Date from DCMI documentation, while the “Local Comment” supplies usage information local to this DCAP.

Annex A: Metadata describing a DCAP

A DCAP should itself be described with Dublin Core metadata, either in a header or in a separate metadata record. At a minimum, this description should include:

Title	A name for the Application Profile.
Contributor	A creator or maintainer of the Profile.
Date	The date of last modification.
Identifier	An unambiguous reference to the Profile. Best practice is to provide a URL by which a copy of the document or schema can be retrieved over the Web.
Description	A concise description of the Profile. As appropriate, the description should elaborate on the context and purposes in which the DCAP is intended to be used; the organizations or individuals involved in its development; any arrangements, policies, or intentions regarding the future development and maintenance of the DCAP; or technical characteristics of the instance metadata or database described.

Annex B: Options for machine-interpretable DCAPs

DCAPs can be expressed in machine-interpretable schema languages, and such machine-interpretable schemas can be manipulated by software applications. This CWA does not give detailed recommendations on how such schemas should be structured, as a number of issues are still open for debate. The scope of this CWA is limited to recommending how application profiles can be expressed as text documents. Future options for machine-interpretable DCAPs are outlined below.

Currently, two schema languages specified by W3C might be considered: XML Schema [XML-SCHEMA] and RDF Schema [RDF-SCHEMA]. The choice of schema language will be influenced by the functionality that the schema is intended to support – for example, whether it is required as a predictable format for data exchange or intended to support inferences about existing metadata. Such different objectives imply different choices between the two schema languages. There has been some discussion on ways to combine XML Schema and RDF Schema to more fully express characteristics of application profiles [HUNTER]. More recently there has been an attempt within the W3C to differentiate RDF Schema as a vocabulary description language and XML Schema as a basis for providing structured data exchange.

An XML schema provides a structured expression that supports validation of instance metadata. In effect, an XML schema provides a document "template" which acts as an exchange format for metadata instances. An XML Schema serves the same function as an XML DTD with additional capability for extensibility and namespace handling.

An RDF schema expresses relationships between terms, providing a data model for expressing the semantics of terms – their properties, classes, and definitions. The underlying RDF data model combined with the use of unique identifiers allows software to infer relationships between terms and perform data aggregation.

RDF Schemas are effective for expressing the semantics of application profiles, whilst XML Schemas are more effective for expressing cardinality, data-typing, and constraints. Possible approaches to the expression of application profiles in RDF have been explored within projects such as SCHEMAS [BAKER] and MEG [MEG-REGISTRY].

Bibliography

[BAKER] Thomas Baker, Makx Dekkers, Rachel Heery, Manjula Patel, Gauri Salokhe, What terms does your metadata use? Application profiles as machine-understandable narratives. Journal of Digital Information 2:2 (November 2001), http://jodi.ecs.soton.ac.uk/Articles/v02/i02/Baker.

[CORES-RESOLUTION] Thomas Baker, Makx Dekkers, Identifying Metadata Elements with URIs: the CORES Resolution. D-Lib Magazine (July 2003), http://www.dlib.org/dlib/july03/baker/07baker.html.

[DC-LIBRARY] Library Application Profile, http://dublincore.org/documents/2002/09/24/library-application-profile/.

[DCMI-NAMESPACE] Andy Powell, Harry Wagner, Stuart Weibel, Tom Baker, Tod Matola, Eric Miller, Namespace policy for the Dublin Core Metadata Initiative, http://dublincore.org/documents/dcmi-namespace/.

[DCMI-PRINCIPLES] DCMI Grammatical Principles, http://dublincore.org/usage/documents/principles/.

[DCMI-SCHEMAS] DCMI Schemas, http://dublincore.org/schemas/.

[DCMI-TERMS] DCMI Metadata Terms, http://dublincore.org/documents/dcmi-terms/.

[EGMS] Office of the e-Envoy – Cabinet Office, UK e-Government Metadata Standard Application Profile Version 1, http://purl.oclc.org/NET/eGMSAPv1.

[HEERY] Rachel Heery, Manjula Patel, Application profiles: mixing and matching metadata schemas, Ariadne 25, September 2000, http://www.ariadne.ac.uk/issue25/app-profiles/intro.html.

[HUNTER] Jane Hunter, Carl Lagoze, Combining RDF and XML Schemas to enhance interoperability between metadata application profiles. WWW10, May 1-5, 2001, Hong Kong, http://www10.org/cdrom/papers/572/index.html.

[MEG-REGISTRY] Rachel Heery, Pete Johnston, Dave Beckett, Damian Steer, The MEG Registry and SCART: Complementary Tools for Creation, Discovery and Re-use of Metadata Schemas. In: Proceedings of the International Conference on Dublin Core and Metadata for e-Communities, 2002. Florence: Firenze University Press, 2002, pp. 125-132, http://www.bncf.net/dc2002/program/ft/paper14.pdf.

[RDF-SCHEMA] Brickley, Dan and Guha, R.V, editors. RDF Vocabulary Description Language 1.0: RDF Schema W3C Working Draft 23 January 2003, http://www.w3.org/TR/rdf-schema/.

[RDN] Powell, Andy, RDN OAI Application Profile, [URL to be assigned].

[RENARDUS] Metadata Working Group SUB Goettingen, Renardus Application Profile, http://renardus.sub.uni-goettingen.de/renap/renap.html.

[URI] T. Berners-Lee, R. Fielding, L. Masinter, Uniform Resource Identifiers (URI): Generic Syntax, August 1998, http://www.ietf.org/rfc/rfc2396.txt.

[WEBARCH] Ian Jacobs, ed., Architecture of the World Wide Web, http://www.w3.org/TR/webarch/.

[XML-SCHEMA] Thompson, Henry S. et al., editors. XML Schema Part 1: Structure. W3C Recommendation 2 May 2001, http://www.w3.org/TR/xmlschema-1/.

Title:	DCMI Usage Board Review of Application Profiles
Creator:	Thomas Baker
Identifier:	http://dublincore.org/usage/documents/2003/02/11/profiles/
Latest version:	http://dublincore.org/usage/documents/profiles/
Date modified:	2003-02-11
Description:	This document defines the term "Application Profile" in the context of the Dublin Core Metadata Initiative. Criteria for Usage Board review of Application Profiles and guidelines for submission are outlined in the DCMI Usage Board Administrative Processes document [PROCESS].

"Application Profile" defined

For the purposes of DCMI Usage Board review, an Application Profile (AP) is a declaration of which metadata terms an organization, information resource, application, or user community uses in its metadata. Moreover:

-- By definition, an AP cannot "declare" new metadata terms and definitions; it only "reuses" terms from existing element sets [HEERY].

-- The ideal element set will use URIs to uniquely identify its terms within XML namespaces [DCMI-NAMESPACE]. As of 2002, however, this cannot be required.

-- By definition, any new term coined for use in an AP must first be declared in a form citable in the AP.

```
--  An AP may also provide additional documentation
    on how the terms used are constrained, encoded, or
    interpreted for particular purposes.

As of 2002, APs are seen primarily as a form of documentation,
the purpose of which is to help implementor communities
harmonize their metadata practice.  It is hoped that in the
longer term, machine-processable versions of such APs based
on data models such as RDF will provide a basis for automating
metadata interoperability functions such as semantic crosswalks
and format conversions.


References
```

[DCMI-NAMESPACE] Andy Powell, Harry Wagner, Stuart Weibel, Tom Baker, Tod Matola, Eric Miller, Namespace policy for the Dublin Core Metadata Initiative, http://dublincore.org/documents/dcmi-namespace/.

[HEERY] Rachel Heery and Manjula Patel, Application profiles: mixing and matching metadata schemas, Ariadne 25, September 2000, http://www.ariadne.ac.uk/issue25/app-profiles/intro.html.

[PROCESS] http://dublincore.org/usage/documents/process/.

Metadata associated with this resource: http://dublincore.org/usage/documents/profiles/index.shtml.rdf

Title:	**Using Dublin Core**
Creator:	Diane Hillmann
Date Issued:	2003-08-26
Identifier:	http://dublincore.org/documents/2003/08/26/usageguide/
Replaces:	http://dublincore.org/documents/2001/04/12/usageguide/
Is Replaced By:	Not applicable
Latest Version:	http://dublincore.org/documents/usageguide/
Translations:	http://dublincore.org/resources/translations/
Status of Document:	DCMI Recommended Resource
Description of Document:	This document is intended as an entry point for users of Dublin Core. For non-specialists, it will assist them in creating simple descriptive records for information resources (for example, electronic documents). Specialists may find the document a useful point of reference to the documentation of Dublin Core, as it changes and grows.

TABLE OF CONTENTS

1. INTRODUCTION

1.1. What is Metadata?

Metadata has been with us since the first librarian made a list of the items on a shelf of handwritten scrolls. The term "meta" comes from a Greek word that denotes "alongside, with, after, next." More recent Latin and English usage would employ "meta" to denote something transcendental, or beyond nature. Metadata, then, can be thought of as data about other data. It is the Internet-age term for information that librarians traditionally have put into catalogs, and it most commonly refers to descriptive information about Web resources.

A metadata record consists of a set of attributes, or elements, necessary to describe the resource in question. For example, a metadata system common in libraries -- the library catalog -- contains a set of metadata records with elements that describe a book or other library item: author, title, date of creation or publication, subject coverage, and the call number specifying location of the item on the shelf.

The linkage between a metadata record and the resource it describes may take one of two forms:

1. elements may be contained in a record separate from the item, as in the case of the library's catalog record; or
2. the metadata may be embedded in the resource itself.

Examples of embedded metadata that is carried along with the resource itself include the Cataloging In Publication (CIP) data printed on the verso of a book's title page; or the TEI header in an electronic text. Many metadata standards in use today, including the Dublin Core standard, do not prescribe either type of linkage, leaving the decision to each particular implementation.

Although the concept of metadata predates the Internet and the Web, worldwide interest in metadata standards and practices has exploded with the increase in electronic publishing and digital libraries, and the concomitant "information overload" resulting from vast quantities of undifferentiated digital data available online. Anyone who has attempted to find information online using one of today's popular Web search services has likely experienced the frustration of retrieving hundreds, if not thousands, of "hits" with limited ability to refine or make a more precise search. The wide scale adoption of descriptive standards and practices for electronic resources will improve retrieval of relevant resources in any venue where information retrieval is critical. As noted by Weibel and Lagoze, two leaders in the field of metadata development:

> "The association of standardized descriptive metadata with networked objects has the potential for substantially improving resource discovery capabilities by enabling field-based (e.g., author, title) searches, permitting indexing of non-textual objects, and allowing access to the surrogate content that is distinct from access to the content of the resource itself." (Weibel and Lagoze, 1997)

In the last years we have also seen an increase in the application of Dublin Core metadata in more closed environments. There are implementations where Dublin Core metadata is used to describe resources held, owned or produced by companies, governments and international organisations to supporting portal services or internal knowledge management. There are also implementations where Dublin Core metadata is used as a common exchange format supporting the aggregation of collections of metadata, such as the case of the Open Archive Initiative. In these cases, like in the open environment of the Web, the concept of standardized descriptive metadata provides a powerful mechanism to improve retrieval for specific applications and specific user communities. It is this need for "standardized descriptive metadata" that the Dublin Core addresses.

1.2. What is the Dublin Core?

The Dublin Core metadata standard is a simple yet effective element set for describing a wide range of networked resources. The Dublin Core standard includes two levels: Simple and Qualified. Simple Dublin Core comprises fifteen elements; Qualified Dublin Core includes an additional element, Audience, as well as a group of element refinements (also called qualifiers) that refine the semantics of the elements in ways that may be useful in resource discovery. The semantics of Dublin Core have been established by an international, cross-disciplinary group of professionals from librarianship, computer science, text encoding, the museum community, and other related fields of scholarship and practice.

Another way to look at Dublin Core is as a "small language for making a particular class of statements about resources". In this language, there are two classes of terms--elements (nouns) and qualifiers (adjectives) -- which can be arranged into a simple pattern of statements. The resources themselves are the implied subjects in this language. (For additional discussion of Dublin Core Grammar, see "DCMI Grammatical Principles") In the diverse world of the Internet, Dublin Core can be seen as a "metadata pidgin for digital tourists": easily grasped, but not necessarily up to the task of expressing complex relationships or concepts.

The Dublin Core basic element set is outlined in Section 4. Each element is optional and may be repeated. Most elements also have a limited set of qualifiers or refinements, attributes that may be used to further refine (not extend) the meaning of the element. The Dublin Core Metadata Initiative (DCMI) has established standard ways to refine elements and encourage the use of encoding and vocabulary schemes. The full set of elements and element refinements conforming to DCMI "best practice" is available, with a formal registry in process.

Three other Dublin Core principles bear mentioning here, as they are critical to understanding how to think about the relationship of metadata to the underlying resources they describe.

1. The One-to-One Principle. In general Dublin Core metadata describes one manifestation or version of a resource, rather than assuming that manifestations stand in for one another. For instance, a jpeg image of the Mona Lisa has much in common with the original painting, but it is not the same as the painting. As such the digital image should be described as itself, most likely with the creator of the digital image as Creator or Contributor, rather than the painter of the original Mona Lisa. The relationship between the metadata for the original and the reproduction is part of the metadata description, and assists the user in determining whether he or she needs to go to the Louvre for the original, or whether his/her need can be met by a reproduction.

2. The Dumb-down Principle. The qualification of Dublin Core properties is guided by a rule known colloquially as the Dumb-Down Principle. According to this rule, a client should be able to ignore any qualifier and use the value as if it were unqualified. While this may result in some loss of specificity, the remaining element value (minus the qualifier) must continue to be generally correct and useful for discovery. Qualification is therefore supposed only to refine, not extend the semantic scope of a property.

3. Appropriate values. Best practice for a particular element or qualifier may vary by context, but in general an implementor cannot always predict that the interpreter of the metadata will always be a machine. This may impose certain constraints on how metadata is constructed, but the requirement of usefulness for discovery should be kept in mind.

Although the Dublin Core was originally developed with an eye to describing document-like objects (because traditional text resources are fairly well understood), DC metadata can be applied to other resources as well. Its suitability for use with particular non-document resources will depend to some extent on how closely their metadata resembles typical document metadata and also what purpose the metadata is intended to serve. (Implementors interested in using Dublin Core for diverse resources are encouraged to browse the Dublin Core Projects pages for ideas on using Dublin Core metadata for their resources.)

Dublin Core has as its goals:

Simplicity of creation and maintenance

> The Dublin Core element set has been kept as small and simple as possible to allow a non-specialist to create simple descriptive records for information resources easily and inexpensively, while providing for effective retrieval of those resources in the networked environment.

Commonly understood semantics

> Discovery of information across the vast commons of the Internet is hindered by differences in terminology and descriptive practices from one field of knowledge to the next. The Dublin Core can help the "digital tourist" -- a non-specialist searcher -- find his or her way by supporting a common set of elements, the semantics of which are universally understood and supported. For example, scientists concerned with locating articles by a particular author, and art scholars interested in works by a particular artist, can agree on the importance of a "creator" element. Such convergence on a common, if slightly more generic, element set increases the visibility and accessibility of all resources, both within a given discipline and beyond.

International scope

> The Dublin Core Element Set was originally developed in English, but versions are being created in many other languages, including Finnish, Norwegian, Thai, Japanese, French, Portuguese, German, Greek, Indonesian, and Spanish. The DCMI Localization and Internationalization Special Interest Group is coordinating efforts to link these versions in a distributed registry.
>
> Although the technical challenges of internationalization on the World Wide Web have not been directly addressed by the Dublin Core development community, the involvement of representatives from virtually every continent has ensured that the development of the standard considers the multilingual and multicultural nature of the electronic information universe.

Extensibility

While balancing the needs for simplicity in describing digital resources with the need for precise retrieval, Dublin Core developers have recognized the importance of providing a mechanism for extending the DC element set for additional resource discovery needs. It is expected that other communities of metadata experts will create and administer additional metadata sets, specialized to the needs of their communities. Metadata elements from these sets could be used in conjunction with Dublin Core metadata to meet the need for interoperabilbility. The DCMI Usage Board is presently working on a model for accomplishing this in the context of "application profiles."

Rachel Heery and Manjula Patel, in their article "Application profiles: mixing and matching metadata schemas" define an application profile as:

> " ... schemas which consist of data elements drawn from one or more namespaces, combined together by implementors, and optimised for a particular local application."

This model allows different communities to use the DC elements for core descriptive information, and allowing domain specific extensions which make sense within a more limited arena.

1.3. The Purpose and Scope of This Guide

This document is intended to be an entry point for users of Dublin Core. For non-specialists, it will assist in creating simple descriptive records for information resources (for example, electronic documents, JPEG images, video clips). Specialists may find the document a useful point of reference to the documentation of Dublin Core, as it changes and grows.

"Using Dublin Core" will show in a non-technical fashion how Dublin Core metadata may be used by anyone to make their material more accessible. It discusses the principles, structure and content of Dublin Core metadata elements, how to use them in composing a complete Dublin Core metadata record, as well as how to qualify elements to support use by a wide variety of communities.

Another important goal of this document is to promote "best practices" for describing resources using the Dublin Core element set. The Dublin Core community recognizes that consistency in creating metadata is an important key to achieving optimal retrieval and intelligible display across disparate sources of descriptive records. Inconsistent metadata effectively hides desired records, resulting in uneven, unpredictable or incomplete search results.

As a general introduction, this document is necessarily brief, and cannot address all the issues implementors may encounter while planning their use of metadata. Several avenues remain for those who have additional questions beyond those addressed in this guide.

1. Appended to this guide are references to relevant articles and other resources, including those with more technical guidance for implementors
2. The Dublin Core Website contains references to additional documents and resources of the DCMI community and ways for implementors to become involved in the DCMI
3. Specific questions can be addressed to AskDCMI. In addition to fielding questions, the AskDCMI service maintains a searchable archive of already answered questions and links to additional resources.

2. Syntax Issues

In this guide, we have chosen to represent Dublin Core examples in a "generic" form (Element="value"). Examples of other syntaxes, including: HTML (the Web's Hypertext Markup Language format), RDF/XML (the Resource Description Framework using eXtensable Markup Language) and in plain XML can be found in syntax-specific documents available on the DCMI Website. Some are also referenced within this document and in the Bibliography Section of this guide.

HTML provides an easily understood format for demonstrating Dublin Core's underlying concepts, but more complex applications using qualification may find that using XML or RDF makes more sense. When considering an appropriate syntax, it is important to note that Dublin Core concepts are equally applicable to virtually any file format, as long as the metadata is in a form suitable for interpretation both by search engines and by human beings.

2.1. HTML

HTML can be used to express either simple or qualified Dublin Core, although there are limitations inherent in representing refinements in HTML. Specific instructions for expressing Dublin Core in HTML can be found in the following two DCMI documents:

1. Encoding Dublin Core Metadata in HTML
2. Expressing Qualified Dublin Core in HTML/XHTML meta elements

2.2. RDF/XML

RDF (Resource Description Framework) allows multiple metadata schemes to be read by humans as well as parsed by machines. It uses XML (EXtensible Markup Language) to express structure thereby allowing metadata communities to define the actual semantics. This decentralized approach recognizes that no one scheme is appropriate for all situations, and further that schemes need a linking mechanism independent of a central authority to aid description, identification, understanding, usability, and/or exchange.

RDF allows multiple objects to be described without specifying the detail required. The underlying glue, XML, simply requires that all namespaces be defined and once defined, they can be used to the extent needed by the provider of the metadata.

For example:

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:dc="http://purl.org/dc/elements/1.1/">

   <rdf:Description rdf:about="http://media.example.com/
audio/guide.ra">

      <dc:creator>Rose Bush</dc:creator>
      <dc:title>A Guide to Growing Roses</dc:title>
      <dc:description>Describes process for planting and
nurturing different kinds of rose bushes.</dc:description>
      <dc:date>2001-01-20</dc:date>

   </rdf:Description>
</rdf:RDF>
```

This simple example uses Dublin Core by itself to describe an audio recording of a guide to growing rose bushes. With XML or RDF/XML, Dublin Core can now be mixed with other metadata vocabularies. For example, the simple Dublin Core description above might be used alongside other vocabularies such as vCard that can describe the author's affiliation and contact information, or a more specialized "rose description" vocabulary that described the rose bushes in greater detail.

DCMI has made available several recommendations specifically about using these syntaxes:

1. Guidelines for Implementing Dublin Core in XML
2. Expressing Simple Dublin Core in RDF/XML
3. Expressing Qualified Dublin Core in RDF/XML (Proposed

Recommendation)

2.3. Metadata Storage and Maintenance Issues

Some implementations using Dublin Core have chosen to embed their metadata within the resource itself. This approach is taken most often with documents encoded using HTML, but is also sometimes possible with other kinds of documents. Simple tools have been developed to make provision of Dublin Core metadata within HTML encoded pages fairly easy. One such tool, DC.dot, extracts metadata information from an HTML document, and formats it so that it can be edited, then cut and pasted back into the HTML header of the original document.

On the other hand, metadata can be stored in any kind of database, and provide a link to the described resource rather than be embedded within it. This approach is likely to be most practical for many non-textual resources, and is increasingly used for text as well, primarily to support easier maintenance and sharing of metadata.

Each of these approaches have their advantages and disadvantages, and the balance point changes as implementations become larger and more diverse, and also as the metadata ages over time.

3. Element Content and Controlled Vocabularies

Each Dublin Core element is optional and repeatable, and there is no defined order of elements. The ordering of multiple occurrences of the same element (e.g., Creator) may have a significance intended by the provider, but ordering is not guaranteed to be preserved in every user environment. Ordering or sequencing may be syntax dependent, for instance, RDF/XML supports ordering, but HTML does not.

Content data for some elements may be selected from a "controlled vocabulary," which is a limited set of consistently used and carefully defined terms. This can dramatically improve search results because computers are good at matching words character by character but weak at understanding the way people refer to one concept using different words, i.e. synonyms. Without basic terminology control, inconsistent or incorrect metadata can profoundly degrade the quality of search results. For example, without a controlled vocabulary, "candy" and "sweet" might be used to refer to the same concept. Controlled vocabularies may also reduce the likelihood of spelling errors when recording metadata.

One cost of a controlled vocabulary is the necessity for an administrative body to review, update and disseminate the vocabulary. For example, the US Library of Congress Subject Headings (LCSH) and the US National Library of Medicine Medical Subject Headings

(MeSH) are formal vocabularies, indispensable for searching rigorously cataloged collections. However, both require significant support organizations. Another cost is having to train searchers and creators of metadata so that they know when using MeSH, for example, to enter "myocardial infarction" instead of the more colloquial "heart attack." More sophisticated implementations can make such tasks much easier for users, but the controlled vocabulary terms must be present for them to be able to accomplish that task.

Using controlled vocabularies can be done most effectively using qualifiers. Without an encoding scheme specifically designated, a subject which might very well be carefully selected from a controlled vocabulary cannot be distinguished from a simple keyword.

4. The Elements

5. Dublin Core Qualifiers

6. Glossary

7. Bibliography

Title:	**DCMI Glossary**
Creator:	Mary S. Woodley
Contributor:	Gail Clement
Contributor:	Pete Winn
Date Issued:	2003-09-15
Identifier:	http://dublincore.org/documents/2003/08/26/usageguide/glossary.shtml
Replaces:	http://dublincore.org/documents/2001/04/12/usageguide/glossary.shtml
Is Replaced By:	Not applicable
Is Part Of:	http://dublincore.org/documents/2003/08/26/usageguide/
Latest version:	http://dublincore.org/documents/usageguide/glossary.shtml
Status of Document:	DCMI Recommended Resource

The DCMI Glossary is a collaborative effort of the User Guide Committee with special thanks to Gail Clement & Pete Winn, whose original glossary was a basis for this version. Terms included in this glossary are based on DCMI documents, presentations at DC conferences, and discussions on the DC General listserv. We welcome comments and feedback regarding additions, deletions or changes to the terms and/or definitions found below.

The glossary was last updated on 10 September 2003.

A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z

1:1 principle

The principle whereby related but conceptually different entities, for example a painting and a digital image of the painting, are described by separate metadata records

A

AACR2
: See Anglo-American Cataloguing Rules

administrative metadata
: Metadata used in managing and administering information resources, e.g., location or donor information. Includes rights and access information, data on the creation and preservation of the digital object.

Anglo-American Cataloguing Rules (AACR2)
: The dominant bibliographic standard regulating cataloging in the English-speaking world. AACR2 represents a set of rules for the standard description of and access to all materials which a library holds or to which it has access.

American Standard Code for Information Interchange (ASCII)
: A scheme that provides standard numeric values to represent letters, numbers, punctuation marks and other characters. The use of standard values allows computers and computer programs to exchange data.

application profile
: In DCMI usage, an application profile is a declaration of the metadata terms an organization, information resource, application, or user community uses in its metadata. In a broader sense, it includes the set of metadata elements, policies, and guidelines defined for a particular application or implementation. The elements may be from one or more element sets, thus allowing a given application to meet its functional requirements by using metadata elements from several element sets including locally defined sets. For example, a given application might choose a specific subset of the Dublin Core elements that meets its needs, or may include elements from the Dublin Core, another element set, and several locally defined elements, all combined in a single schema. An application profile is not considered complete without documentation that defines the policies and best practices appropriate to the application.

Appropriate values
: Best practice for a particular Element or Qualifier may vary by context. Definitions may provide some guidance; other information may be found in *"Using Dublin Core"*.

ASCII
: See American Standard Code for Information Interchange

Author
: See Creator

authority control
A set of rules or procedures that assist in the maintenance of consistent forms of names or terms within a database.

authority file
A collection of authority records.

authority record
A record that registers the preferred form of a personal or corporate name, geographic region or subject term. It may indicate variant forms of the established heading, biographical or cultural information associated with the heading, as well as related headings.

B

Basic Semantics Register
An ISO Standard ISO/TS 16668:2000 which identifies and defines semantic components for use in data exchange.

best practice
Guidance and documentation to describe and standardize the use of metadata elements that best support a community's needs.

BSR
See Basic Semantics Register

C

case-sensitive
Lower and upper case letters are not treated as if they were interchangeable; e.g. 'a' is not the same as 'A'.

classification
A logical scheme for arrangement of knowledge, usually by subject. Classification schema are alpha and/or numeric; for example, Library of Congress Classification, Dewey Classification, Universal Decimal Classification.

controlled vocabulary
A prescribed set of consistently used and carefully defined terms.

Contributor
The Dublin Core element used to designate the entity responsible for making contributions to the content of the resource. Examples of a Contributor include a person, an organization or a service. Typically, the name of a Contributor should be used to

indicate the entity. See also *"Using Dublin Core".*

Coverage
: The Dublin Core element used to designate the extent or scope of the content of the resource. Coverage will typically include spatial location (a place name or geographic co-ordinates), temporal period (a period label, date, or date range) or jurisdiction (such as a named administrative entity). Recommended best practice is to select a value from a controlled vocabulary, and that, where appropriate, named places or time periods be used in preference to numeric identifiers such as sets of co-ordinates or date ranges. See also *"Using Dublin Core"*

Creator
: The Dublin Core element used to designate the entity primarily responsible for making the content of the resource. Examples of a Creator include a person, an organization, or a service. See also *"Using Dublin Core".*

Crosswalk
: A table that maps the relationships and equivalencies between two or more metadata schemes. Crosswalks or metadata mapping support the ability of search engines to search effectively across heterogeneous databases.

D

Date
: The Dublin Core element used to designate the date associated with an event in the life cycle of the resource. Typically, Date will be associated with the creation or availability of the resource. See also *"Using Dublin Core".*

DCMES
: Dublin Core Metadata Element Set. See Dublin Core.

DCMI
: See Dublin Core Metadata Initiative

DCMI recommendation
: A DCMI recommendation is a human-readable document that may define one or more DCMI terms

DCMI term
: A DCMI term is a DCMI element, a DCMI qualifier or term from a DCMI-maintained controlled vocabulary. Each DCMI term is defined in a DCMI recommendation and is identified by a Uniform Resource Identifier (URI) within a DCMI namespace.

DCMI term declaration
: A DCMI term declaration is the machine-processable representation of one or more

DCMI terms, expressed in a schema language.

Description
: The Dublin Core element used to designate a textual description of the content of the resource. See also *"Using Dublin Core".*

DCSV
: See Dublin Core Structured Value

descriptive metadata
: Metadata that supports the discovery of an object.

digital tourist
: An inexperienced searcher in the digital environment who does not possess knowledge of community- specific vocabularies. The Dublin Core provides a rudimentary vocabulary, or "pidgin language" for information discovery when exploring new digital territories. Coined by Ricky Erway at the Metadata Workshop on Metadata for Networked Images, September 24-25, 1996.

discovery software
: A computer application designed to simplify, assist and expedite the process of finding information resources.

Digital Object Identifier
: DOI was developed by the International DOI Foundation as a system for identifying and exchanging intellectual property in the digital environment. It provides a mechanism to link a searcher to digital content and facilitates copyright management.

Document Type Definition (DTD)
: In SGML or XML, a formal description of the components of a specific document or class of documents. DTDs provide a formal grammar used for machine processing (parsing) of documents expressed in SGML or XML. A DTD description includes:

- The containers or elements that make up the document; e.g., paragraphs, headings, list items, figures, tables, etc.
- The logical structure of the document; e.g., chapters containing sections, etc.
- Additional information associated with elements (known as attributes); e.g., identifiers, date stamps, etc.

document-like object (DLO)
: Originally defined as an entity that resembles a document from the standpoint that it is substantially text-based and shares other properties of a document; e.g., electronic mail messages or spreadsheets. The definition was expanded at the 3rd DC workshop to refer to any discrete information resource that are characterized by being fixed (i.e., having identical content for each user). Examples include text, images, movies, and performances.

DOI
: see Digital Object Identifier

dot.syntax
: A mechanism for refining the meaning of the element in HTML; for example, <META NAME="DC.Title.Alternative" CONTENT="*Title*">

DTD
: See Document Type Definition

Dublin Core
: The Dublin Core is a metadata element set. It includes all DCMI terms (that is, refinements, encoding schemes, and controlled vocabulary terms) intended to facilitate discovery of resources. The Dublin Core has been in development since 1995 through a series of focused invitational workshops that gather experts from the library world, the networking and digital library research communities, and a variety of content specialties. See the Dublin Core Web Site for additional information.

Dublin Core Metadata Initiative
: The Dublin Core Metadata Initiative is the body responsible for the ongoing maintenance of Dublin Core. DCMI is currently hosted by the OCLC Online Computer Library Center, Inc., a not-for-profit international library consortium. The work of DCMI is done by contributors from many institutions in many countries. DCMI is organized into working groups to address particular problems and tasks. DCMI working groups are open to all interested parties. Instructions for joining can be found at the DCMI web site under Working Groups (http://dublincore.org/groups/)

Dublin Core Structured Values
: DCSV recognizes two types of substrings: labels and values. A label is the name of the type of a value, and a value is the data itself. A value that is comprised of components, i.e. a value which has its own label and value, is called a structured value. Punctuation supports the parsing of the DCSV.

Dublin Core Terms
: See DCMI term

Dumb-down Principle
: The qualification of Dublin Core Elements is guided by a rule known colloquially as the Dumb-Down Principle. According to this rule, a client should be able to ignore any qualifier and use the value as if it were unqualified. While this may result in some loss of specificity, the remaining term value (minus the qualifier) must continue to be generally correct and useful for discovery. Qualification is therefore supposed only to refine, not extend the semantic scope of an Element.

E

EAD
: see Encoded Archival Description

electronic information resource
: An information resource that is maintained in electronic, or computerized format, and may be accessed, searched and retrieved via electronic networks or other electronic data processing technologies (e.g., CD-ROM)

element
: An element is a property of a resource. As intended here, "properties" are attributes of resources -- characteristics of a resource, such as a Title, Publisher, or Subject. Elements are formally defined terms which are used to describe attributes and properties of a resource.

element refinement (qualifier)
: Qualifiers make the meaning of an element narrower or more specific. An element refinement is a property of a resource which shares the meaning of a particular DCMI Element but with narrower semantics. In some application environments (notably HTML-based encodings), Element refinements are used together with elements in the manner of natural-language "qualifiers" (i.e., adjectives) . However, since element refinements are properties of a resource (like elements), element refinements can alternatively be used in metadata records independently of the properties they refine. In DCMI practice, an Element refinement refines just one parent DCMI property.

embedded metadata
: Metadata that is maintained and stored within the object it describes; the opposite of stand-alone metadata.

Encoded Archival Description
: An SGML DTD that represents a highly structured way to create digital finding aids for a grouping of archival or manuscript materials. The standard is maintained in the Network Development and MARC Standards Office of the Library of Congress (LC) in partnership with the Society of American Archivists. For more information see http://lcweb.loc.gov/ead/.

encoding scheme
: . An encoding scheme provides contextual information or parsing rules that aid in the interpretation of a term value. Such contextual information may take the form of controlled vocabularies, formal notations, or parsing rules. If an encoding scheme is not understood by a client or agent, the value may still be useful to a human reader. There are two types of encoding schemes: Vocabulary Encoding Schemes and Syntax Encoding Schemes

extensible
Having the potential to be expanded in scope, area or size. In the case of Dublin Core, the ability to extend a core set of metadata with additional elements.

Extensible Markup Language (XML)
A subset of Standard Generalized Markup Language (SGML), a widely used international text processing standard. XML is being designed to bring the power and flexibility of generic SGML to the Web, while maintaining interoperability with full SGML and HTML. For more information, see http://www.w3.org/XML/

F

Format
The Dublin Core element used to designate the physical or digital manifestation of the resource. See also *"Using Dublin Core"*..

G

GIF
See Graphics Interchange Format

GILS
See Global Information Locator Service

glossary
An alphabetized list of terms with definitions often created by an organization to reflect its needs. Normally lacks hierarchical arrangement or cross references. Also known as a term list.

Global Information Locator Service (GILS)
GILS embraces open standards to implement interoperable searching across diverse, decentralized information 'locators' to return references to all kinds of electronic and non-electronic information resources. Locators are implemented as common semantics for characterizing information resources, i.e. common metadata semantics. Formally known as Government Information Locator Service.

Graphics Interchange Format (GIF)
The dominant graphics format on the Web, limited to 256 colors. GIFs provide sharper black & white images than JPEGs.

granularity
The level of detail at which an information object or resource is viewed or described.

H

Harvester
A harvester is a client application that issues OAI-PMH requests. A harvester is operated by a service provider as a means of collecting metadata from repositories. (http://www.openarchives.org/OAI/openarchivesprotocol.html#harvester)

HTML
See Hypertext Markup Language

Hypertext Markup Language (HTML)
The standard text-formatting language for documents on the World Wide Web. HTML text files contain content that is rendered on a computer screen and markup, or tags, that can be used to tell the computer how to format that content. HTML tags can also be used to encode metadata and to tell the computer how to respond to certain user actions, such as a mouse click. For more information, see http://www.w3.org/MarkUp/.

I

Identifier
The Dublin Core element that is an unambiguous reference to the resource within a given context. Recommended best practice is to identify the resource by means of a string or number conforming to a formal identification system. See also *"Using Dublin Core".*

IETF
See Internet Engineering Task Force

IMT
See Internet Media Type

indexing
The process of evaluating information entities and creating terms that aid in finding and accessing the entity. Index terms may be in natural language or controlled vocabulary or a classification notation.

indexing program
Computer software used to order things; frequently used to refer to software that alphabetizes some or all of the terms in one or more electronic documents.

information resource
Any entity, electronic or otherwise, capable of conveying or supporting intelligence or knowledge; e.g. a book, a letter, a picture, a sculpture, a database, a person. See also DLO

instantiation

An identifiable occurrence or occasion of something; in the case of Dublin Core, a specific occurrence of an information resource.

International Organization for Standardization

ISO was established in 1947 as a worldwide federation of national standards bodies from some 130 countries.

Internet Commons

The global Internet environment, collection of information-bearing repositories whose data can be accessed through the Internet.

Internet Engineering Task Force (IETF)

The IETF is responsible for solving short-term engineering needs of the Internet. It has over 40 Working Groups.

Internet Media Type (IMT)

A set of terms that describe types of resources on the Internet. Used as an encoding scheme for the Format element in Dublin Core. http://www.isi.edu/in-notes/iana/assignments/media-types/media-types

interoperability

The ability of different types of computers, networks, operating systems, and applications to work together effectively, without prior communication, in order to exchange information in a useful and meaningful manner. There are three aspects of interoperability: semantic, structural and syntactical.

Interoperability Qualifiers

Additional metadata used either to refine the semantics of a Dublin Core metadata element's value, or to provide more information about the encoding scheme used for the value.

ISO

See International Organization for Standardization

J

Joint Photographic Experts Group (JPEG)

A standard for compressing digital images. The advantage of JPEG is that it uses compression to make graphics files smaller, making them faster to transfer and view over the World Wide Web. More than 16 million color hues are available. Better than GIF for color photographs. The disadvantage is some loss of image quality due to data loss during compression. For more information see http://www.jpeg.org/

JPEGs

See Joint Photographic Experts Group

K

Keywords
See Subject

L

Language
The Dublin Core element used to designate the language of the intellectual content of the resource. Recommended best practice for the values of the Language element is defined by RFC 3066. See also *"Using Dublin Core".*

literal
A literal or "appropriate literal" is the value of any given metadata entity that can be either a hyperlink or a string value (literal). A literal affords a great deal of flexibility and power, but increases complexity. Metadata should as well include an appropriate literal that reflects the base value of the metadata entity. For example, in these fragments: creator = "Public, John Q." creator = " http://authority.org/public-john-q-1234" the first has a value expressed as an appropriate literal whereas the second has a (hypothetical) link to an authority structure. It is not entirely clear what a person or application will find at the end of the link, so the metadata should contain an appropriate literal for simple discovery purposes.

M

mapping metadata
See crosswalk

MARC
Machine-Readable Cataloging Record. The MARC formats are standards for the representation and communication of bibliographic and related information (authority, holdings, classification, community information) in machine-readable form. MARC 21 grew out of the harmonization of USMARC and CAN/MARC, formerally national standards, and has emerged as an international standard. MARC21 is an implementation of the American National Standard, Information Interchange Format (ANSI Z39.2) and its international counterpart, Format for Information Exchange (ISO 2709). UniMARC was originally designed for conversion between national formats but now has been adopted by some countries as their national standard.

META tag
The HTML element used to demarcate metadata on a Web page. <META> </META>.

metadata

In general, "data about data;" functionally, "structured data about data." Metadata includes data associated with either an information system or an information object for purposes of description, administration, legal requirements, technical functionality, use and usage, and preservation. . In the case of Dublin Core, information that expresses the intellectual content, intellectual property and/or instantiation characteristics of an information resource. See Section 1.1 of this guide. For a history of the term See Caplan,pp. 1-3.

metadata record

A syntactically correct representation of the descriptive information (metadata) for an information resource. In the case of Dublin Core, a representation of the Dublin Core elements that has been defined for the resource. The majority of metadata records and record fragments in this document are presented in HTML syntax.

metadata schema registry

A publicly accessible system that records the semantics, structure and interchange formats of any type of metadata. A formal authority, or agency, maintains and manages the development and evolution of a metadata registry. The authority is responsible for policies pertaining to registry contents and operation. See also http://www.dlib.org/dlib/may02/wagner/05wagner.html

MIME

See Multipurpose Internet Mail Extensions

Multipurpose Internet Mail Extensions

The standard for attaching files to Internet e-mail messages. Attached files may be text, graphics, spreadsheets, documents, sound files, etc.

N

National Information Standards Organization

NISO, accredited by ANSI, develops and promotes technical standards used in a wide variety of information services.

namespace

A DCMI namespace is a collection of DCMI terms. Each DCMI namespace is identified by a URI. An XML namespace [XML-NAMES] is a collection of names, identified by a URI reference [RFC2396], that are used in XML documents as element types and attribute names. The use of XML namespaces to uniquely identify metadata terms allows those terms to be unambiguously used across applications, promoting the possibility of shared semantics. DCMI adopts this mechanism for the identification of all DCMI terms. For example, the namespace for Dublin Core elements and qualifiers would be expressed respectively in XML as:

xmlns:dc = "http://purl.org/dc/elements/1.1/
xmlns:dcterms = "http://purl.org/dc/terms/

The use of namespaces allows the definition of an element to be unambiguously identified with a URI, even though the label "title" alone might occur in many metadata sets. In more general terms, one can think of any closed set of names as a namespace. Thus, a controlled vocabulary such as the Library of Congress Subject Headings, a set of metadata elements such as DC, or the set of all URLs in a given domain can be thought of as a namespace that is managed by the authority that is in charge of that particular set of terms.

networked resource
An object that is available electronically via a network.

NISO
See National Information Standards Organization

O

OAI
See Open Archives Initiative

OAI-PMH
See Open Archives Initiative Protocol for Metadata Harvesting

OCLC
See Online Computer Library Center

Online Computer Library Center (OCLC)
The major source of cataloging data for libraries around the world; located in Dublin, Ohio, US. For more information, see http://www.oclc.org/home/.

Open Archives Initiative
"Develops and promotes interoperability standards that aim to facilitate the efficient dissemination of content. The Open Archives Initiative has its roots in an effort to enhance access to e-print archives as a means of increasing the availability of scholarly communication" Fro more information see http://www.openarchives.org/organization/index.html.

OpenURL
A method for describing resources and associated resources that are referenced in a network environment. It defines the mentods for transporting these descriptions between networked systems. NISO standard NISO Z39.88-2003 (still in draft). The Standard has been issued in two parts and it available for comment through California Institute of

Technology http://library.caltech.edu/openurl/Public_Comments.htm

Open Archives Initiative Protocol for Metadata Harvesting

The Protocol "provides an application-independent interoperability framework based on metadata harvesting. There are two classes of participants in the OAI-PMH framework: Data Providers administer systems that support the OAI-PMH as a means of exposing metadata; and Service Providers use metadata harvested via the OAI-PMH as a basis for building value-added services. " For more information see http://www.openarchives.org/organization/index.html.

P

parsing

Parsing may be divided into parts: lexical analysis and semantic parsing. Lexical analysis divides strings into components based on punctuation or tagging. Semantic parsing then attempts to determine the meaning of the string.

Persistent Uniform Resource Locator

An approach to the URL permanence problem proposed by OCLC. A PURL is a public alias for a document. A PURL remains stable, while the document's background URL will change as it is managed (e.g. moved) over time. A PURL is created by a Web administrator who is registered as a PURL "owner" and who maintains a mapping of the PURL to a current and functioning URL. A PURL is a form of URN.

Property

A property is a specific aspect, characteristic, attribute, or relation used to describe a resource. Dublin Core metadata elements are properties http://dublincore.org/documents/2003/04/02/dc-xml-guidelines/

Publisher

The Dublin Core element used to designate the entity responsible for making the resource available. Examples of a Publisher include a person, an organization, or a service. Typically, the name of a Publisher should be used to indicate the entity. See also *"Using Dublin Core".*

PURL

See Persistent Uniform Resource Locator

Q

qualifier

"Qualifiers" is the generic heading traditionally used for terms now usually referred to specifically as Element Refinements or Encoding Schemes. A qualifier must follow the

Dumb-Down Principle. There are two broad categories of qualifiers: Encoding schema and Element refinement.

Qualified Dublin Core
Qualified Dublin Core includes an additional element, Audience, as well as a group of element refinements (also called qualifiers) that refine the semantics of the elements in ways that may be useful in resource discovery

R

RDF
See Resource Description Framework.

RDF Site Summary
RSS was created and popularized by Netscape for their personalized portal site. Rich Site Summary (RSS) is a lightweight XML application designed to exchange headline metadata between news content providers and portals.

record
A record is some structured metadata about a resource, comprising one or more properties and their associated values. http://dublincore.org/documents/2003/04/02/dc-xml-guidelines/

registry
A system to provide management of metadata elements. See also metadata schema registry The DCMI Registry Working Group (WG) is the development of a metadata registry providing authoritative information regarding the DCMI vocabulary and the relationship between terms in that vocabulary.

Relation
The Dublin Core element used to designate A reference to a related resource. Recommended best practice is to reference the resource by means of a string or number conforming to a formal identification system. See also *"Using Dublin Core".*

Request for Comment (RFC)
A Request for Comment (RFC) is the process of establishing a standard on the Internet. Discussion of the proposed standard on the Internet is facilitated by the Internet Engineering Task Force (IETF). Once approved, the standard receives a unique number which identifies it; e.g., RFC See http://www.isi.edu/rfc-editor/. and http://www.ietf.org/rfc.html

resource
A resource is anything that has identity. Familiar examples include an electronic document, an image, a service (e.g., "today's weather report for Los Angeles"), and a

collection of other resources. Not all resources are network "retrievable"; e.g., human beings, corporations, and bound books in a library can also be considered resources. http://dublincore.org/documents/2003/04/02/dc-xml-guidelines/

Resource Description Framework (RDF)
The basic language for writing metadata; a foundation which provides a robust flexible architecture for processing metadata on the Internet. RDF will retain the capability to exchange metadata between application communities, while allowing each community to define and use the metadata that best serves their needs. For more information see http://www.w3.org/RDF/

resource discovery
The process through which one searches and retrieves an information resource.

Resource Type
See Type.

Resource Description
See Description.

Resource Identifier
See Identifier

RFC
See Request for Comment

Rights
The Dublin Core element used to provide a link to information about rights held in and over the resource. Typically a Rights element will contain a rights management statement for the resource, or reference a service providing such information. Rights information often encompasses Intellectual Property Rights (IPR), Copyright, and various Property Rights. If the rights element is absent, no assumptions can be made about the status of these and other rights with respect to the resource. See also *"Using Dublin Core".*

Rights Management
See Rights

ROADS
Resource Organisation And Discovery in Subject based services. A UK funded project whose aim is to develop discovery software for Internet resources.

RSS
See RDF Site Summary .

S

schema or scheme(plurals schemas or schemata; schemes)

In general terms, any organization, coding, outline or plan of concepts. In terms of metadata, a systematic, orderly combination of elements or terms. In terms of DCMI term declarations represented in XML or RDF schema language, schemas are machine-processable specifications which define the structure and syntax of metadata specifications in a formal schema language. In terms of an encoding scheme, is a set of rules for encoding information that supports a specific community of users. See also Encoding scheme.

scheme

See schema

search engine

A utility capable of returning references to relevant information resources in response to a query.

semantic interoperability

Ability to search for digital information across heterogeneous distributed databases whose metadata schemas have been mapped to one another. It is achieved through agreements about content description standards; for example, Dublin Core, Anglo-American Cataloging Rules.

Semantic Web

A term coined by Tim Berners-Lee which views the future Web as a web of data, like a global database. The infrastructure of the Semantic Web would allow machines as well as humans to make deductions and organize information. The architectural components include semantics (meaning of the elements), structure (organization of the elements), and syntax (communication). http://www.w3.org/DesignIssues/Semantic.html

semantics

Significance or meaning. In the case of Dublin Core, the significance or intended meaning of individual metadata elements and their components.

SGML

See Standard Generalized Markup Language

SICI

Serial Item and Contribution Identifier (ANSI/NISO Z39.56-1996 Vers. 2) A numeric notation to identify serial issues and articles uniquely regardless of their distribution medium (paper, electronic, microform).

Simple Dublin Core

The fifteen Dublin Core elements used without qualifiers, that is without element refinement or encoding schemes.

software agent
: A computer program that carries out tasks on behalf of another entity. Frequently used to reference a program that searches the Internet for information meeting the specified requirements of an individual user.

Source
: The Dublin Core element used to designate a reference to a resource from which the present resource is derived. The present resource may be derived from the Source resource in whole or part. Recommended best practice is to reference the resource by means of a string or number conforming to a formal identification system. See also *"Using Dublin Core".*

spatial
: An element refinement of Coverage. Spatial characteristics of the intellectual content of the resource.

Standard Generalized Markup Language (SGML)
: A non-proprietary language/enabling technology for describing information. Information in SGML is structured like a database, supporting rendering in and conversion between different formats. Both XML and later versions of HTML are instances of SGML. For more information see http://www.w3.org/MarkUp/SGML/.

stand-alone metadata
: Metadata that is created, maintained and stored independently of the object it describes. The opposite of embedded metadata.

structured value
: See Dublin Core Structured Value

structural interoperability
: Is achieved through data models for specifying semantic schemas in a way that they can be shared; for example, RDF.

structural metadata
: Structural metadata defines the digital object's internal organization and is needed for display and navigation of that object.

sub-element
: See element refinement

Subject
: The Dublin Core element used to describe the content of the resource. The element may use controlled vocabularies or keywords or phrases that describe the subject or content of the resource. See also *"Using Dublin Core".*

Subject Headings

An alphabetical list of words or phrases that represent a concept that is under authority control, e.g., the Library of Congress Subject Headings.

surrogate content
: Metadata as a substitute for an actual resource.

switching language
: A mediating language used to establish equivalencies among various indexing languages. Dublin Core has been viewed as a switching "language" between various metadata schemas.

syntactic interoperability
: Achieved by marking up our data in a similar fashion so we can share the data and so that our machines can understand and take the data apart in sensible ways; for example, XML, EAD and MARC.

syntax
: The form and structure with which metadata elements are combined. In the case of Dublin Core, the form and structure of how metadata elements and their components are combined to form a metadata record.

Syntax Encoding Schemes
: Syntax Encoding Schemes indicate that the value is a string formatted in accordance with a formal notation, such as "2000-01-01" as the standard expression of a date.

T

taxonomy
: In general terms, systematic classification according to principles or general laws. In digital terms, automated classification of documents in a hierarchy based on information gathered by a metacrawler. May refer to a classification of DCMI terms. A thesaurus is an example of a taxonomy.

technical metadata
: Metadata that documents the creation and the digital characteristics of the files.

TEI
: See Text Encoding Initiative

temporal
: An element refinement of coverage. Temporal characteristics of the intellectual content of the resource.

term
: See DCMI term

Text Encoding Initiative (TEI)

An international project to develop guidelines for the preparation and interchange of electronic texts for scholarly research as well as a broad range of other language industry uses. The TEI DTD is an SGML Document Type Definition for encoding literary works. For more information, seehttp://www.tei-c.org/

thesaurus

A structured vocabulary mae up of names, words, and other information, typically including synonyms and/or hierarchical relationships for the purpose of cross-referencing in order to organize a collection of concepts for reference and retrieval. See the ANSI/ NISO Standard for thesaurus construction Z39.19-2003 (R1998; ISO 2788). A controlled vocabulary of terms or concepts that are structured hierarchically (parent/child relationships) or as equivalences (synonyms), and related terms (associative). See also Subject headings and glossary. A thesaurus is a taxonomy.

Thesaurus of Geographic Names

The TGN is a controlled vocabulary containing around 1,000,000 names and other information about places. It includes physical features and administrative entities, such as cities and nations. The emphasis in TGN is on places important for art and architecture.

Title

The Dublin Core element used to designate the name given to the resource. Typically, a Title will be a name by which the resource is formally known. See also *"Using Dublin Core".*

tokens

The means to denote the status of an element or qualifier within a registry; e.g., proposed, recommended, conforming (to the namespace), obsolete, or local.

Type

The Dublin Core element used to designate the nature or genre of the content of the resource. Type includes terms describing general categories, functions, genres, or aggregation levels for content. Recommended best practice is to select a value from a controlled vocabulary. See also *"Using Dublin Core".*

U

ULAN

See Union List of Artist Names

Unicode

A universal encoding scheme designed to allow interchange, processing and display of the world's principal languages, as well as many historic and archaic scripts. Unicode supports and fosters a multilingual computing world community by allowing computers using one language to "talk" to computers using a different language. A registered trademark of Unicode, Inc.

Unicode Transformation Format, 8-bit (UTF-8)
A temporary form of Unicode that is well suited for routing data through systems that are not designed for Unicode, such as some email servers and Web clients. UTF-8 is an attractive way of storing multilingual data on the Internet, without requiring full Unicode compliance.

Uniform Resource Identifier (URI)
The syntax for all names/addresses that refer to resources on the World Wide Web. For information about Internet addressing, see http://www.w3.org/Addressing/Addressing.html.

Uniform Resource Locator (URL)
A technique for indicating the name and location of Internet resources. The URL specifies the name and type of the resource, as well as the computer, device and directory where the resource may be found. The URL for Dublin Core Metatdata Initiative http://dublincore.org/. For information about Internet addressing, see http://www.w3.org/Addressing/Addressing.html.

Uniform Resource Name (URN)
A URI (name and address of an object on the Internet) that has some assurance of persistence beyond that normally associated with an Internet domain or host name. For information about Internet addressing, see http://www.w3.org/Addressing/Addressing.html.

Union Lists of Artists' Names (ULAN)
Union List of Artist Names. A controlled vocabulary of artists' names and biographical and bibliographic information produced by the Getty Vocabulary Program.

URI
See Uniform Resource Identifier

URL
See Uniform Resource Locator

URN
See Uniform Resource Name

USMARC
See MARC

UTF-8
See Unicode Transformation Format, 8-bit.

V

value qualifier
Value qualifier refers to either an encoding rule or controlled vocabulary that aids in the

interpretation of the value within the metatag. See encoding scheme.

vCard
A standard for storing information about individuals or corporations; an electronic business card.
For more information, check the Internet Mail Consortium page on personal data exchange.

Vocabulary Encoding Schemes
Vocabulary Encoding Schemes indicate that the value is a term from a controlled vocabulary, such as the value "China - History" from the Library of Congress Subject Headings.

Vocabulary Terms
The Usage Board maintains the DCMI Type Vocabulary [7] -- a general, cross-domain list of recommended terms that may be used as values for the Resource Type element to identify the genre of a resource. The member terms of the DCMI Type Vocabulary are called Vocabulary Terms.

W

Warwick Framework
An architecture for the interchange of metadata packages, or "containers"; designed to satisfy the need for competing, overlapping, and complementary metadata models. For more information, see http://www.dlib.org/dlib/july96/07weibel.html.

World Wide Web (WWW)
The panoply of Internet resources (text, graphics, audio, video, etc.) that are accessible via a Web browser.

World Wide Web Consortium (W3C)
An international industry consortium founded in October 1994 to lead the World Wide Web to its full potential by developing common protocols that promote its evolution and ensure its interoperability. For additional information see http://www.w3.org/Consortium/.

WWW
See World Wide Web

W3C
See World Wide Web Consortium

X

XML

See Extensible Markup Language

Y

Z

Z39.50
A NISO standard for an application layer protocol for information retrieval which is specifically designed to aid retrieval from distributed servers. http://lcweb.loc.gov/z3950/agency

Dublin Core Projects

Related Metadata Standards

Acknowledgements

Many sources were consulted for the creation of this glossary:

BIBLINK: Objectives, Scope and Glossary

Clement, Gail and Peter Winn. A user guide for simple Dublin Core: glossary (draft). Last updated 05/12/99.

Baca, Murtha, ed. Introduction to metadata: glossary. Version 2.0 http://www.getty.edu/research/institute/standards/intrometadata/index.html

Caplan, Priscilla. *Metadata Fundamentals for all Librarians.* Chicago: American Library Association, 2003.

Lanzi, Elisa. Introduction to vocabularies: enhancing access to cultural heritage information. Los Angeles: Getty Information Institute, 1998. Updated by Patricia Harpring, 2000. http://www.getty.edu/research/institute/vocabulary/introvocabs/

Moen, William. An Overview of Z39.50, Supplemented by a Case Study of Implementing the Zebra Server Under the Linux Operating System http://www.unt.edu/wmoen/Z3950/GIZMO/appendix_d.htm

Schemas glossary http://www.schemas-forum.org/related/glossary.html

Smith, Allison. Terms commonly used in authority control and thesaurus construction. Word document provided to DC-general listserv.

Other useful glossaries

Digital Library Initiative at the Univerisity of Illinois at Urbana-Champaign. http://dli.grainger.uiuc.edu/glossary.htm

UKOLN Glossary http://www.ukoln.ac.uk/metadata/glossary/

National Library of Canada. A Glossary of Digital Library Standards, Protocols and Formats. http://www.nlc-bnc.ca/pubs/netnotes/notes54.htm

The online edition of Digital Libraries, by William Arms, (c) 2000 MIT Press, updated with additional material by the author. Glossary

Web Thesaurus Compendium. Provides listings of thesaurai by alphabetical order and subject. Has links to related literature and software for building thesaurai. http://www.darmstadt.gmd.de/~lutes/theslit.html

---

Title:	**Guidelines for registration of vocabulary and encoding scheme qualifiers**
Creator:	Traugott Koch
Identifier:	http://dublincore.org/usage/documents/2003/05/15/vocabulary-guidelines/
Date Issued:	2003-05-15
Latest Version:	http://dublincore.org/usage/documents/vocabulary-guidelines/
Replaces:	http://dublincore.org/documents/2002/12/02/vocabulary-guidelines/
Replaced by:	Not applicable
Document Status:	This is a DCMI Usage Board Working Draft.
Description:	This document describes processes related to the submission and review of proposed encoding schemes for identifying controlled vocabularies of values.
Date Valid:	2003-05-15

DCMI recognizes that different discourse and practice communities have legitimate, particular needs to be able to select either vocabulary schemes from an array of recognized controlled vocabularies (e.g., thesauri, classification systems, taxonomies, ontologies, and word lists) or encoding schemes that determine the syntactic structure of the values (e.g., date encoding schemes). To promote the greatest degree of interoperablity, DCMI encourages the registration of recognized schemes with DCMI.

DCMI recognizes that in order to promote interoperability through the common assignment of terms from established, publicly recognized controlled vocabularies and encoding schemes, the most critical, immediate need is to provide registration mechanisms for vocabularies for the “Subject” vocabulary.

1. General

- 1.1 DCMI does not approve vocabulary schemes, but acknowledges formally maintained schemes as suitable for use with DC metadata. Thus, the schemes have the status "Registered" which does not necessarily imply that they are recommended by the DCMI.

- 1.2 DCMI maintained schemes have the status "Recommended".

2. Registration process

- 2.1 Anyone can propose a scheme for registration by submitting the required information to the web form at: http://wip.dublincore.org:8080/schemes/submitServlet . Institutions who maintain a stable registry containing their own vocabularies in a considerable number can agree with the DCMI Usage Board on a simplified submission process.

- 2.2 The DCMI Usage Board applies a "fast track" decision process involving the guidelines and criteria listed below.

- 2.3 For each scheme, the registrant should provide the following information:
    - Full label of the scheme
    - Suggested name (acronym)
    - Maintenance agency
    - Maintenance agency contact person
    - Maintenance agency contact email address
    - Submitter email address (if different from the maintenance agency contact email address)
    - Online access point (URL if applicable)
    - Access information (URL or physical address)
    - Additional information about the scheme
    - Domain(s) and extent of usage
    - Associated element(s) or element refinement(s)

Example

Full label of the scheme	Dewey Decimal Classification
Suggested name (acronym)	DDC
Maintenance agency	OCLC Forest Press
Maintenance agency contact person name	{ Name of current editor or contact person }
Maintenance agency contact email address	dewey@oclc.org
Submitter email address	{ Email address of submitter if different than the maintenance agency }

Online access point	Web Dewey in CORC (http://purl.oclc.org/corc/)
Access information	http://www.oclc.org/fp/products/index.htm
Additional information about the scheme	License required
Domain(s) and extent of usage	Most frequently used universal classification system for library OPACs and national bibliographies; limited recent usage in web catalogues etc.
Associated element(s) or element refinement(s)	Subject

- 2.4 Schemes that adhere to the guidelines below will be given the status "Registered" and are included in the DCMI Registry. The registration tool contains schemes with the status "Registered" or "Rejected". The scheme will be given an URI of the form: http://purl.org/dc/terms/(Acronym)

3. Guidelines

- 3.1 **Kind of schemes to be registered**

  - 3.1.1 Only schemes which are created and maintained by recognized entities and properly published may be registered.

- 3.2 **Labeling of the schemes**

  - 3.2.1 The scheme label should be the official title the vocabulary is known under. The name of the organization maintaining or owning the scheme is rarely sufficient since it does not unambiguously stand for the vocabulary alone.

  - 3.2.2 The scheme labels and acronyms are only appropriate for an unchanged use of an official version of the scheme. Unofficial versions, modified versions, unofficial translations and similar should not use the official label or acronym but apply a local one (e.g. based on the service, project or provider. Ex.: The DutchESS service is using a local variant of the BC classification. It should be called DutchESSC or DutchESS-BC if it is really close to the official scheme).

- 3.2.3 A subset of an official scheme where terms are unaltered may not be registered separately.

- 3.3 **Acronyms to be used as DCMI qualifier names**

  - 3.3.1 The acronyms must be unique and start with an initial upper case. Every effort will be made to maintain the short name proposed by the maintaining agency. In case of collisions, a suitable alternative will be chosen in consultation with the maintaining agency.

  - 3.3.2 Existing official acronyms or short names should be used as acronyms.

  - 3.3.3 Official translated versions receive an acronym where a standard language code is added, e.g. DDC-fr. This is necessary since translated versions are rarely fully equivalent. Other translations, if registered, will be assigned an alternative acronym.

- 3.4 **Specification of scheme versions**

  - 3.4.1 DCMI will register multiple versions of schemes if they appear to be important and/or it is requested by a user applying for registration.

  - 3.4.2 Versioned schemes should be registered and used when there is a considerable probability that databases exist which apply terms and classes belonging to older versions of the scheme.

  - 3.4.3 The official version of the scheme used should be indicated like in the following examples: DDC21, DDC21ab-fr (abridged DDC version 21 in French), MSC2000.

Note: In order to stay in sync with DCMI conventions, in these guidelines the full name of a scheme is called "label" and an acronym or token is called "name".

Traugott Koch (Traugott.Koch@ub2.lu.se)

Created: 2001-05-11
Last update: 2003-02-20
URL: http://www.lub.lu.se/~traugott/drafts/vocab-guide5.html

Metadata associated with this resource: http://dublincore.org/usage/documents/vocabulary-guidelines/index.shtml.rdf

CORES Action Plan: MARC 21 and MODS metadata elements

[May 29, 2003, submitted by Rebecca Guenther as input to a D-Lib Magazine article on the CORES Resolution.]

The Library of Congress plans to explore the possibility of providing persistent URIs for its metadata elements. This will include, initially, "http:" URIs for MARC and MODS content designators (elements, subelements, etc). As a first step, this will take the form of a statement that details the convention for naming elements. LC will explore the use of URNs as element names as a more appropriate means to provide persistent identifiers for elements, particularly since naming, rather than retrieval, is the primary objective of a metadata element identifier, and resolution is not an issue. URIs may also be provided for values in controlled MARC/MODS value lists, also initially by publishing a naming convention, and perhaps later explicitly in mark-up (with the exception of the MARC relators as detailed below).

As a result of a request by the DCMI Usage Board to make available the terms in the *MARC Code List for Relators* for use with the Dublin Core Creator and Contributor elements, the Library of Congress will establish URIs for the values on this list as an initial test case. This will initially take a form compatible with DCMI (using RDF) however in the long term it may be replaced by a more comprehensive solution. This work has already been largely completed, although a few details need to be worked out.

Establishing URIs will enable an unambiguous reference to any MARC/MODS element.

1. Assignment of MARC/MODS URIs (short-term).

1.1. MARC

Because there are different MARC formats (i.e., bibliographic, authority, holdings, classification, and community information), it is necessary to identify both that the namespace is MARC 21 and what the format is.(Note that a namespace has been established for MARCXML.) URIs for MARC elements might be constructed as follows:

http://www.loc.gov/marc.[format].[fieldname].[subfield]

Bibliographic format (subfield):
http://www.loc.gov/marc.bibliographic.245.a
would be a URI for subfield $a of field 245 of the MARC bibliographic format.

Bibliographic format (field):
http://www.loc.gov/marc.bibliographic.245

Bibliographic format (indicator):
http://www.loc.gov/marc.bibliographic.245.i1

Bibliographic format (008 element):

http://www.loc.gov/marc.bibliographic.008-s-03
for sound recording 008 character position 03.

Authority format (008 element):
http://www.loc.gov/marc.authority.008

1.2. MARC Code lists

MARC Code List for Relators. Each term on the Relators list would be assigned a URI taking the following form:
http://www.loc.gov/marc.relators.adp
where the last element is the code that represents the term on the list. The above would be the URI for "adaptor".

At a later date, this pattern would be followed for the values on other code lists, where the URI would be constructed with the code attached to a namespace. An unresolved question will be establishing these for MARC language codes, since they are equivalent to ISO 639-2/B codes. This will be considered at a later date.

1.3. Source codes

There are many places in MARC 21 that use a code to identify a source, or in Dublin Core terms, an encoding scheme. The DCMI Library Application profile recommends registering these. Currently there are several encoding schemes that have been registered to be used with Dublin Core elements that also exist in the MARC namespace (examples: lcsh, lcc). If LC assigns URIs for each of these values the redundant namespace identification will need to be worked out (i.e., are they redundantly registered in the DCMI namespace or is the MARC 21 namespace referenced? What will happen with the current overlap?).

1.4. MODS elements

Some MODS elements are already used in the DCMI Library application profile, so URI assignment for these would be useful. Those that are referenced in DCMI-Lib AP are:
    Edition
    Location
The namespace for MODS is:
http://www.loc.gov/mods

The above could be specified as follows:
http://www.loc.gov/mods.originInfo.edition
(Note that edition is a subelement of originInfo)
http://www.loc.gov/mods.location
It needs to be considered whether a version number should be included in the URI.

MODS elements that are subelements could be assigned URIs as follows:

http://www.loc.gov/mods.titleInfo.partNumber

2. Assignment of MARC/MODS URIs (long-term)

LC plans to assign persistent URIs utilizing URN namespaces. It is currently investigating options for naming elements within this mechanism, rather than assigning http: URIs. Examples might be:

URN:[urn namespace id]:marc.bibliographic.245.a
URN:[urn namespace id]:mods.originInfo.edition
URN:[urn namespace id]:marc.relators.adp

3. Assignment of identifiers to metadata schemas

LC plans to register handles to be used as persistent names for XML schemas which it maintains. These handles will be expressed as URLs which resolve through LC's proxy server (http://hdl.loc.gov). For example:

http://hdl.loc.gov/loc.standards/mods
http://hdl.loc.gov/loc.standards/marc21.slim

4. Timeframe

LC plans to make the relator list available (with URIs for each entity) after the DCMI Usage Board meeting in June 2003. It should be possible to establish persistence policies and URI assignment policies by the second half of 2003, pending investigation of URN namespaces. Minimally, these will be made available as a tool for referencing MARC and MODS metadata elements. Assigning URIs to MODS elements will probably take less time than to all MARC elements. It is expected that assigning URIs may take a year or so.

Rebecca Guenther
Library of Congress
Network Development and MARC Standards Office

```
Title:   Excerpt from the RDF schema declaring MARC relator terms
Date:    2003-06-11
```

```
<?xml version='1.0' ?>
<rdf:RDF xmlns:marcrel="http://www.loc.gov/marc.relators"
         xmlns:dc="http://purl.org/dc/elements/1.1/"
         xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
         xmlns:dcterms="http://purl.org/dc/terms/"
         xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">

  <rdf:Description rdf:about="http://www.loc.gov/marc.relators">
    <dc:title xml:lang="en-US">MARC Relator Terms</dc:title>
    <dc:publisher xml:lang="en-US">Library of Congress</dc:publisher>
    <dc:description xml:lang="en-US">A list of relator terms and
their
    associated codes. The purpose of this list is to indicate the
relationship
    between the resource being described and a named person or
    corporate body.</dc:description>
    <dc:language xml:lang="en-US">English</dc:language>
    <dcterms:issued>2002-07-02</dcterms:issued>
    <dcterms:modified>2002-05-22</dcterms:modified>
    <dc:source rdf:resource="http://www.loc.gov/marc.relators"/>
  </rdf:Description>

  <rdf:Property rdf:about="http://www.loc.gov/marc.relators.act">
    <rdfs:label xml:lang="en-US">Actor</rdfs:label>
    <rdfs:comment xml:lang="en-US">A person who principally exhibits
    acting skills in a musical or dramatic presentation or
entertainment.</rdfs:comment>
    <dc:description xml:lang="en-US">Typically, the name of the Actor
    should be used to indicate the person.</dc:description>
    <rdfs:subPropertyOf rdf:resource="http://purl.org/dc/terms/
contributor"/>
    <rdfs:isDefinedBy rdf:resource="http://www.loc.gov/marc/
relators/"/>
    <dcterms:issued>2002-11-13</dcterms:issued>
  </rdf:Property>
  <rdf:Property rdf:about="http://www.loc.gov/marc.relators.arr">
    <rdfs:label xml:lang="en-US">Arranger</rdfs:label>
    <rdfs:comment xml:lang="en-US">A person who transcribes a musical
    composition, usually for a different medium from that of the
original;
    in an arrangement the musical substance remains essentially
unchanged.</rdfs:comment>
    <dc:description xml:lang="en-US">Typically, the name of the
Arranger
    should be used to indicate the person.</dc:description>
    <rdfs:subPropertyOf rdf:resource="http://purl.org/dc/terms/
contributor"/>
    <rdfs:isDefinedBy rdf:resource="http://www.loc.gov/marc/
relators/"/>
    <dcterms:issued>2002-11-13</dcterms:issued>
  </rdf:Property>

  <rdf:Property rdf:about="http://www.loc.gov/marc.relators.art">
    <rdfs:label xml:lang="en-US">Artist</rdfs:label>
    <rdfs:label xml:lang="en-US">Graphic technician</rdfs:label>
    <rdfs:comment xml:lang="en-US">A person (e.g., a painter) who
conceives, and
    perhaps also implements, an original graphic design or work of
art, if specific
    codes (e.g., [egr], [etr]) are not desired. For book
illustrators, prefer
    Illustrator [ill].</rdfs:comment>
    <dc:description xml:lang="en-US">Typically, the name of the
Artist should be
    used to indicate the person.</dc:description>
    <rdfs:subPropertyOf rdf:resource="http://purl.org/dc/terms/
contributor"/>
    <rdfs:isDefinedBy rdf:resource="http://www.loc.gov/marc/
relators/"/>
    <dcterms:issued>2002-11-13</dcterms:issued>
  </rdf:Property>

</rdf:RDF>
```

AskDCMI
Dublin Core Metadata Initiative

home | my questions | search archives | experts | help

Login

username:

password:

GO

Register

Other Links

DCMI Home

Categories:

Welcome

About DCMI
Administration ,Boards, Committees, Working Groups ,more...

DCMI Glossary

Implementation
Maintenance ,Metadata in General ,more...

Syntaxes
HTML ,Other ,more...

Terms
Application Profiles , Elements ,more...

Ask a Question

About AskDCMI | AskDCMI Policies

Report on the First Month of the DCMI Service

Diane I. Hillmann, Administrator
May 12, 2003

The new AskDCMI service passed its first anniversary last week, and has so far attracted more questions than anticipated. A total of 40 questions have been asked so far. The questions have been fairly well distributed amongst the available categories:

About DCMI	7
Implementation	8
Syntaxes	6
Terms	18

Most of the questions have been thoughtful and relevant, none so far have been frivolous, but a few have had to be deleted (errors, automated replies, etc.) and one was definitely off-topic.

We currently have 51 registered users and 11 registered experts. Most of the questions have been answered by three of those experts (Diane, Stuart, and Andy), so we clearly have a way to go before we can consider this a fully functioning service.

I see several areas where we need to focus some attention:

1.   Recruitment of experts

Eleven experts are not enough, and we definitely need more experts with enough relevant experience to answer questions on syntax. Recommendation: Stu and Makx make some pointed personal suggestions to folks on the Advisory and Usage Boards, plus others that may be vocal and articulate on the lists.

2.   More timely answers

The original time parameter for unanswered questions to be deemed "overdue" was 5 days. We have not regularly been making that deadline, which results in questions going into an "overdue" queue, which is not viewable by anyone other than the administrator. Aside from the question of whether this is good from the point of view of the software, it is not good for this service. There are currently no overdue questions, but that's a result of a good bit of arm twisting on my part.

3.   Training for experts

It is entirely possible that part of the problem with both (1) and (2) may be that experts feeling 'in-expert' on the system, to the point that they are not stepping up to the plate and claiming and answering questions. There are several things we could do to ameliorate this. First, I'd like to do a short session with the UB when they are in Ithaca in June (this could be done by me or Yvonne). Secondly, we could do a similar session at the Conferenceeither for the Advisory Board, for the AB and an invited group of potential experts, or for anyone who wanted to participate. Again, the trainers for this could be me, Yvonne, or Stuart.

4.   Giving feedback to the VRD

The software we're using is really designed more for use in an 'educational' context, where there is a broader disparity between askers and answerers. AskDCMI is operating in a slightly different context, one that is more 'professional' (though we hope not less educational for that). Stuart has said that the software will be revised to bring it into an 'open source' state by next year, which is an excellent opportunity for us to provide useful feedback to them. We've done very well so far, but I'd like to consider how we could do more. It'd be wonder to think that we could be starting a trend: consider how useful it would be to everyone if W3C could start up one of these? And IEEE/LOM? Having a network of these services could make them potentially much more useful, as questions could be referred between the services.

AskDCMI
Dublin Core Metadata Initiative

home | my questions | search archives | experts | help

Login

username:

password:

GO

Register

Other Links

DCMI Home

Tips for AskDCMI Experts

The following suggestions are intended to help experts who volunteer for the AskDCMI service.

Please Do:

- Maintain a tone that is friendly and/or professional in your responses to AskDCMI users
- Use the spell-check feature before submitting your response
- Refer to the work of projects already associated with the Dublin Core Metadata Initiative when possible
- Provide suggestions for resources or a partial answer whenever possible when requesting clarification of a user's question
- Point the user to specific resources relevant to their information request
- "Send to Admin with Comment" any questions that seem to you inappropriate for the AskDCMI service
- Encourage the user to continue their research
- Let an administrator know if you feel that a user is abusing the AskDCMI service in any way (see the AskDCMI Abuse of Service policy for more information)
- Provide the user with citations to any resources that you may use in responding to their question
- Conform with copyright and fair use policies in your responses
- Encourage the user to narrow or broaden their research topic, if appropriate
- Include helpful URLs, if possible, using the following format:

Brief Title of Web Site
Brief description about what the site contains, or instructions on how to locate the helpful information that you would like the user to find
URL of exact page (be careful when providing a URL from a site which uses frames)

- Feel free to edit your answer or include an addendum after submitting or posting

Please Do Not:

- Respond to any questions that you feel might result in illegal or dangerous activities - send these to the administrator instead (Claim the question, then use Send to Admin with Comment)
- Provide a definitive answer to a question that you feel might exceed your expertise - although pointers to resources that will assist the user in continuing their research are fine
- Discourage the user's curiosity, even if their question is extremely vague - instead, take the opportunity to suggest a way for the user to better focus their topic

Some Sample Responses

These responses are intended serve as good examples of the kinds of responses AskDCMI hopes to provide.

Using DC for Series

Thesauri vs. Classification

A few helpful reminders:

The email that a user receives containing your response will include information on how to create a citation to that response

AskDCMI exists to support DCMI's mission "promoting the widespread adoption of interoperable metadata standards and developing specialized metadata vocabularies for describing resources that enable more intelligent information discovery systems." Your participation in AskDCMI is very important and greatly appreciated!

About AskDCMI | AskDCMI Policies