From trees to graphs: Creating Linked Data from XML

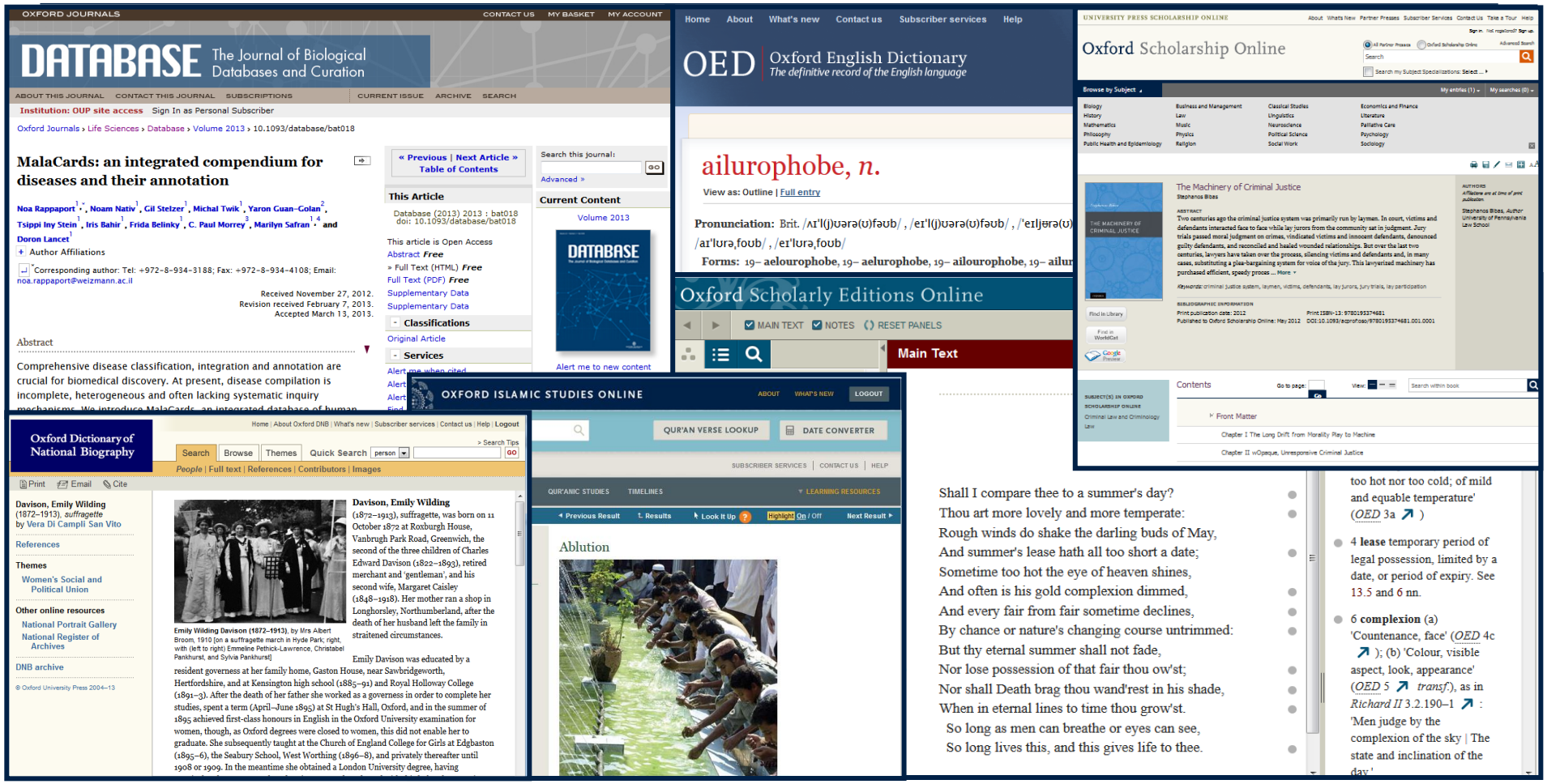
Catherine Dolbear & Shaun McDonald
Content Architecture, Global Academic Business
Oxford University Press

Overview

- OUP and our business drivers
- Approaches in the literature
- Our publishing workflow and XML metadata
- Modelling RDF graphs from XML trees
- Semantic markup: RDFa and schema.org
- Summary

Introduction to OUP

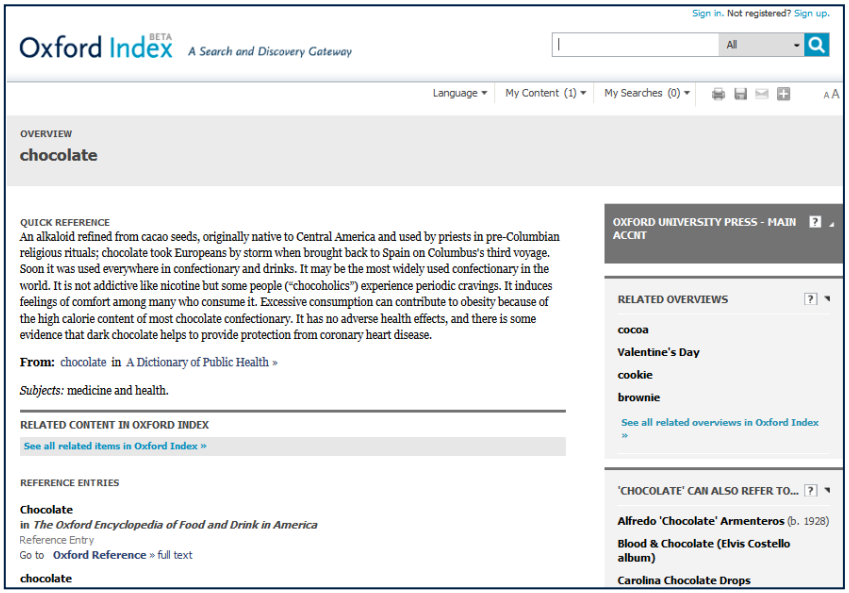
Meet the Press…



Motivation and business drivers

OXFORD
UNIVERSITY PRESS

- Search Engine Optimisation
  - Discoverability of our subscription content
  - “Index card” of XML metadata published open access

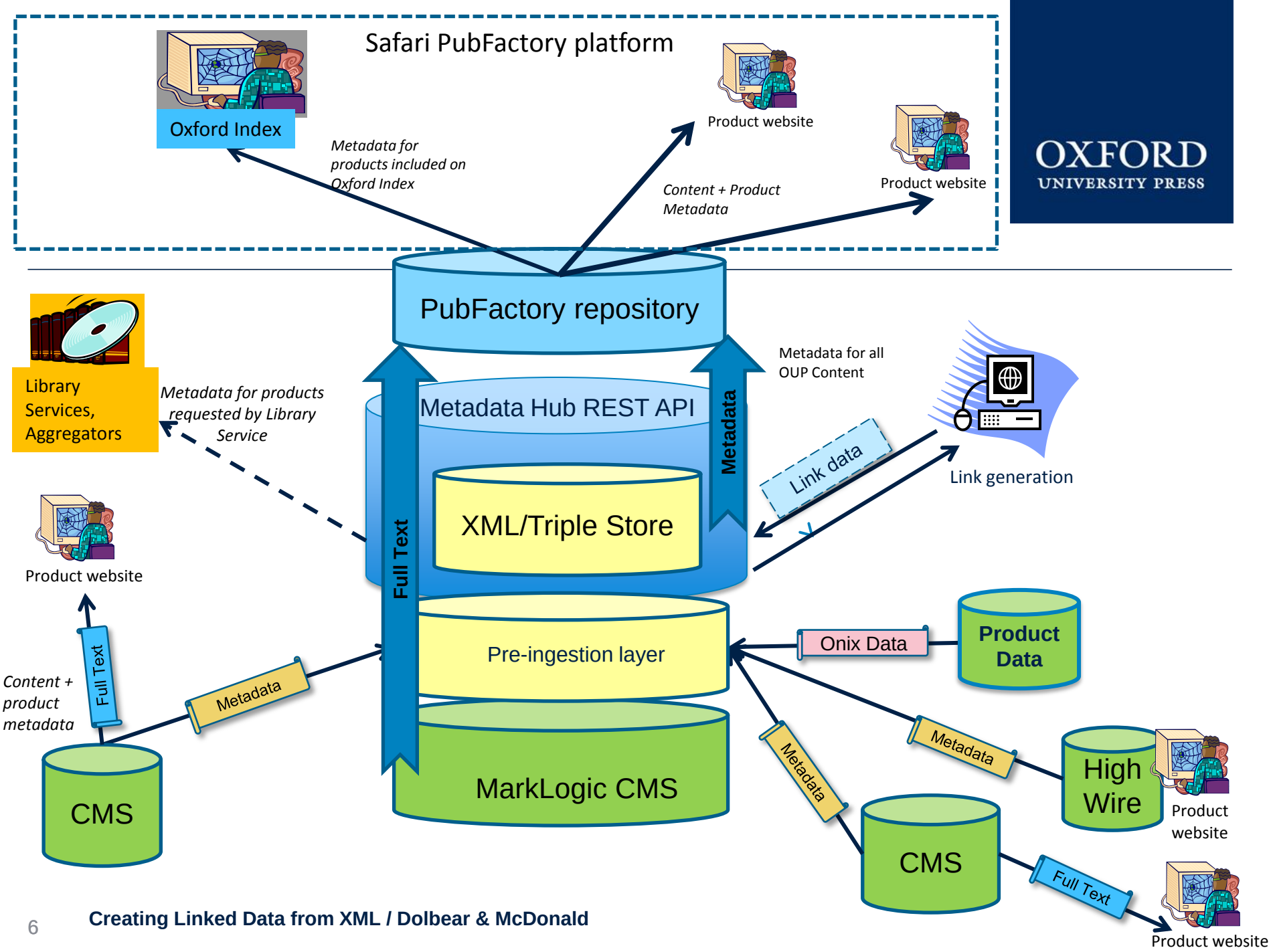


- Improvement of user journeys across multiple products
  - Dynamic links generated as search results
  - Static links e.g. *is Author Of, has Primary Topic* currently stored as XML documents

Approaches in the literature

What's been tried before

- MarkLogic
  - XQuery to construct triples from XML, linked using URIs
  - We follow this pattern using Digital Object Identifiers expressed as URIs
- BBC
  - Statistics and content in MarkLogic XML database
  - Journalists annotate assets according to an ontology, results stored in OWLIM triple store.
  - Content aggregated by combining SPARQL and XQuery e.g. "The league table for the English Premiership"
- Nature Publishing Group
  - Adobe XMP, a subset of RDF embedded in XML documents
  - Triple store enables integrated queries of all XML content distributed across the organisation


Safari PubFactory platform
Oxford Index
Metadata for products included on Oxford Index
Product website
Content + Product Metadata
Product website
OXFORD UNIVERSITY PRESS
PubFactory repository
Library Services, Aggregators
Metadata for products requested by Library Service
Metadata Hub REST API
Metadata
Full Text
XML/Triple Store
Metadata for all OUP Content
Link data
Link generation
Product website
Pre-ingestion layer
Onix Data
Product Data
Content + product metadata
Full Text
Metadata
CMS
MarkLogic CMS
Metadata
Metadata
High Wire
Product website
CMS
Full Text
Product website

OxMetaML

OUP's XML schema for metadata

- Single vocabulary for metadata for all products
  - Originates from multiple sources with varying DTDs or none
  - MarkLogic, FileMaker, SQL server, even Excel spreadsheets
- Reuses some Dublin Core vocabulary, plus terms based on our own needs
- Links embedded in XML document or "stand-alone" OxMetaLinkML documents
  - Named predicates like "*is author of*", "*is related to*", "*is primary topic of*"
- Published as XML for externally-developed product website platform
  - Document-centric

Modelling RDF graphs

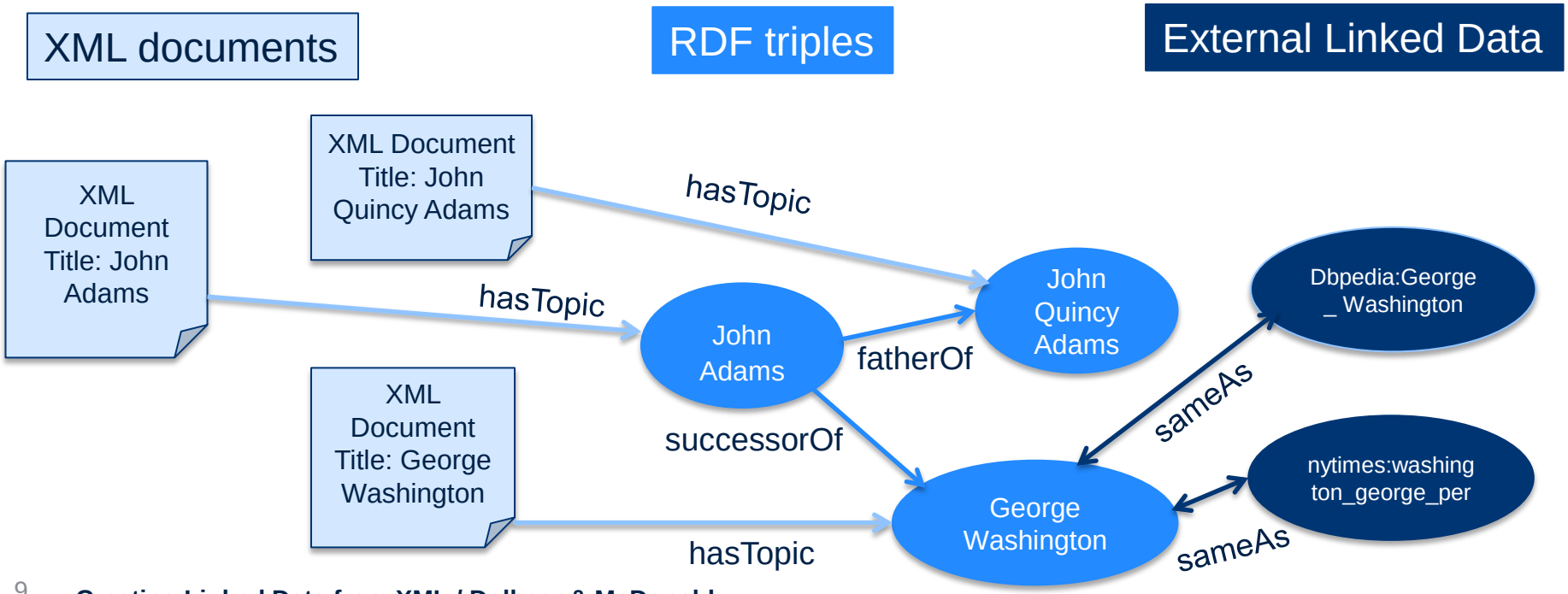
There is no order…

- XML: documents, elements, sequential order – *trees*
- RDF: relationships between concepts - *vertices and arcs*
  - Difficult to manipulate relationships in XML
- XML for content, RDF for metadata
- Our metadata includes abstracts and must be output to XML
- But as more concepts in the XML become linked in their own right and given identifiers, more can migrate to a graph model.

Bibliographic versus semantic metadata

Information versus meaning

- Bibliographic information (author, title, ISBN etc)
- Semantic or contextual information - what the document is about (academic subject, person, organisation etc)



RDF Data Model

OXFORD
UNIVERSITY PRESS

- RDF is a data model (graph) not a syntax
- Use Turtle, not RDF/XML
  - Less verbose, less syntactic variation
  - Can concentrate on knowledge modelling
  - Element order and syntactic use of rdf:Description or rdf:about is irrelevant
- Better performance to generate inverse triples from SPARQL query rather than store explicitly or use inference

Examples

Turtle and SPARQL

OXFORD UNIVERSITY PRESS

```
DOI123 a oup:Document.
DOI123 foaf:hasTopic URI456.
URI456 oup:hasName “George Washington”.
URI456 oup:hasSuccessor URI789.
URI789 oup:hasName “John Adams”.
```

Examples

Turtle and SPARQL

DOI123 a oup:Document.

DOI123 foaf:hasTopic URI456.

URI456 oup:hasName “George Washington”.

URI456 oup:hasSuccessor URI789.

URI789 oup:hasName “John Adams”.

URI789 oup:isSuccessorOf URI456.

Encode inverse triple explicitly

Examples

Turtle and SPARQL

OXFORD UNIVERSITY PRESS

DOI123 a oup:Document.

DOI123 foaf:hasTopic URI456.

URI456 oup:hasName “George Washington”.

URI456 oup:hasSuccessor URI789.

URI789 oup:hasName “John Adams”.

Infer inverse triple using inference engine

oup:hasSuccesor a rdf:Property.

oup:hasSuccessor owl:inverseOf oup:isSuccessorOf.

=> URI789 oup:isSuccessorOf URI456.

Examples

Turtle and SPARQL

```
DOI123 a oup:Document.
DOI123 foaf:hasTopic URI456.
URI456 oup:hasName “George Washington”.
URI456 oup:hasSuccessor URI789.
URI789 oup:hasName “John Adams”.
```

```
CONSTRUCT {?subject oup:isSuccessorOf URI456}
WHERE {
    URI456 oup:hasSuccessor ?subject.
}
```

Generate inverse triple as query result

Result:
URI789 oup:isSuccessorOf URI456.

Reification

Information about the triples

- Accuracy of the link, date of creation, approval status etc.
- Can store a fourth piece of information in RDF by:
  - Named graphs aka “quads”. More suited to groups of triples
  - Assign a URI to each triple and treat as a resource using RDF *reification* vocabulary

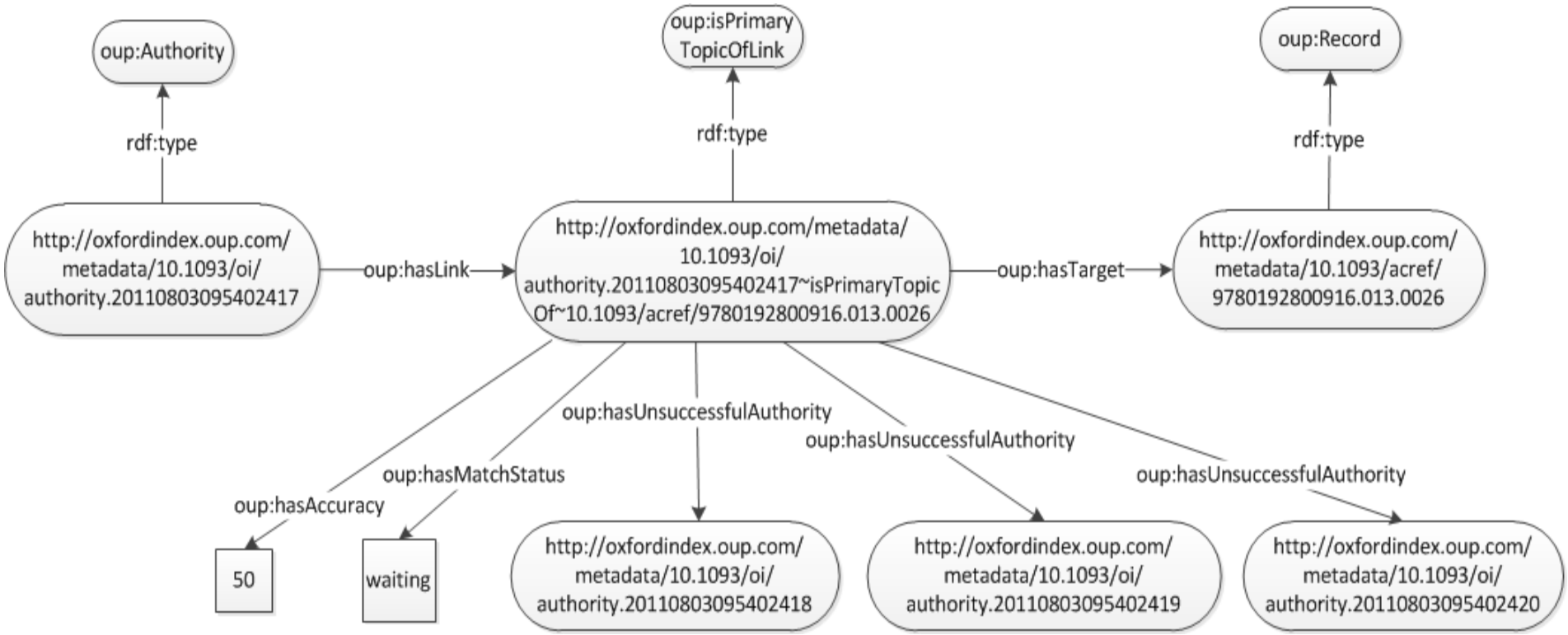
```
<URI20110803100243337> oup:hasOccupation “President of the United States”.

<Statement12345> a rdf:Statement;
                 rdf:subject <URI20110803100243337>;
                 rdf:predicate oup:hasOccupation;
                 rdf:object “President of the United States”.

<Statement12345> oup:isValidFrom “20 January 2009”.
```

Reification using RDFS Classes

Simpler queries; better performance



Linked Data

principles for connecting 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 RDF information
4. Include RDF statements that link to other URIs so that they can discover related things

- Connections across content, not just documents
- Distinguishes between a *document about* Barack Obama, and the man himself
- At the moment, our DOIs provide documents, not data

Business cases for Linked Data

Where's the money?

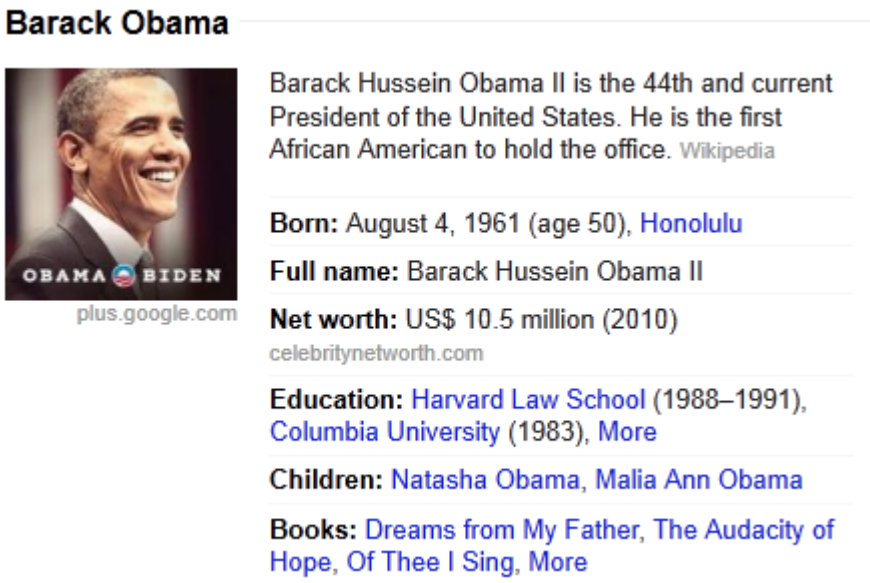
- Internal benefits for using RDF:
  - Storing links between XML documents
  - Using external RDF data to augment our metadata (e.g. OBO ontology to identify gene names in abstracts)
- ROI from publishing OUP metadata as Linked Data less clear
- Could be used to supply metadata to library services and aggregators (e.g. EBSCO, Summon)
- Business models: branding, freemium, **traffic model**
  - First step to publish RDF as embedded markup

RDFa and schema.org markup

Embedding RDF in HTML

- Improves click-through rate (30% reported by BestBuy) as search results more eye-catching

```
<div vocab="http://schema.org/"
 typeof="Person"
 about="http://oxfordindex.oup.com/view/10.1093/oi/authority.20110803100243337">
 <span property="name">Barack Obama</span> <p/>
 <span property="jobTitle">American Democratic statesman</span> <p/>
 born <span property="birthDate">4 August 1961</span> <p/>
</div>
```



RDFa versus schema.org

OXFORD
UNIVERSITY PRESS

- RDFa allows for richer descriptions
  - Can provide our full metadata “under the hood”
- But schema.org fully supported by major search engines
  - We could use CreativeWork schema (Book, Article concepts) as well as Person
- Drawback is that only simple markup can be used
  - Can introduce semantic mismatch – is “American democratic statesman” really a job title?
  - Not a full alternative to an API or Linked Data publication

Summary

Our journey from XML to Linked Data

- We're still in the early days…
- Internal business case for semantic technologies and link generation (SEO, user journeys) is much stronger than for Linked Data publication itself
- XML for documents, RDF for relationships
  - How much of our metadata should we store as RDF?
- Is our experimental architecture of an XML store for documents and a triple store for links the most performant?