DSpace History System: Descriptive Note

Jason A. Kinner

14 May 2003

Contents

1 Introduction

1.1 DSpace

DSpace is a digital library system that contains metadata about varous works within one or more collections within a library.

1.2 History System

The History System describes how the content and metadata associated with a work changes over time. A key feature of the History System is that these changes are noted using the Resource Description Framework [7]. RDF models are serialized as XML according to the RDF-XML syntax specification [5]. The current history system uses RDF as a model for generating XML that is stored in order to track the history of a managed item, but it does not use facilities such as RDF-Schema [4] to describe the models that are used to track this history.

1.3 About This Document

This document is the first deliverable of the History System Statement of Work [10]. Its goal is to describe the current operation of the History System, to capture any outstanding issues with its operation, and to recommend modifications or enhancements to address these issues. The issues and recommendations identified in this document will be addressed in an upcoming document, the second deliverable of the Statement of Work, which will describe how recommendations will be implemented.

1.4 Motivation

The current implementation of the DSpace History System deviates from known standards and best practices for RDF in several ways:

1. There is no explicit schema by which the output can be identified or validated.
2. The output is inconsistent with the latest revisions of public schemas that identify properties used within the History System
3. The use of RDF to model static and dynamic properties is inconsistent with published best practices within the RDF and Semantic Web communities.

These issues become apparent in any use case in which an external system is attempting to access the History System data. The lack of a schema makes it impossible for the external system to validate models that are being read (or written). Output that is inconsistent with public schemas prevents a system that *does* know the proper schema from being able to interpret the output. Finally, the models that are produced prevent effective queries into the model by failing to follow best practices in RDF modeling. These issues are detailed in the section, Issues and Recommendations.

These high-level issues are driven by the following requirements for the History System.

1.4.1 Accessibility of Metadata

RDQL [6], part of the Jena [3] toolkit, provides a useful mechanism for querying RDF models, including RDF-Schema. Joseki [8] provides a network interface to these query capabilities. In order to perform useful queries against an RDF model, data must be modeled as RDF wherever appropriate. Values stored in the model should be queryable by known RDF query tools, including RDQL. RDQL has a basic type facility, corresponding to RDF, consisting of statements and typed literals — text, numeric, and boolean. Any value that must be parsed by the query engine will be difficult to query in the current implementation. Future implementations of RDQL are expected to be more flexible in this regard, due to the extensibility built into the language via functions.

1.4.2 Validation of Metadata

The SIMILE Research Drivers statement [9] describes the importance of validating metadata models. The current History System implementation produces models that are difficult or impossible to validate against metadata schemas. In addition, the models do not reference any existing schema to perform validation. Existing schemas that have been referenced by the current work are not fully utilized for validation purposes.

2 History System Output

2.1 Namespaces

Namespace usage within the DSpace History System falls into three categories. One usage of a namespace is to include a subset of an existing metadata schema in the usage of the history system. The following namespaces are used in this way:

- `http://www.dspace.org/dublincore/`
- `http://www.dspace.org/harmony/`

The `http://www.dspace.org/dublincore/` namespace represents the elements that have been imported from the Dublin Core Metadata Set [1]. Similarly, the `http://www.dspace.org/harmony/` namespace indicates elements that have been imported from the Harmony ABC [2] metadata set. There is no schema that explicitly identifies the relationships between elements of these namespaces and the namespaces from which they draw their properties.

In another usage, namespaces are used to contain properties or classes. The following namespaces are used in this way:

- `http://www.dspace.org/`
- `http://www.dspace.org/bitstream/`
- `http://www.dspace.org/collection/`
- `http://www.dspace.org/community/`
- `http://www.dspace.org/eperson/`
- `http://www.dspace.org/item/`
- `http://www.dspace.org/personalworkspace/`
- `http://www.dspace.org/workflowitem/`

Used for internal purposes, the `http://www.dspace.org/` is used to contain properties that are required by the implementation of the History System. They are not generally useful metadata. All other namespaces represent a class of resource that is managed by DSpace. Within the history system, RDF models are generated that refer to resources within these namespaces.

For example, a property that applies to an *Item* called `hasPart` would be referred to within the History System as `http://www.dspace.org/item/hasPart`. However, the same property referenced from a *Collection* would be referred to as `http://www.dspace.org/collection/hasPart`. The namespaces are not used to define explicit RDF classes via a schema. URL containment is the only mechanism used to identify that a resource is of a given type within the History System.

The final use of namespaces is to import the RDF namespace, required for the RDF-XML serialization [5] that is used by the History System. The following namespace represents the syntax for RDF:

- `http://www.w3.org/1999/02/22-rdf-syntax-ns#`

2.2 Resources

2.2.1 Collection

Set of *Items* that are logically grouped.

2.2.2 Item

A manifestation of a conceptual work, as described by Harmony [2]. Each *Item* has associated Dublin Core metadata.

2.2.3 Bundle

Although explicitly omitted from the current implemention of the History System, each *Bundle* associated with an *Item* contains the *Bitstreams* that represent a unit of content. A set formatted *Bitstreams* that, when decoded by a compliant piece of software, will provide a representation of the *Item*. Different *Bundles* may refer to different renderings of the same piece of content.

2.2.4 Bitstream

Individual encoded segments of an *Item*. Multiple related *Bitstreams* are collected by a *Bundle*. Different *Bitstreams* could be formatted differently (e.g., Word, RTF, and PDF).

2.2.5 Community

A container for one or more *Collections* that represent items that are of interest to a given user group. The *Community* has associated display metadata.

2.2.6 EPerson

An agent that may perform actions on items in DSpace. By Harmony's definitions, an *Agent*.

2.2.7 Workspace Item

An *Item* that has not yet entered the submission workflow.

2.2.8 Workflow Item

An *Item* that is being processed through the submission workflow.

2.3 Actions

2.3.1 Create

Applies to: Collection, Item, Community, EPerson, Workspace Item, Workflow Item.

2.3.2 Modify

Applies to: Collection, Item, Community, EPerson, Workspace Item, Workflow Item.

2.3.3 Remove

Applies to: Collection, Item, Community, EPerson, Workspace Item, Workflow Item.

2.4 Properties

Any of the following described classes can also have RDF properties, such as `ID`, that may be used for internal purposes. The properties outlined here are the descriptive properties from one of the namespaces referenced from or defined by DSpace.

2.4.1 Collection

DSpace:

- approvers
- collection_id
- copyright_text
- introductory_text
- license
- name
- provenance_description
- reviewers
- short_description
- side_bar_text

Harmony ABC:

- hasPart

2.4.2 Item

Dublin Core:

- contributor
- contributor.author
- date.accessioned
- date.available
- date.issued
- description
- description.abstract
- description.provenance
- description.sponsorship
- format.extent
- format.mimetype
- identifier.uri
- language.iso
- relation.ispartofseries
- subject
- title
- *An Item can also have additional Dublin Core properties, as required; there is no validation of additional properties to ensure that they are valid Dublin Core.*

Harmony ABC:

- hasPart

DSpace:

- in_archive
- item_id
- submitter_id

2.4.3 Bitstream

DSpace:

- bitstream_id
- bitstream_type_id
- checksum
- checksum_algorithm

- description
- name
- size
- source
- user_type_description

2.4.4 Community

DSpace:

- community_id
- copyright_text
- introductory_text
- logo_bitstream_id
- name
- short_description
- side_bar_text

Harmony ABC:

- hasPart

2.4.5 EPerson

DSpace:

- active
- email
- eperson_id
- firstname
- lastname
- phone
- require_certificate

2.4.6 Workspace Item

DSpace:

- collection_id
- item_id
- multiple_files
- multiple_titles

- personal_workspace_id
- published_before
- stage_reached
- *Includes the properties of the referenced* Item.

2.4.7 Workflow Item

DSpace:

- collection_id
- item_id
- multiple_files
- multiple_titles
- owner
- published_before
- state
- workflow_id
- *Includes the properties of the referenced* Item.

3 Issues and Recommendations

3.1 Schematic Issues

3.1.1 Usage of External Schemas

Issue

The Dublin Core Element Set [1] and the Harmony ABC metadata set [2] are currently being used by importing elements from those metadata sets into local namespaces. If an agent understands these metadata sets but does not recognize the DSpace-specific metadata sets, then these properties are not accessible to the agent.

Recommendation

For elements that have the same meaning between the local namespace metadata set and the Harmony ABC or Dublin Core metadata set, the elements within the metadata set can be used directly within the DSpace History System. For properties in the local namespace that do not exist in these broader-purpose metadata sets but are related to them by meaning, the `subPropertyOf` facility within RDF-Schema can be used to define the relationship between the properties. Defining these relationships would broaden the accessibility of the History System metadata from tools that do not understand the properties defined in the `http://www.dspace.org/` domain. If such a tool supports RDF-Schema and inferencing, it would

even have access to properties in that domain, but through a simplified façade.

Before Applying Recommendation

```
<rdf:RDF
  xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'
  xmlns:dc='http://www.dspace.org/dublincore/'
  ...
 >
  <rdf:Description
    rdf:about='http://www.dspace.org/item/1721/174'>
    <dc:language.iso>en_US</dc:language.iso>
    <dc:title>My Title</dc:title>
  </rdf:Description>
</rdf:RDF>
```

After Applying Recommendation (see figure 1)

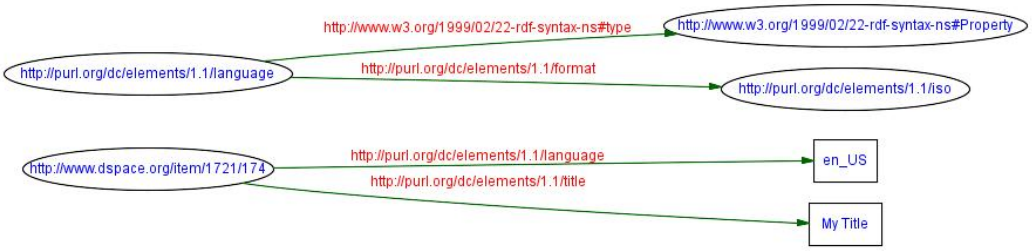
```
<rdf:RDF
  xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'
  xmlns:dc='http://purl.org/dc/elements/1.1/'
  ...
 >
  <rdf:Description
    rdf:about='http://www.dspace.org/item/1721/174'>
    <dc:language>en_US</dc:language>
    <dc:title>My Title</dc:title>
  </rdf:Description>
</rdf:RDF>
<!-- Elsewhere... -->
...
  <rdf:Property
    rdf:about="http://purl.org/dc/elements/1.1/language">
    <dc:format
      rdf:resource="http://purl.org/dc/elements/1.1/iso" />
  </rdf:Property>
...
```

3.1.2 Multiply-Defined Properties

Issue

Certain properties appear in multiple metadata sets. Without a schema to identify the relationships between these properties, a client must have knowledge of all metadata sets in which the property appears. For example, the `hasPart` property appears in many of the object types (*Collection*,

Figure 1: Language Format Specification



Community, *Item*, and *Bitstream*). Since the property is defined in many namespaces, an agent would have a difficult time in querying based on the value of such a property.

Recommendation

In the example above, the `hasPart` should be the Dublin Core `hasPart` throughout the data. In cases where the source vocabulary for the History System do contain redundant definitions, the relationships should be made explicit using RDF-Schema.

3.1.3 Lack of Type Information

Issue

Current History System constructs treat all elements of DSpace as RDF `Resource` objects. An RDF `Resource` is the highest level of abstraction within RDF and does not identify the type of resource being described. The lack of type information within the History System makes the RDF instances impossible to validate against a schema. The current scheme uses a URL fragment to identify the type of the resource, which allows an intelligent client to determine the type of resource, but it does not allow an RDF-aware client that is not aware of the implied schema to operate intelligently on the History System instance data.

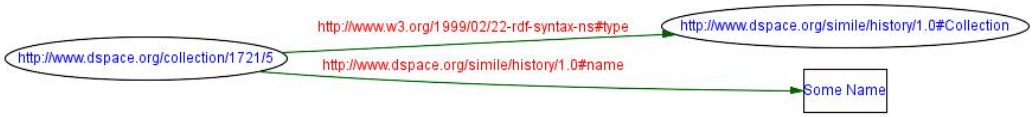
Recommendation

Instead of using the initial part of the URL to identify the type of a resource, the RDF-Schema type facility may be used to promote these resource types to first-class RDF types. This migration would allow DSpace models to be validated against a schema without the need of additional application logic to validate the URL fragment that identifies the type of a resource. Allowing RDF-Schema validation would simplify integration with RDF-compliant tools including repositories and query facilities.

Before Applying Recommendation

```
<rdf:RDF
```

Figure 2: Collection With Type




```
  xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'
  ...
 >
  <rdf:Description
    rdf:about='http://www.dspace.org/collection/1721/5'>
    <dspace:name>Some Name</dspace:name>
  </rdf:Description>
 </rdf:RDF>
```

After Applying Recommendation (see figure 2)

```
<rdf:RDF
  xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'
  ...
 >
  <dspace:Collection
    rdf:about='http://www.dspace.org/collection/1721/5'>
    <dspace:name>Some Name</dspace:name>
  </dspace:Collection>
 </rdf:RDF>
```

3.1.4 Empty or Missing Properties

Issue

Certain properties within the History System data do not contain character data values. This appears to be ambiguous, as it could refer to a missing value (`null` value) or to an empty value (e.g., the empty string).

Recommendation

Missing values (`null` values) should always be encoded using an absent XML element. Empty values should always be encoded using a present, but empty, XML element.

Before Applying Recommendation

```
<rdf:RDF
  xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'
  ...
 >
```

```
 <rdf:Description
   rdf:about='http://www.dspace.org/collection/1721/5'>
   <dspace:name>Some Name</dspace:name>
   <dspace:copyright_text></dspace:copyright_text>
 </rdf:Description>
</rdf:RDF>
```

After Applying Recommendation

```
<rdf:RDF
  xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'
  ...
 >
  <rdf:Description
    rdf:about='http://www.dspace.org/collection/1721/5'>
    <dspace:name>Some Name</dspace:name>
  </rdf:Description>
 </rdf:RDF>
```

3.1.5 Resolution of Namespace URIs

Issue

The current namespace URIs for DSpace (`http://www.dspace.org/` and sub-domains) do not resolve to any descriptive resources.

Recommendation

Current best practices recommend that the namespace URL resolve to a schema or a descriptive definition of the namespace being used. It is recommended that the namespace URLs for DSpace resolve to an RDF-Schema document rendered as RDF-XML.

3.2 URI Encoding Issues

3.2.1 Usage of Handles

Issue

Many of the URIs produced by DSpace represent CNRI Handles. These URIs do not use any common representation for Handle URIs.

Recommendation

Use the Handle URI scheme (`hdl`) for Handles generated by the History System.

3.2.2 Expression of Qualified Properties

Issue

Qualified Dublin Core elements are encoded using the form *element.qualifier*. This usage does not enable a client that understands unqualified Dublin Core to understand the properties, nor does it enable a client that understands qualified Dublin Core to understand those properties.

Recommendation

Dublin Core properties should use the proposed recommended RDF/XML encoding of qualified Dublin Core properties [11]. By conforming to this recommendation, a compliant query engine that supports inferencing would allow a client to query for the basic element (e.g., `description`) and retrieve the value of the qualified property (e.g., `provenance`).

3.2.3 Relationships Expressed Using Local Identifiers

Issue

There are elements of the current history system data that use system-specific identifiers (or "magic numbers") to indicate values of properties that indicate a resource. One such example is the `logo_bitstream_id` property which contains a single numeric value that indicates a relational database key. Using this technique disallows annotating the Bitstream to which the numeric identifier refers.

Recommendation

Items such as the logo bitstream can be represented as a resource (e.g., `http://www.dspace.org/bitstream/359`). This URL could then be used to associate useful metadata to that Bitstream, such as its format. The XML serialization of the RDF would then include an RDF resource reference instead of a literal value, allowing navigation to any metadata that may be associated with the instance.

Before Applying Recommendation

```
<rdf:RDF
  xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'
  ...
 >
  <rdf:Description
    rdf:about='http://www.dspace.org/collection/1721/113'>
    <dspace:logo_bitstream_id>202</dspace:logo_bitstream_id>
  </rdf:Description>
</rdf:RDF>
```

After Applying Recommendation

```
<rdf:RDF
  xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'
  ...
 >
  <rdf:Description
    rdf:about='http://www.dspace.org/collection/1721/113'>
    <dspace:logo_bitstream
      rdf:resource="http://www.dspace.org/bitstream/1721/202 />
  </rdf:Description>
</rdf:RDF>
```

3.2.4 Usage of Local URIs

Issue

The URIs in the History System RDF model are local to the serialization. This prevents the use of a broader RDF repository or database for storage of these models.

Recommendation

All URIs used in the History System output should be globally unique. This can be accomplished by assigning a globally unique URI to the output model, then using that URI as the namespace for local names or by generating a globally unique URI for each statement that is referenced from within the model.

3.2.5 Formatted Text in Property Values

Issue

User annotations of various objects may require formatting. The present formatting is HTML, which is suitable only for Web-based presentation. The HTML is then stored as escaped XML text, providing no facility for validating the contents. These two issues are considered to be out of the scope of the current efforts to normalize the schema of DSpace objects in the History System.

Recommendation

It may be acceptable to have HTML stored in the RDF model of an object. Minimally, the schema or the model should note in what format the contained literal value is encoded. This technique would allow a client to discard the content if it were not in an acceptable format, and would allow a client that recognizes the format to take full advantage of it. Alternatively, an encoding at a higher level of abstraction could be used to encode formatting in the descriptive text associated with an object. This would allow the storage of XML literal values that can be checked for well-formedness before inserting them into the model.

3.3 Harmony Usage Issues

3.3.1 Usage of Outdated Harmony Properties

Issue

The properties that are used in the `http://www.dspace.org/harmony/` namespace are not up to date with the current Harmony ABC properties.

Recommendation

In conjunction with using the official Harmony namespace, the properties used should be brought up to date with version 3 of the schema.

3.3.2 References to Non-existent States

Issue

Serializations of Harmony history data refer to internal states that are encoded using a database identifier, not to prior state information that is also serialized. Harmony ABC provides facilities to encode prior and present states so that previous states are still accessible.

Recommendation

When modeling changes in the properties of a DSpace object, the previous state of any changed properties should be encoded into the model. Modeling prior states provides full versioning of DSpace metadata, allowing for rollback and/or auditing of these properties.

4 Conclusion

Most of the proposed recommendations are within the scope of the current DSpace History System reengineering effort. One notable exception is the encoding of formatted values, such as HTML markup. There is likely a larger content management issue involved in the storage and retrieval of formatted annotations. Whatever the approach, it will need to be applied consistently across all formatted-value properties, which is out of the scope of the current effort.

Although there are many recommendations contained within this note, none of the individual changes is drastic, as indicated in the sample data that appears after the various recommendations. These changes significantly impact the discoverability and usability of DSpace History System metadata by providing multiple levels of complexity and increased consistency across DSpace metadata.

References

[1] Dublin Core Metadata Initiative. `http://dublincore.org/`.

[2] Harmony Metadata Set. http://metadata.net/harmony.

[3] Jena. http://www.hpl.hp.com/semWeb/jena.

[4] RDF Vocabulary Description Language 1.0: RDF Schema. http://www.w3.org/TR/rdf-schema/.

[5] RDF/XML Syntax Specification (Revised). http://www.w3.org/TR/rdf-syntax-grammar/.

[6] RDQL - RDF Data Query Language. http://www.hpl.hp.com/semweb/rdql.htm.

[7] Resource Description Framework (RDF) Model and Syntax Specification. http://www.w3.org/TR/REC-rdf-syntax/.

[8] The Jena RDF Server. http://www.joseki.org/.

[9] Mick Bass and Mark H. Butler. SIMILE Research Drivers. http://web.mit.edu/simile/www/resources/researchDrivers-0.27/.

[10] Jason A. Kinner. History System Statement of Work. http://web.mit.edu/simile/www/resources/history-harmony/history-stateme%nt-of-work.htm.

[11] Stefan Kokkelink and Roland Schwänzl. Expressing Qualified Dublin Core in RDF/XML. http://www.dublincore.org/documents/2002/05/15/dcq-rdf-xml/.