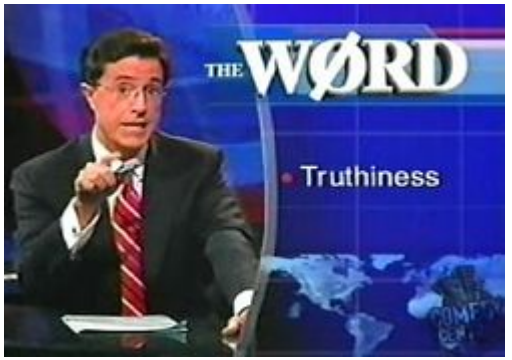
Persistent Identifier Principles and Practice

24 September 2008, Berlin

John Kunze, California Digital Library

Preservation and loss

Data preservation: storing objects while retaining a balance of usability and faithfulness to creators' original intent



Loss happens

- Hard loss: some or all the bits are missing
- Semantic loss: data not renderable/understandable
- Legal loss: data or its format is legally encumbered

CDL

Preservation and (im)persistent identifiers

No matter how well you preserve the bits

- if you lose access, you lose everything
- so, if access via URL or other identifier breaks, you have loss

Broken identifier

The persistent problem of persistent identifiers

But we *have* persistent identifiers: ISBN, LCCN, etc.

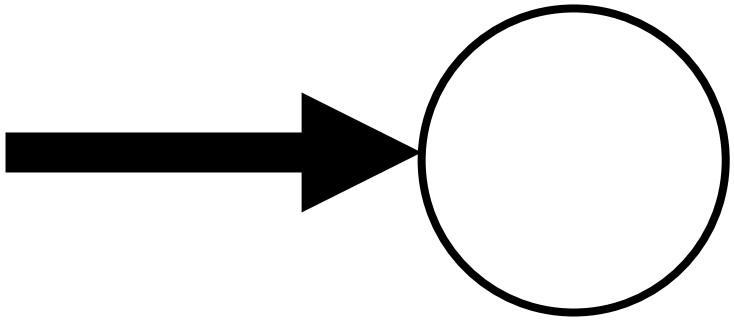
- Right, but we want *actionable* identifiers
- That is, we want URL-like ease of use

Identifier schemes claiming suitability for persistence

- ARK, DOI, Handle, INFO, ISSN, MD5, PURL, URI, URL, URN, UUID, and others
- Which id schemes help?
- Which id schemes hurt?

CDL

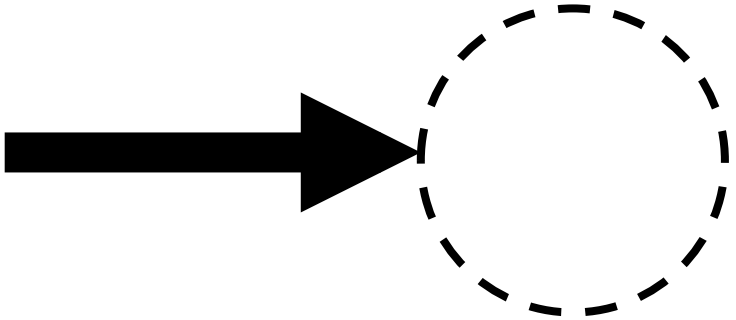
Imagine the perfect identifier



Imagine the perfect identifier *broken*

Identifier completely intact,

but object gone due to…



Common disasters

- Funding or political loss
- Power outage, disk failure, human error
- Objects that are *removed*, *replaced*, or *moved* without an HTTP redirect being set up

Real, but less common disasters

- War, social upheaval, flood, earthquake, etc

CDL

The persistent problem of persistent identifiers

Don’t we *have* persistent ids? (ISBN, LCCN, etc)

- Yes! But we really want *actionable* identifiers
  - That is: we want URL-like ease of use
  - Identifiers that are *not* embeddable in URLs have proven to be only of theoretical interest

All real work focuses on identifiers carried in URLs

- So why do URLs have such a bad reputation?

CDL

URL fallacies

STOP saying URLs are less stable than scheme X ids

STOP saying a URL is a location

FALSE: URLs are doomed since hostnames go away

UNTRUE: URLs are doomed since HTTP will go away
(what's more stable than a dead namespace?)

MAGICAL THINKING: a scheme confers persistence

CDL

Other identifier schemes

Id schemes claiming suitability for persistence

- ARK, DOI, Handle, INFO, ISSN, MD5, PURL, URI, URL, URN, UUID, and others

- Which id schemes help? Which ones hurt?
  - The answers may surprise you

CDL

What do we want from persistent identifiers?

First, persistent *access* to data objects

- No more “404 Not Found”

Other things we want:

- Persistent *reference*
- Trust in the “authentic object”

CDL

Who is *we* and what is *have*?

How can *we have* identifiers that don't break?

- Ids for Our Stuff vs ids for Their Stuff

  - ids for what we manage and/or control

    ≠what others manage and/or control

- *We* are not most providers

  - not all identifiers should be persistent, and few organizations are suited to it

CDL

Our Stuff vs Their Stuff

Persistence can be split into

- the Our Stuff Problem
- the Their Stuff Problem

It makes no sense for any organization to assign persistent ids to Their Stuff

- Their Stuff can be hugely important to our users, but we don’t control it and cannot vouch for it

Focus on assigning persistent ids to Our Stuff

CDL

What identifiers *cannot* do

Cannot protect against any important threat:

- Political or financial loss (e.g., bankruptcy)
- War, social upheaval, natural disaster
- Power outage, disk failure, human error
- Objects that are *removed*, *replaced*, or *moved* without a redirect being set up

Changing id scheme won't help you, just as changing your wallet won't make you rich.

Don't expect too much. The id was the last person to see the object alive, but not the murderer.

CDL

Ids useless for persistence?

Almost. Moreover, they can get in the way

- May increase your costs, risks, and dependencies

A good id should stay out of the way and be

- Low-cost, low-complexity, low-dependence
- Allow you to focus on identifier *services* (eg, description, access, policy) that instill confidence
- Help eliminate the few failures still traceable to poor identifier string assignment practices

CDL

What identifiers *can* do

As a string, an identifier can provide "words"

- with clues to the *object's* alleged nature
  - short-term feature but long-term risk
- with clues to the *identifier's* alleged nature
- and, if actionable, access to *services* that may provide things we actually want:
  - objects, metadata, policies, versions

CDL

Identifiers: 4 big issues

1. Form: the string's syntax and structure
   http://…, ark:/…, doi:…, hdl:…, urn:…, etc.
2. Assignment: local practice and ways to screw it up
   (stupid semantics vs smart semantics)
3. Resolution computation, mapping id *to* a thing
   id2file (id-to-file), id2metadata, id2otherid, etc.
4. Maintenance: keeping ids actionable/resolvable
   "id2thing" table administration, GUIs, APIs

CDL

Opaque ids vs semantic ids

Tension between access and longevity

- Want opaque ids for names that age and travel well
- But semantically laden ids provide many id services

Hybrid:

- Opaque ids used to name abstract objects
- Semantic and sometimes transient extensions address components inside of objects (the set of components evolves over time anyway)

CDL

Persistence and identifier strings

Desirable characteristics

- Longevity -- make them opaque (free of meaning) to avoid inevitable semantic “rot”
- Transcribability -- make them short (eg, tinyurl)
- Keyboarding, typesetting, error detection
- Pressure to accommodate meaning, at least selectively, even in long-term persistent identifiers
- User-friendliness: non-opaque, transient qualifiers
- Provider-friendliness: branding
- Tool-friendliness: non-opaque filename extensions
- The best place to accommodate meaning is in metadata

CDL

Identifier lexical features

Hyphens ignored?

- Neutralizes harm done by typesetters

Too many search results? Can providers disclose (without requiring tedious database lookup) …

- Sub-object hierarchy, e.g., reserving ‘/’
- Variant objects, e.g., reserving ‘.’

CDL

String features by scheme

Uniqueness: globally unique prefix plus unique name

Registry-based prefixes from

- DNS, Handle, DOI, URN, ARK, even UUID

Plus unique name

- Local practices assisted by NOID, UUID

Opacity for longevity (NOID, UUID)

Actionability (any scheme that is embeddable in URLs)

- Identity-inert hostnames for branding (ARK)

Structure that can disclose object versions & hierarchy (ARK)

Contains a check digit (NOID, ISBN)

String extends to address metadata and policy records (ARK)

CDL

Do your ids get in the way?

Are you distracted by your persistent id system?

Is it even possible that your persistent identifier system increases instead of decreases your risk?

- Do you pay to use your ids or their resolver?
- How large and complex a system is it to maintain and who has to fix it?

Scheme costs and risks

All schemes need indefinite support for at least

- Web server, web browser, and domain names
- Indirection or redirection tables

In addition, URN, Handle, and DOI resolution need a global proxy or a plugin *for every* access

- URL, ARK, and PURL need no plugin

Handle and DOI require

- You to maintain an extra local server
- Your community to maintain global servers
- You to accept these costs and risks

CDL

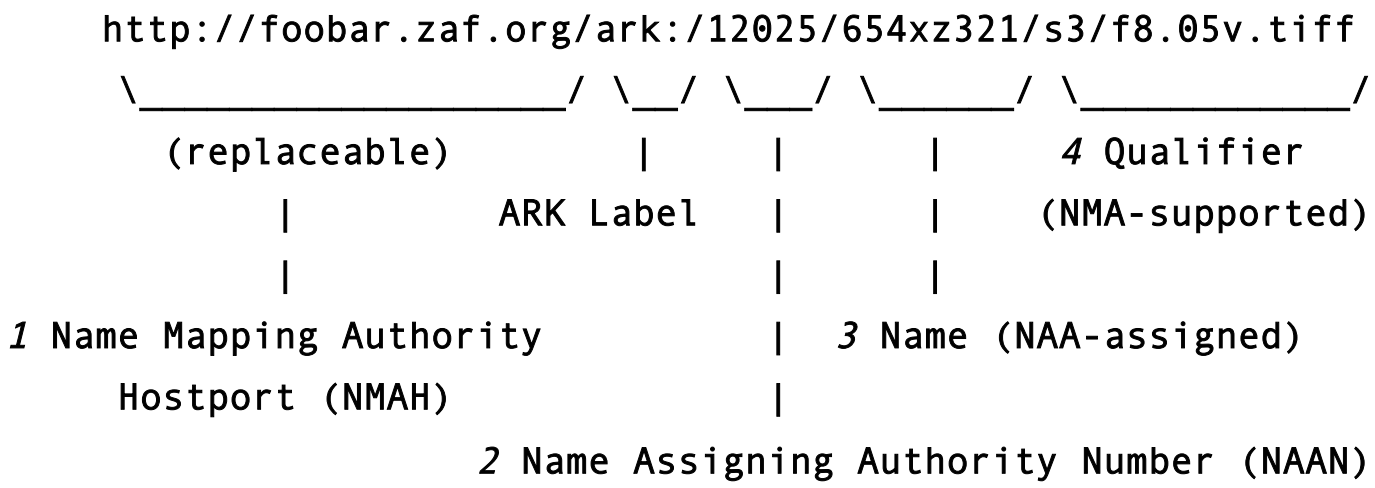
Advertisement: ARK namespaces reserved

12025	National Library of Medicine
12026	Library of Congress
26677	Library and Archives Canada
13030	California Digital Library
13960	Internet Archive
13038	World Intellectual Property Organization
78319	Google
61001	University of Chicago
28722	University of California Berkeley
15230	Rutgers University Libraries
64269	UK Digital Curation Centre
62624	New York University Libraries
52327	National Library and Archives of Quebec
27927	Portico/Ithaka Electronic-Archiving Initiative
12148	National Library of France

Reserve a namespace by email to ark@cdlib.org

ARK Summary

Instead of one Name Authority: Assigning Authority + Mapping Authorities



1 = current service provider; identity inert; replaceable

2 = organization that originally assigned the id

3 = name originally assigned to the abstract object, often opaque

4 = extension disclosing object hierarchy & variants, often non-opaque

CDL

ARK usage

Two ARKs accessing the same thing

`http://loc.gov/ark:/12025/654xz321`

`http://rutgers.edu/ark:/12025/654xz321`

Access to metadata -- add a ‘?’

`http://loc.gov/ark:/12025/654xz321?`

Access to support statement -- add ‘??’

`http://loc.gov/ark:/12025/654xz321??`

3 minimal requirements to be an ARK

- An archive that can’t do all 3 -- trustworthy?
- Is an ARK persistent? Maybe. Have to *ask*.

CDL

ARKs return DCMI kernel metadata

An ARK identifies itself when you append a '?'

```
http://ark.cdlib.org/ark:/13030/tf0v19n804?
```

When given to a browser, this returns

```
erc:
who:   (:unav) unavailable
what:  "Pack Shinto Temple Property for Moving -- Fumiko Miyoshi,
    18-year-old daughter of the priest of a Japanese Shinto temple
    in a Southern California defense area from which all Japanese
    are being evicted, was helped February 19 by Jimmy Okumura as
    she started packing some of the temple property in preparation
    for moving.  She is wrapping a koto, Japanese harp."--caption
    on photograph
when:  (:unav) unavailable
where: http://ark.cdlib.org/ark:/13030/tf0v19n804
```

Opaque identifier tools

Non-opaque identifier strings are chosen deliberately to assert facts

- But only true at the time of assignment

Opaque identifier strings are best chosen by automated means, such as

- NOID (nice opaque identifier) or
- UUID/GUID (universally unique identifier)

CDL

Nice opaque identifiers (NOID)

A noid *minter* is a lightweight database for generating, tracking, and binding unique ids

The `noid` tool creates minters and accepts commands that operate them

- Mints random or sequential, opt. check digits
- Includes a web-based name resolver
- Open source, available at www.cpan.org

CDL

Using NOID

Identifiers minted according to a template:

```
noid dbcreate f5.reedeedk long 13030
```

which produces as first minted id

```
13030/f54x54g11
```

Noid is scheme-independent

- Used for ARKs, Handles, URLs, session ids, etc.

CDL

Some documentation

ARK specification

http://www.cdlib.org/inside/diglib/ark/arkspec.html

NOID

http://www.cdlib.org/inside/diglib/ark/noid.pdf

CDL

Identifier resolvers: how do identifiers break?

Whatever the string, what matters is the *thing*

Realistically, only URLs matter, where broken means either

1. The hostname is broken

   Server down, gone, or renamed *

   Domain name lost, provider out of business or

2. The pathname to the thing is broken

   Thing down, gone, or renamed *

- Global name resolvers to the rescue? NO!
  - No global fix for these*, only individual providers can fix
  - And a local resolver only helps for renamed objects (eg, HTTP redirects, or simple 2-column database mapping)

CDL

Name-to-Thing (N2T) overview

Each member publishes URLs under n2t.info:

`http://n2t.info/12345/foo/bar.zaf`

…which redirects to member 12345's server

The URL is *protected* from local server name instability (but not from local server stupidity)

CDL

Resolution via N2T (Name-to-Thing)

N2T is two things at once

A consortium of cultural memory organizations

… and …

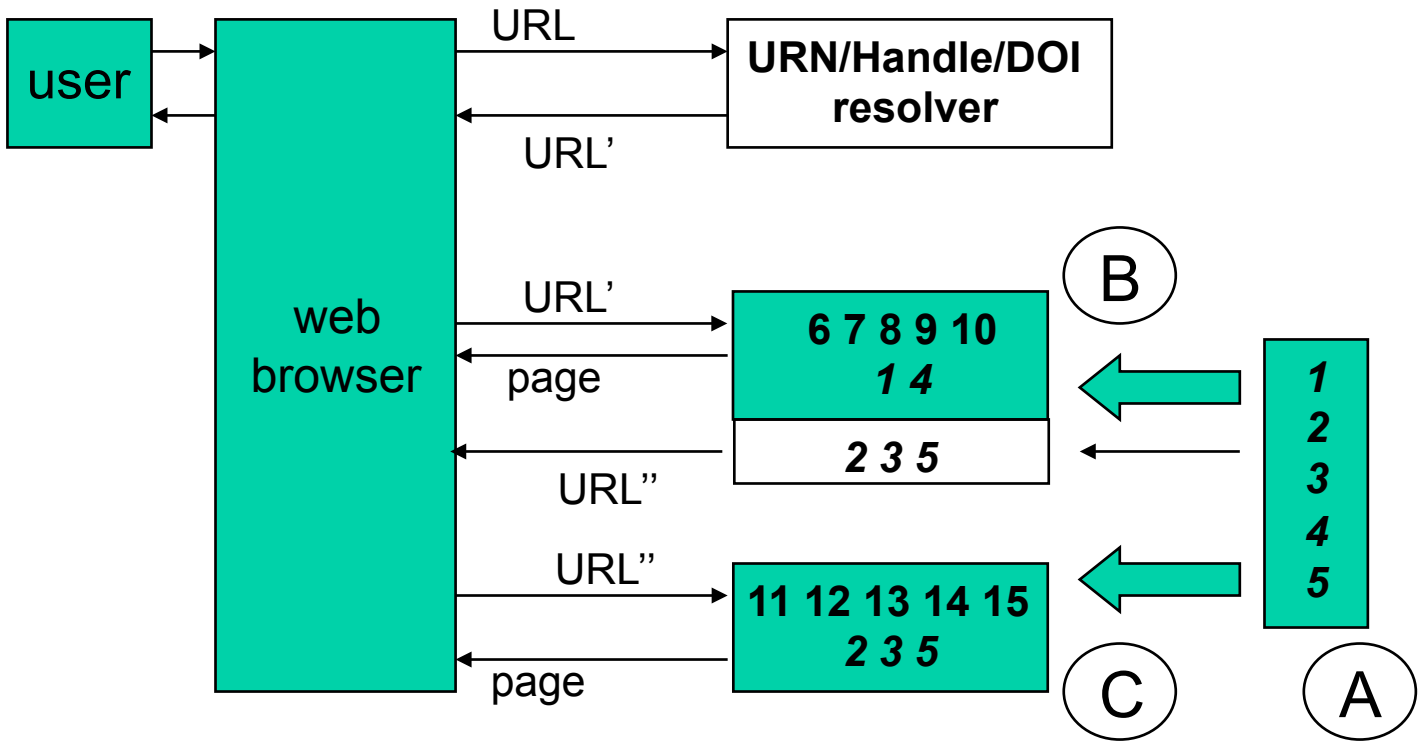
A small, ordinary web server, mirrored in several instances globally for reliability

Basic vision: protect 200 organizations' URLs from *hostname instability* with 200 rewrite rules

How: simple HTTP redirects, one per organization

CDL

Namespace Splitting Problem



Org'n A's namespace splits when B and C inherit its objects.
Under the URN/Handle/DOI model, B must still forward to C.
A table is needed where it can be supported, e.g., N2T.

N2T and multiple id schemes

All depend on Name Assigning Authority

- http://n2t.info/NAA/...
- http://n2t.info/ark:/NAA/...
- http://n2t.info/urn:NAA:...
- http://n2t.info/hdl:NAA/...
- http://n2t.info/doi:NAA/...
- http://n2t.info/purl:/NAA/...

Where to register NAA number (eg, 12345)?

CDL

n2t.info summary

Persistent identifier resolution using a very simple architecture

No proprietary, special-purpose infrastructure to carry forward as a liability to persistence

No browser modification required

Identifier scheme-agnostic

CDL

Conclusion: identifier big issue breakdown

Form: the string's syntax and structure

http://…, ark:/…, doi:…, hdl:…, urn:…, etc.

Assignment: local practice and ways to screw it up

stupid semantics vs smart semantics

Resolution: a computation with id in & something out

id2file (id-to-file), id2metadata, id2otherid, etc.

Maintenance: keeping ids actionable/resolvable

"id2thing" table administration, GUIs, APIs

CDL

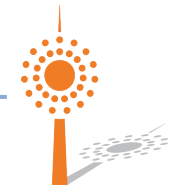
International Conference on Dublin Core and Metadata Applications

Dublin Core and other metadata schemas

Mikael Nilsson
<mikael@nilsson.name>

Four rules for exposing information on the web

1) Use URIs as names for things
2) Use HTTP URIs so that people can look up those names.
3) When someone looks up a URI, provide useful information.
4) Include links to other URIs. so that they can discover more things.



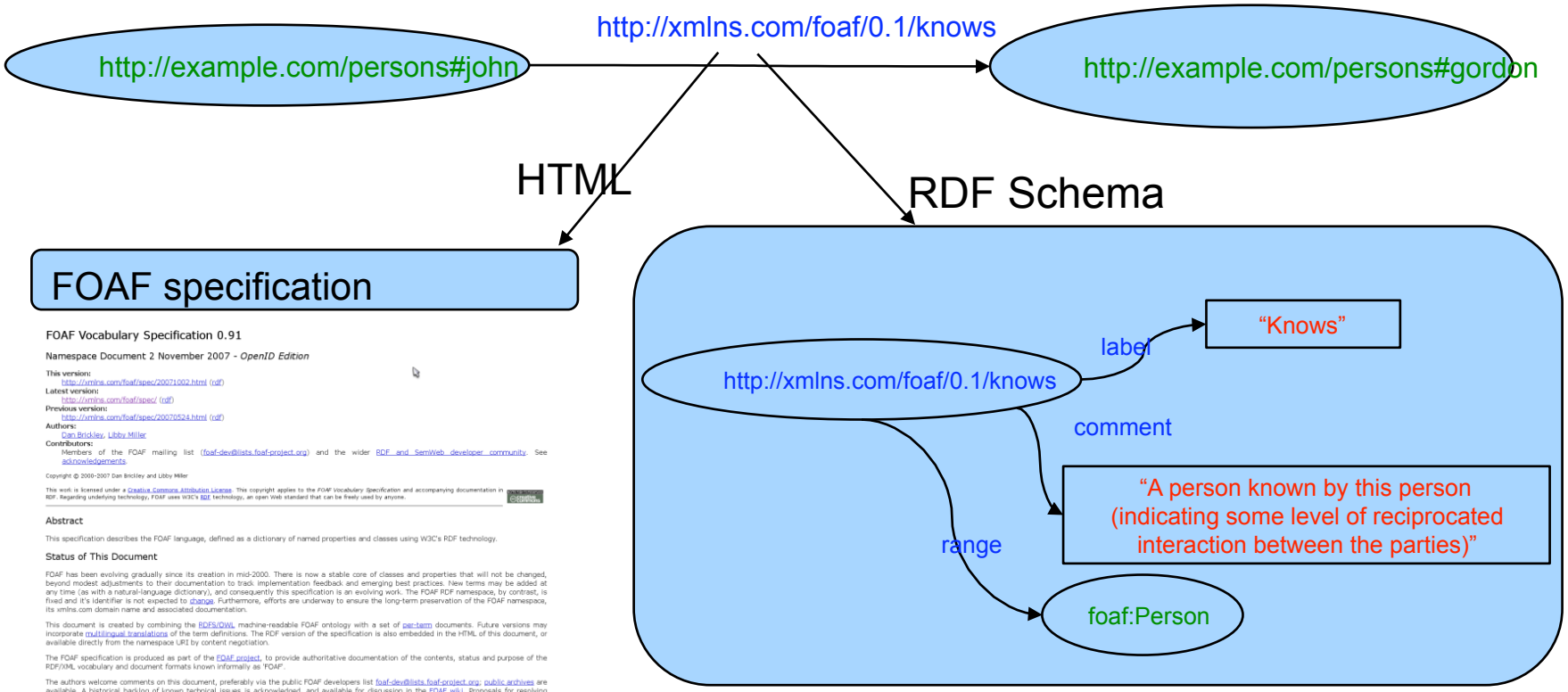
Uses of machine semantics

- The use of a common framework means metadata...
  - from different domains
  - using different vocabularies
  - used in different technical environments...

... can be combined without effort.

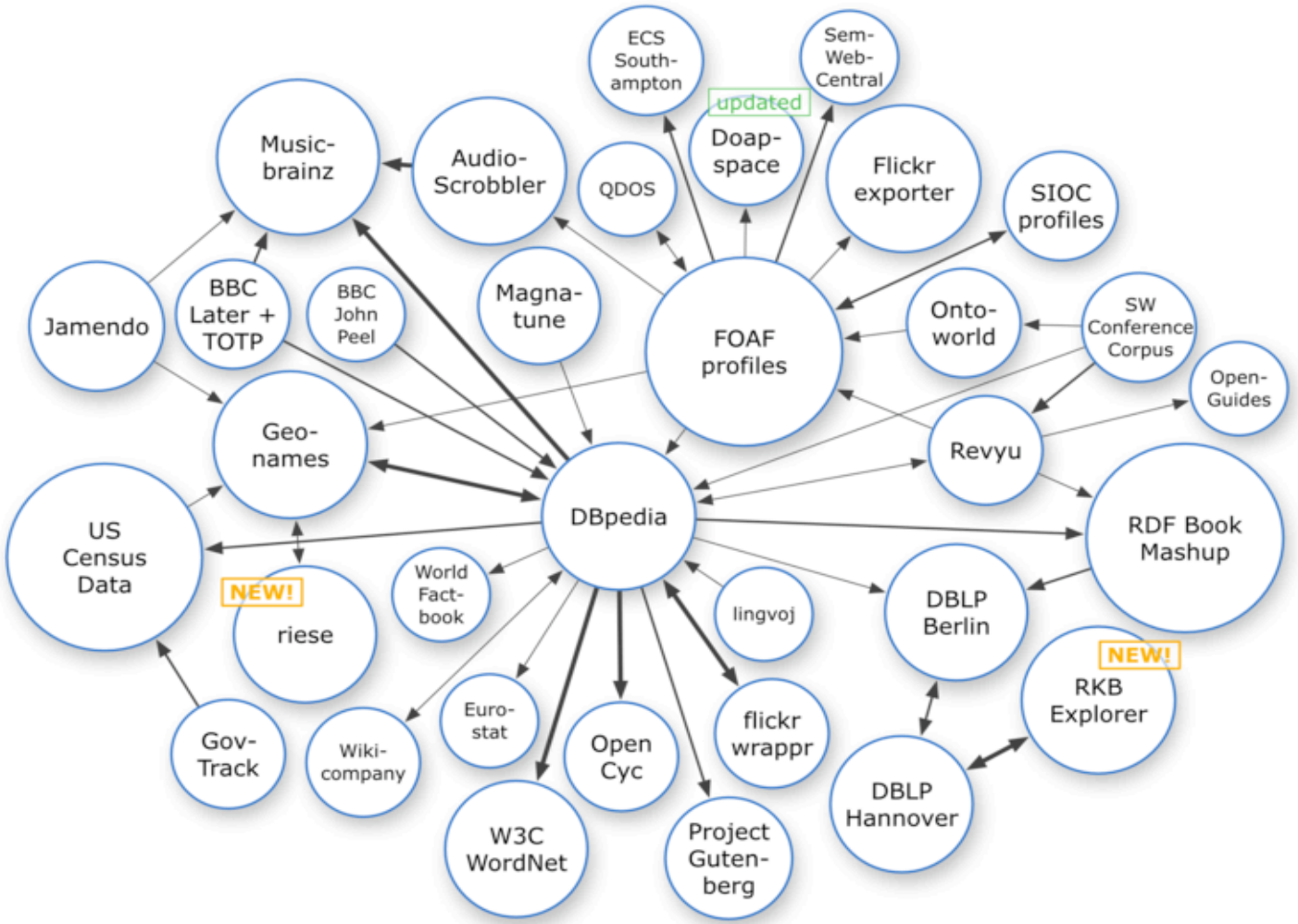
- Ontologies
  - Enable advanced processing of metadata
- Automatic discovery of term definitions
  - “Follow your nose”
- Linked open data
  - Giant global graph of metadata

Follow your nose




```
<rdf:Property rdf:about="http://xmlns.com/foaf/0.1/knows"
vs:term_status="testing" rdfs:label="knows" rdfs:comment="A person known by this
person (indicating some level of reciprocated interaction between the parties).">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
    <rdfs:domain rdf:resource="http://xmlns.com/foaf/0.1/Person"/>
    <rdfs:range rdf:resource="http://xmlns.com/foaf/0.1/Person"/>
    <rdfs:isDefinedBy rdf:resource="http://xmlns.com/foaf/0.1/"/>
</rdf:Property>
```

Linked Open Data

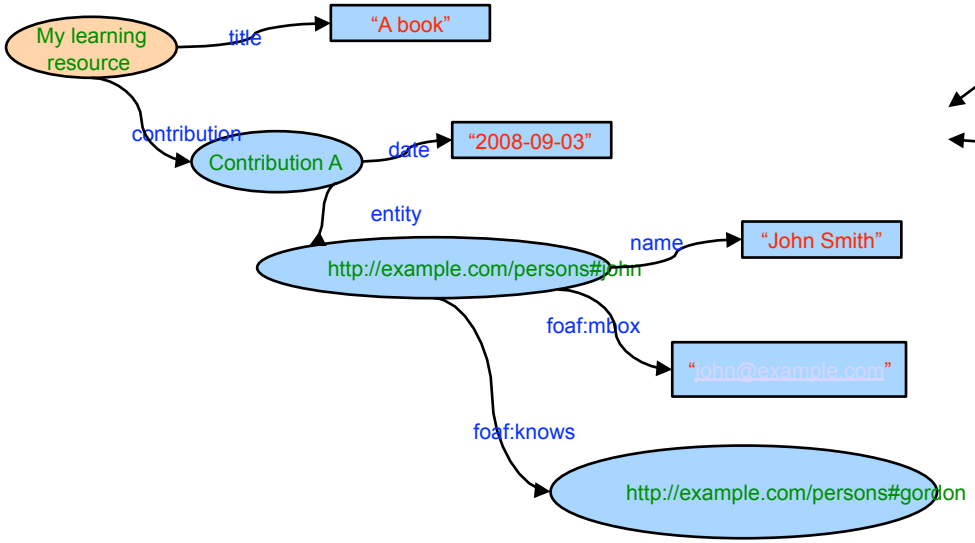


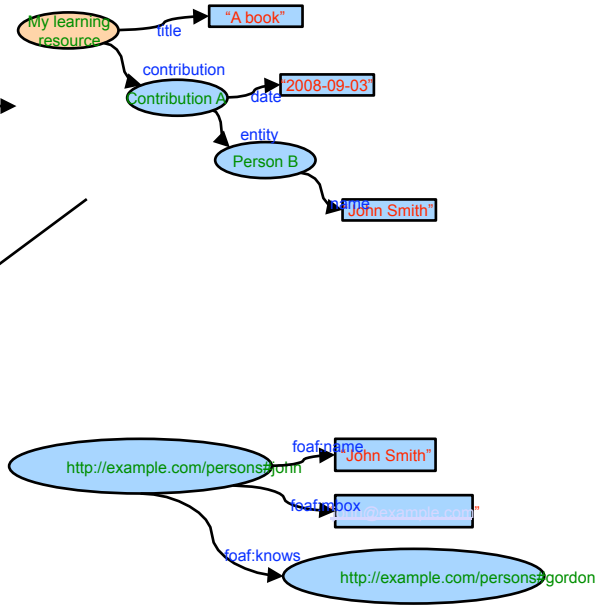
- More than 2 billion RDF triples

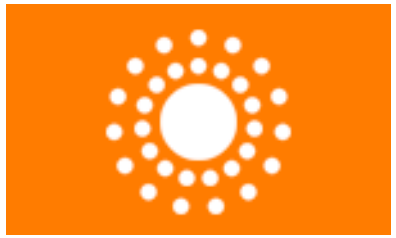
Example: From XML to graphs

```
<LearningResource grddl:transform="http://yyy/mlr.xsl">
      <Title>A book</title>
      <Contribution>
          <Date>2008-09-03</Date>
            <Entity>
                <Name>John Smith</Name>
            </Entity>
      </Contribution>
</LearningResource>
```

http://yyy/mlr.xsl







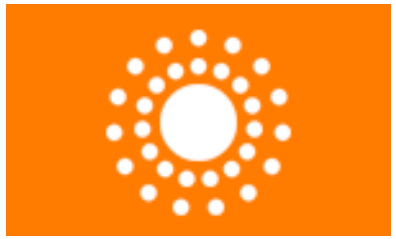
DCMI Identifiers Community Session

~~Douglas Campbell~~ ~~Muriel Foulonneau~~

John Kunze

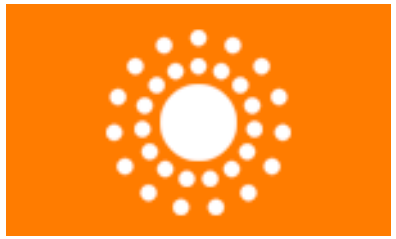
24 September 2008 – DC2008 Berlin

Identifiers Agenda



- Introduction to the Identifiers Community
- Mikael Nilsson: W3C's linked data approach
- John Kunze: Persistent identifier principles and practice
- Brief updates from the room on identifier-related work or news (all invited to talk)
- Open discussion

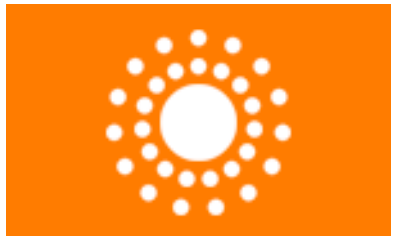
Identifiers Community



- Started Oct 1, 2007
- Charter
  - A forum for individuals and organisations with an interest in the design and use of identifiers in metadata
  - A liaison channel for those involved in identifier efforts in other domains
- Home page

  http://dublincore.org/groups/identifiers/

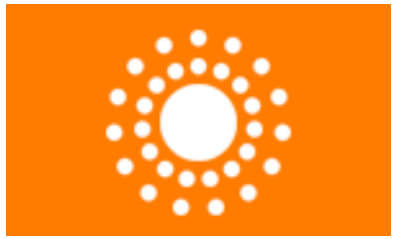
Talk: Mikael Nilsson



“W3C’s linked data approach”

Rules for using URIs to assign names to everything so they can be linked together

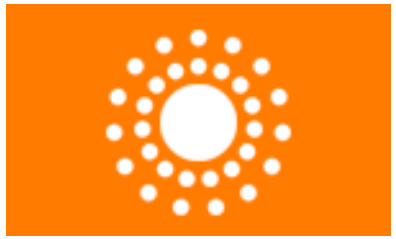
Talk: John Kunze



“Persistent identifier principles and practice”

The goals of persistent identifiers and designing services around identifiers

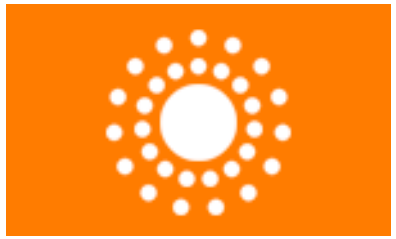
Updates from the room



Brief updates on identifier-related work or news

- All present are invited to talk

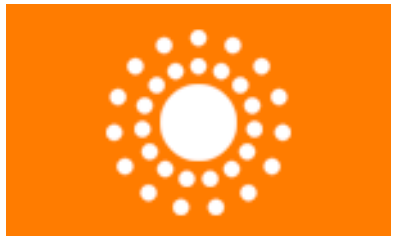
Open Discussion



Potential topics include:

- Opinions on W3C's linked data approach
- The use of URI-based identifiers in library-based metadata
- Possible involvement in other DCMI communities and task forces
- Identifying topics for ongoing discussion on the listserv
- What else?

Identifiers Wrapup



- Wrap up and next steps