

# RDF, Dublin Core, etc.

## Embedding Metadata in CellML

Warren Hedley<sup>1</sup>, Melanie Nelson<sup>2</sup>

<sup>1</sup>The Bioengineering Research Group, The University of Auckland

<sup>2</sup>Physiome Sciences Inc, New Jersey

# Metadata – what and why?

- The short answer: “Data about data”
- The classic example: a library catalogue
- In a CellML document, the data is the model’s structure and mathematics
- Metadata adds context to the data
- In CellML, metadata allows searches by model type, function and author

# CellML – What to store?

- Document Information – names, dates, publisher, copyright, contact details
- Human-readable names for objects
- Annotations – inline documentation, and associated authoring information
- Biological Info – species, cell type, etc.
- Mathematics Info – problem types
- Citation / Reference Info

# Approach 1

- It's XML – we can make up own tags, right?

```
<model name="hodgkin_huxley_squid_1952" id="hh_1952">
  <metadata>
    <name>Hodgkin Huxley Squid Axon Model</name>
    <document_creator>Warren Hedley</document_creator>
    <doc_last_modified>2001/05/10</doc_last_modified>
    <from_paper
      bibtex_file="/usr/share/references.bib"
      identifier="hh_squid_1952" />
    <species>bloody big squid</species>
  </metadata>
  <component> ... </component>
</model>
```

# Approach 1 – Why it's bad

- Not portable:
  - Only software that knows about "CellML Metadata" can understand CellML metadata
  - Data doesn't adhere to standard schemes
  - Information in local resources is not available to others, and access protocol not specified

# RDF – What?

- Resource Description Framework (RDF) Model and Syntax Specification
- W3C – 22 February 1999
- RDF provides a model for associating metadata with objects and classifying that metadata
- The RDF model includes grouping types
- The spec provides an XML serialization of this data model

# RDF – Why?

- The semantic web – TBL's vision
- Software can discover metadata without needing to know about the data language or the metadata language
- Software knows the type of relationship between metadata and data
- Allows powerful searches:
  - Find all documents where "Warren Hedley" is the "document\_creator" of something
  - Find all "document\_creator"s of "[http://www.cellml.org/models/hh.xml#hh\\_1952](http://www.cellml.org/models/hh.xml#hh_1952)"

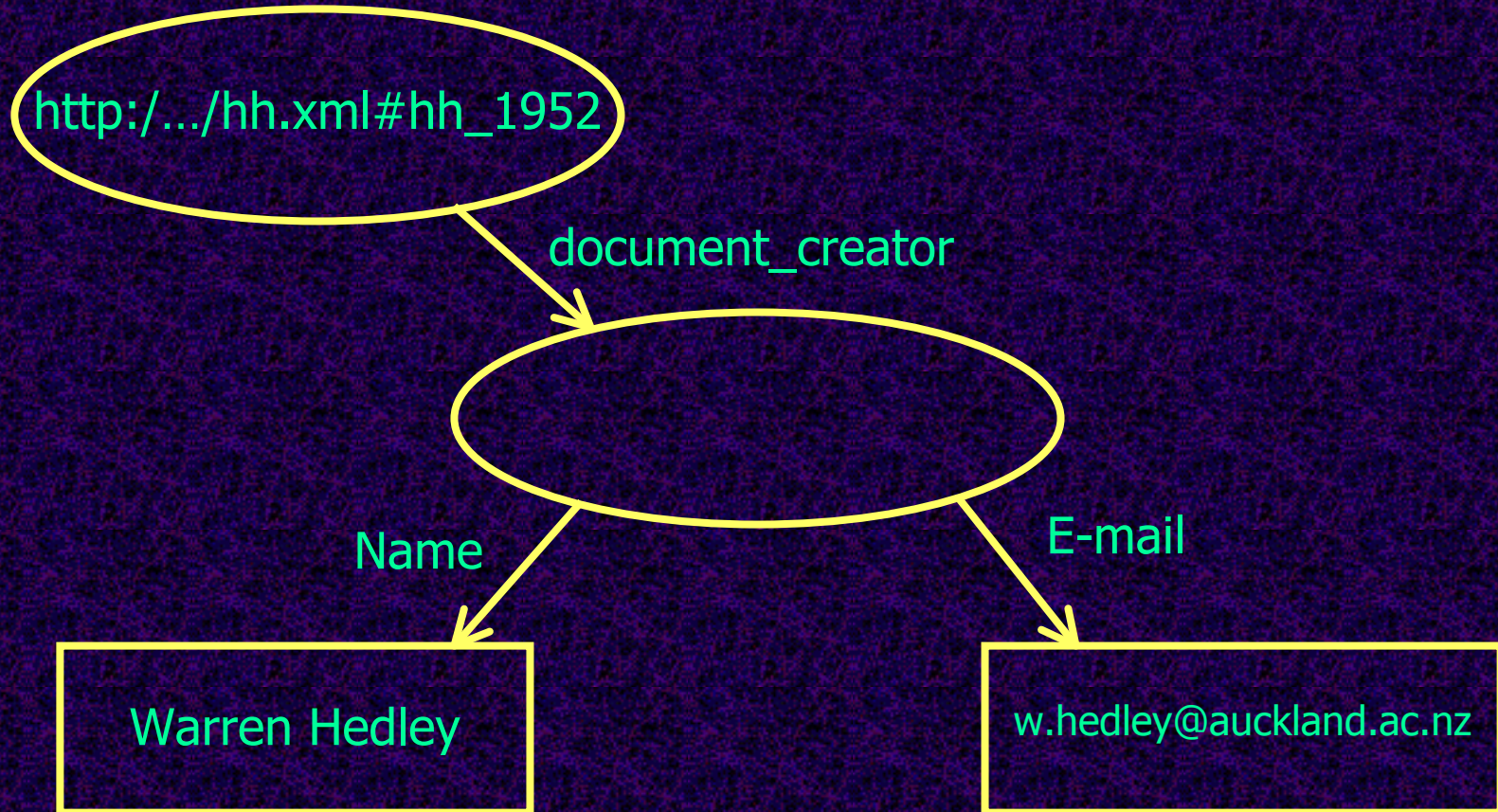
# RDF – The Data Model

- An RDF statement has three parts:
  - Subject: the resource being described
  - Object: metadata about the resource
  - Predicate: the relationship between subject and object
- A statement goes:  
    <subject> HAS <predicate> <object>
- RDF is best represented with diagrams



# RDF – Bigger Data Models

- Objects can be resources too!



# RDF – The XML Serialization

- Framework elements are in the RDF namespace (namespaces allow elements and attributes from a language to be grouped using a prefix)
- A `<rdf:Description>` element declares a subject
- The embedded elements define predicate/object pairs

# RDF – The XML Serialization

```
<model name=" hodgkin_huxley_squid_1952" id="hh_1952">
  <rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
    <rdf:Description about="#hh_1952">
      <document_creator>
        <rdf:Description>
          <Name>Warren Hedley</Name>
          <E-mail>w.hedley@auckland.ac.nz</E-mail>
        </rdf:Description>
      </document_creator>
    </rdf:Description>
  </rdf:RDF>
</model>
```

- RDF has more than one way to do it!
- This is the long way

# Approach 2 – Use RDF

```
<model name=" hodgkin_huxley_squid_1952" id="hh_1952">
  <rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    <rdf:Description about="#hh_1952">
      <document_creator>Warren Hedley</document_creator>
      <doc_last_modified>2001/05/10</doc_last_modified>
      <from_paper
        bibtex_file="/usr/share/references.bib"
        identifier="hh_squid_1952" />
      <species>bloody big squid</species>
    </rdf:Description>
  </rdf:RDF>
</model>
```

- The attributes on <from\_paper /> work as predicate/object pairs

# Approach 2 – Why it's bad

- Software does not recognise the types of relationship between metadata and data
  - What is a "document\_creator"?
- Some aspects still not portable
  - Data doesn't adhere to standard schemes
  - Information in local resources is not available to others, and access protocol not specified

# Dublin Core – What?

- A bunch of librarians developed a standard system of classifications for metadata
- The "Dublin Core Element Set (1.1)" spec defines 15 elements for metadata classification
- The "Dublin Core Qualifiers" spec refines the semantics of the Element Set and defines encoding identifiers
- The Dublin Core is a data model, and independent of serialization

# Dublin Core – Why?

- Allows software to recognise types of metadata
- Recommendations exist for using Dublin Core in HTML and XML (within RDF)
- Allows even more powerful searches
  - 'Find all documents created by "Warren Hedley" ' should work if everyone uses the Dublin Core
  - Could be used to find documents in multiple formats

# The Dublin Core

- Elements by identifier:
  - Title, Creator, Subject, Description, Publisher, Contributor, Date, Type, Format, Identifier, Source, Language, Relation, Coverage, Rights
- Some element refinements:
  - Title/Alternative, Description/Abstract, Date/Created, Date/Valid, Date/Modified
- Some element encodings:
  - Date/W3C-DTF, Identifier/URI, Format/IMT

# Dublin Core in HTML

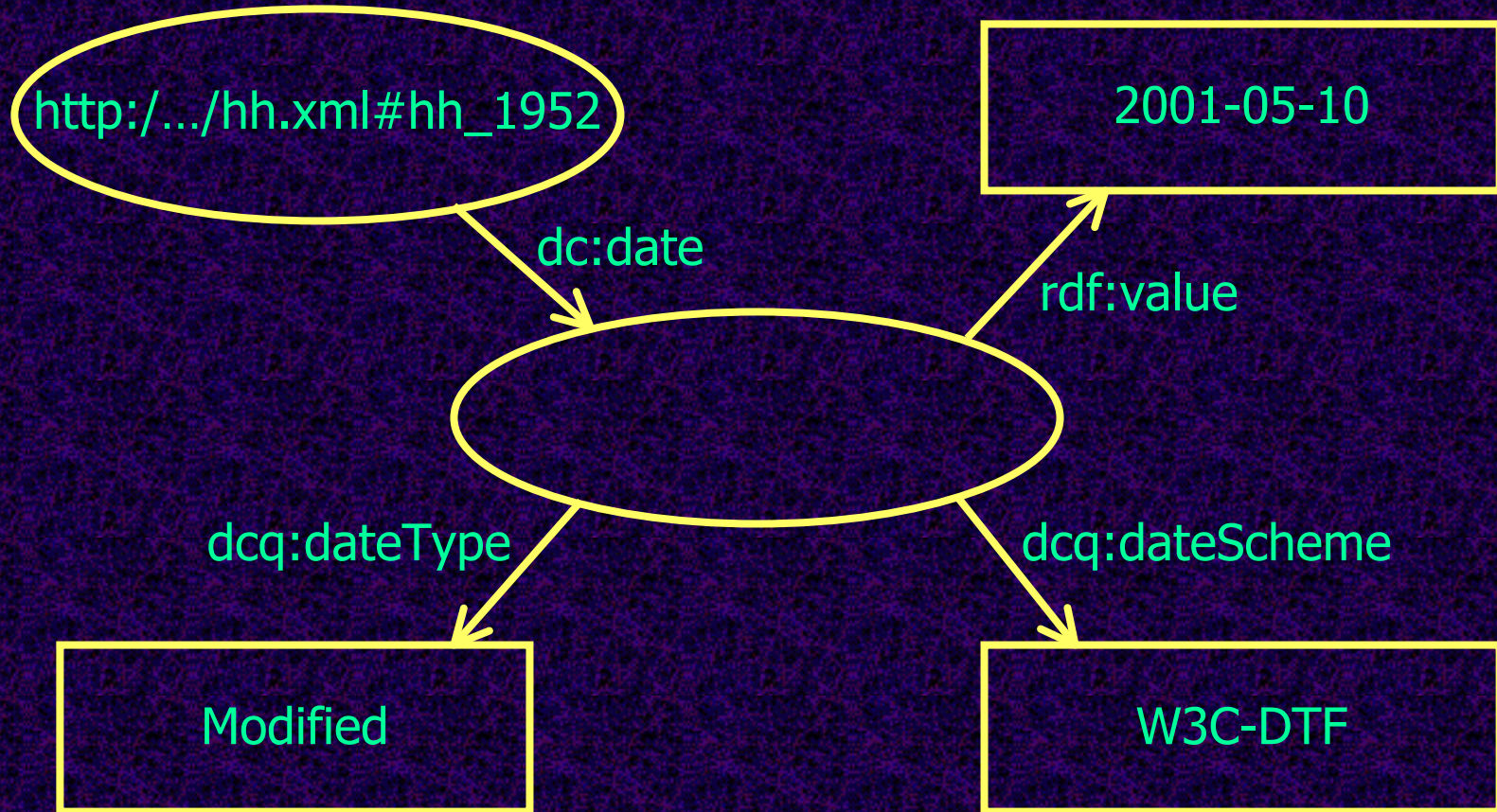
- The following is already understood by some search engines

```
<!DOCTYPE html PUBLIC
  "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <title>Hodgkin Huxley Model 1952</title>
    <link rel="schema.DC"
      href="http://purl.org/DC/elements/1.1/">
    <meta name="dc.Creator" content="Warren Hedley">
    <meta name="dc.Date.Modified" content="2001-05-10">
  </head>
  <body> ... </body>
</html>
```

# Dublin Core in XML (with RDF)

- All Dublin Core classifications must be in the Dublin Core namespace
- All qualifiers must be in the Dublin Core Qualifier namespace
- Element refinements are "Type"s
- Element encodings are "Scheme"s
- Qualified objects are usually anonymous subjects with multiple predicate/object pairs

# Dublin Core in XML (with RDF)



# Dublin Core in XML (with RDF)

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:dcq="http://purl.org/dc/qualifiers/1.0/">
  <rdf:Description about="#hh_1952">
    <dc:Date>
      <rdf:Description>
        <rdf:value>2001-05-10</rdf:value>
        <dcq:dateType>Modified</dcq:dateType>
        <dcq:dateScheme>W3C-DTF</dcq:dateScheme>
      </rdf:Description>
    </dc:Date>
  </rdf:Description>
</rdf:RDF>
```

- Cf. <doc\_last\_modified>2001/05/10</doc\_last\_modified>

# vCard – What?

- vCard stores electronic business cards
- Structured and qualified names, roles, affiliations, addresses, electronic addresses, telephone numbers, etc.
- Developed by the Internet Mail Consortium – currently at version 3
- Serialization as a MIME directory profile specified in RFC2426 (Internet Standard Protocols)

# vCard in RDF

- An XML/RDF serialization of the vCard data model
- As of February 2001 a "note" with the W3C
- This note includes examples mixing Dublin Core and vCard elements
- Defines a W3C namespace for vCard predicates

# vCard in RDF

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:vCard="http://www.w3.org/2001/vcard-rdf/3.0#">
  <rdf:Description about="#hh_1952">
    <dc:creator rdf:parseType="Resource">
      <vCard:FN>Warren Hedley</vCard:FN>
      <vCard:N rdf:parseType="Resource">
        <vCard:Family>Hedley</vCard:Family>
        <vCard:Given>Warren</vCard:Given>
      </vCard:N>
      <vCard:EMAIL rdf:parseType="Resource">
        <rdf:value>w.hedley@auckland.ac.nz</rdf:value>
        <rdf:type
          rdf:resource="http://www.w3.org/2001/vcard-rdf/3.0#internet"/>
      </vCard:EMAIL>
    </dc:creator>
  </rdf:Description>
</rdf:RDF>
```

# Bibliographic Query Service – What ?

- There are currently no XML-based citation standards. Docbook is too complex
- BQS provides an elegant data model for storing citation information and querying it. Includes citing by database ID
- Specification developed by the European Bioinformatics Institute under the OMG umbrella
- Aimed at CORBA-based querying of citation databases
- Serialization of data model in IDL

# BQS in CellML

- The CellML Metadata specification defines an XML serialization of the BQS data model
- It uses Dublin Core and vCard where appropriate
- The namespace is in the cellml.org domain
- Has been sent to the BQS authors, may evolve separately from the rest of CellML metadata

# BQS in CellML

- Jafri, M.S., Rice, J.J., Winslow, R.L. "Cardiac Ca<sup>2+</sup> dynamics: the role of ryanodine receptor adaptation and sarcoplasmic reticulum load" (1998) *Biophys J* 74: 1149-1168.
- 71 lines = 38 RDF triples

```
<rdf:RDF
  xmlns:bqs="http://www.cellml.org/bqs/1.0#"
  xmlns:vCard="http://www.w3.org/2001/vcard-rdf/3.0#"
  xmlns:dcq="http://purl.org/dc/qualifiers/1.0/"
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
  <rdf:Description about="#jrw_1998">
    <bqs:reference rdf:parseType="Resource">
      <bqs:reference_type
        rdf:resource="http://www.cellml.org/bqs/1.0#JournalArticle" />
      <dc:creator>
        <rdf:Seq>
          <rdf:li rdf:parseType="Resource">
            <bqs:provider_type
              rdf:resource="http://www.cellml.org/bqs/1.0#Person" />
            <rdf:value rdf:parseType="Resource">
              <vCard:N rdf:parseType="Resource">
                <vCard:Family>Jafri</vCard:Family>
                <vCard:Given>M</vCard:Given>
                <vCard:Other>S</vCard:Other>
              </vCard:N>
            </rdf:value>
          </rdf:li>
          <rdf:li rdf:parseType="Resource">
            <bqs:provider_type
              rdf:resource="http://www.cellml.org/bqs/1.0#Person" />
            <rdf:value rdf:parseType="Resource">
              <vCard:N rdf:parseType="Resource">
                <vCard:Family>Rice</vCard:Family>
                <vCard:Given>J</vCard:Given>
                <vCard:Other>J</vCard:Other>
              </vCard:N>
            </rdf:value>
          </rdf:li>
        </rdf:Seq>
      </dc:creator>
      <dc:title>
        Cardiac Ca2+ dynamics: the role of ryanodine receptor
        adaptation and sarcoplasmic reticulum load
      </dc:title>
      <dc:date rdf:parseType="Resource">
        <dcq:dateType>issued</dcq:dateType>
        <dcq:dateScheme>W3C-DTF</dcq:dateScheme>
        <rdf:value>1998</rdf:value>
      </dc:date>
      <bqs:journal rdf:parseType="Resource">
        <dc:title>Biophysical Journal</dc:title>
        <bqs:abbreviation rdf:parseType="Resource">
          <bqs:abbreviation_scheme>Medline</bqs:abbreviation_scheme>
          <rdf:value>Biophys J</rdf:value>
        </bqs:abbreviation>
        </bqs:journal>
        <bqs:volume>74</bqs:volume>
        <bqs:first_page>1149</bqs:first_page>
        <bqs:last_page>1168</bqs:last_page>
      </bqs:reference>
    </rdf:Description>
  </rdf:RDF>
```

# Miscellaneous Metadata

- For the annotation, biological and mathematical metadata we need to store there are no widely used standards
- We created elements in the CellML Metadata namespace
- Use type/encoding schemes similar to the Dublin Core for controlled vocabularies, e.g. SWISS-PROT, Genbank

# Approach 3 – Bringing it together

- Use RDF to identify subjects, objects and predicates
- Use standards wherever possible
  - Dublin Core for general categories
  - vCard for personal information
  - BQS for citation information
- Use CellML Metadata elements as a last resort

# Approach 3 – Why it's bad

- RDF – "More than one way to do thing" makes parsing and validating a pain
- RDF – extreme verbosity = high bandwidth
- RDF – the spec induces extreme drowsiness, which puts off implementors and authors alike
- The XML serialization is not intuitive

## Approach 3 – Why it's good

- Non-CellML capable software can "understand" the metadata in CellML documents
- Other XML languages can use all or part of the CellML metadata model (e.g., SBML)
- Tools exist for parsing and validating RDF
- RDF can be still transformed to other languages if need be
- I don't have to write a complete spec

# Links

- CellML – <http://www.cellml.org/>
- CellML Metadata –  
<http://www.cellml.org/public/metadata/>
- RDF – <http://www.w3.org/RDF/>
- Dublin Core – <http://www.dublincore.org/>
- vCard – <http://www.imc.org/pdi/>
- BQS – <http://www.omg.org/lsr/>