DSpace History System: RDF Schema Design

Jason A. Kinner

28 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 [5]. RDF models are serialized as XML according to the RDF-XML syntax specification [4]. 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 [3] 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 [7]. 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 About This Document

This document is the second in the series of documentation about the DSpace History System. It is a deliverable item from the History System Statement of Work [7]. The purpose of this document is to describe the design and architecture changes that should be made to the History System in order to resolve the issues identified in the first document, the DSpace History System: Descriptive Note [6].

The updated History System will produce RDF instances that will be compliant and consistent with the classes and properties described in this document. Appendix C describes these classes and properties in the RDF Schema Language [3].

2 Functional Changes

This section outlines the changes to be made to the History System that may impact its clients.

2.1 Identification Schemes

The following identification schemes will be used within the History System. Existing identification schemes that are not listed here are to be replaced with an appropriate choice from the following schemes.

2.1.1 URIs

URIs will be used throughout the History System and represent the primary identification mechanism. All other identification schemes used within the History System will represent a subset of this scheme. Using URIs as the primary identification scheme makes it possible to annotate resources using RDF.

2.1.2 URLs

URLs may be used to refer to digital manifestations of a resource that are, in fact, accessible via the URL. An example of this may be a bitstream that represents a document, image, or other tangible digital resource.

2.1.3 URNs

URNs may be used to refer to a resource that does not have a digital manifestation or to a resource whose manifestation is not usable outside of DSpace or the History System. Examples include using an MD5 or SHA hash algorithm to generate an identifier for a bitstream that may not be disseminated by DSpace or an internal identifier that cannot be interpreted outside of DSpace.

2.1.4 Handles

CNRI Handles [8] (Handles) will be used to identify all resources maintained within DSpace. Using Handles will allow resources to be uniquely identified locally, within a DSpace installation, and globally, between DSpace installations. Handles have the property that they may be resolved using a Handle Resolution Service. Although this feature may be useful in the future, it is not a requirement for the usage of Handles that the Handle be resolvable. In fact, Handles may not generally be resolvable due to issues such as publishing scope and security constraints.

The Handle URI scheme is described in [8], section 4.3, and is as follows:

HandleURI = **"hdl:"** *<NamingAuthority>* **"/"** *<LocalName>*

Because the History System need not be concerned about the resolution of Handles, they will be treated effectively as URNs.

2.2 Inferencing Support

Inferencing refers to the resolution of a statement within the context of a set of constraints on a given model. Within the History System, these constraints are stated using RDF-Schema [3]. Other languages for expressing these constraints exist, including DAML and OWL. For the present scope of work, however, RDF-Schema offers enough descriptive power to provide utility to a client. RDF-Schema allows relatively weak statements of inheritence via the concepts of sub-classes and sub-properties.

A sub-class represents the definition of a type of resource that is considered to contain the same properties as its super-class, plus its own optional properties. A sub-property represents an attribute that is considered to contain the same descriptive semantics as its super-property, plus its own optional semantics. When RDF statements describing these schematic relationships are combined with statements describing instances of these classes and properties, additional statements regarding the instances can be inferred from the schematic statements. This inference is what can make it possible for a client that understands how to operate on one metadata vocabulary to interoperate with another client that operates on a more specific vocabulary. For example, the Dublin Core Element Set defines the *Date* property. This property can have a refinement, or sub-property, *Created*. Using inference, the relationship between these properties allow the *Created* date to be accessed by a client that understands only the *Date* property. As long as the simpler client is not permitted to write a *Date* property value to a *Created* property slot, the two clients can safely operate on values of these properties.

2.2.1 Use of External Schemas

In order to take full advantage of RDF-Schema inferencing, appropriate use must be made of external public schemas. Presumably, clients will exist that will understand the properties defined in these schemas. There are two external schemas that provide a framework for the History System.

Dublin Core Metadata Element Set

> The Dublin Core Metadata Element Set is a standard metadata vocabulary consisting of 15 terms that are commonly used to describe various works. These terms are general enough to be widely applicable across content types. They form the core of the descriptive metadata both within History System and within DSpace in general.

Harmony ABC

> Harmony ABC defines a set of classes and properties that relate to describing the state of metadata property values over time. Although this vocabulary does describe resource (as any metadata vocabulary does), it is not meant to be a user-accessible or user-defined metadata vocabulary. It is optimized not for human readability, but for automation that needs

to maintain a history of property values and the reasons for their changes over time.

2.2.2 Use of RDF Types

The History System will be redesigned to make heavy use of the schemas described above, as well as the schema that describes the classes and properties required to maintain information related to DSpace and to the History System (administrative metadata). The use of RDF types will improve the capability of the system to validate and to infer relationships between instance statements during queries.

3 Updated RDF Model

3.1 External Schemas

3.1.1 Dublin Core Metadata Element Set

The new History System RDF model will directly use elements of the Dublin Core Element Set (DCES). The schema for the model will also reference this element set via the RDF-Schema `subPropertyOf` relationship whenever appropriate for defining DSpace-specific elements. When making this determination, query syntax and semantics will be considered.

Namespace `http://purl.org/dc/elements/1.1/`

Examples `title`, `date`, `format`, `identifier`, `relation`

3.1.2 DCMI Metadata Terms

The new History System RDF model will directly use elements of the Dublin Core Metadata Initiative (DCMI) Metadata Terms definitions. When appropriate, the DCMI Terms may be referenced by the schema for the model via `subPropertyOf` relationships. When making this determination, query syntax and semantics will be considered.

Namespace `http://purl.org/dc/terms/`

Examples `abstract`, `created`, `available`, `isVersionOf`

3.1.3 DCMI Encoding Schemes

The new History System RDF model will directly use the classes defined in the DCMI Encoding Schemes where appropriate to describe well-known formats. Where needed, these classes will be referenced by the History System RDF-Schema via `subClassOf` relationships.

Namespace `http://purl.org/dc/elements/1.1/`

Examples `URI`, `ISO3166`, `W3CDTF`, `RFC3066`

3.1.4 Harmony ABC

The new History System RDF model will directly use the classes and properties defined in the Harmony ABC metadata vocabulary. This metadata schema will provide the foundation of encoding relationships between the states of DSpace objects as they change over time. The primary impact of using Harmony ABC as the foundation of the History System temporal encoding scheme will be to clients that wish to query along this axis. Although Harmony ABC is not an official standard, there is value in it being a publicly-available and fairly well-known metadata schema. Harmony ABC also has the architectural advantage that it does not require a specific metadata schema to be used to encode the actual state of resources describes using its properties.

Namespace `http://metadata.net/harmony#`

Examples `Event`, `Situation`, `precedes`, `follows`

3.1.5 DSpace History

The new History System RDF model will defined classes and properties in a common namespace. This namespace will be related to public metadata vocabularies whenever appropriate via `subPropertyOf` or `subClassOf` relationships.

Namespace `http://dspace.org/history/1.0#`

Examples `Collection`, `Item`, `reviewer`, `accessioned`

3.2 Classes

3.2.1 DSpace Classes

The following classes represent the types of objects that the History System must be able to represent from within DSpace.

Community

: A *Community* represents a set of *Collections* that are logically grouped.

Collection

: A *Collection* represents a set of *Items* that are logically grouped.

Item

: An *Item* represents an asset that is managed by DSpace.

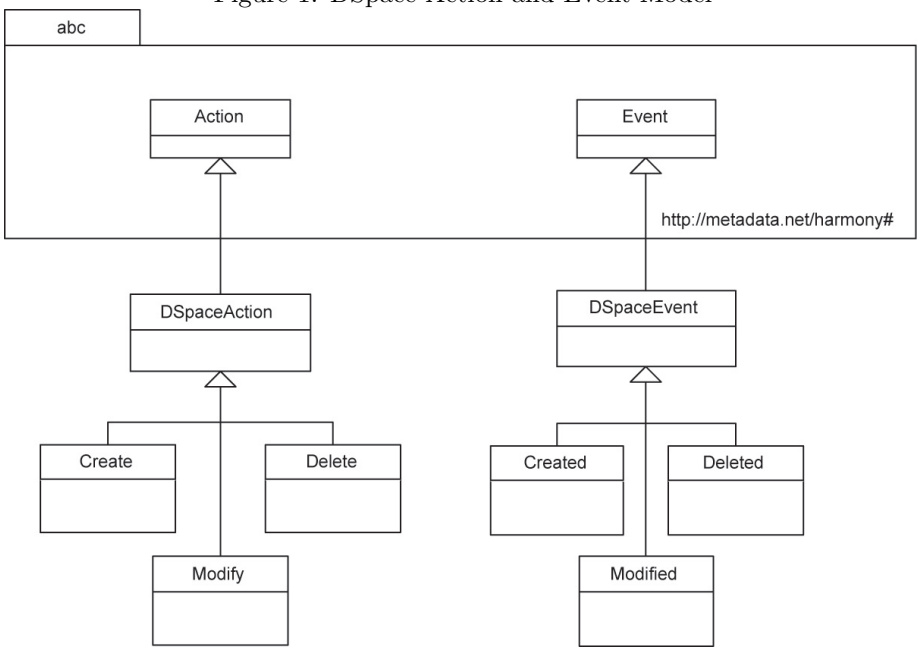
Bundle

: A *Bundle* represents all of the related *Bitstreams* that are required to render a manifestation of an *Item*.

Bitstream

: A *Bitstream* represents a sequence of bits that represent a piece of content.

Figure 1: DSpace Action and Event Model



WorkspaceItem

A *WorkspaceItem* represents an *Item* that has been recognized but has not yet begun the ingest workflow to be officially recognized by DSpace. When a DSpace administrator begins the ingest workflow, an object of this type will be represented as a *WorkflowItem*.

WorkflowItem

A *WorkflowItem* represents an *Item* that has begun the process of ingest into DSpace but has not been cataloged. When the ingest workflow is completed, an object of this type will be represented as an *Item*.

EPerson

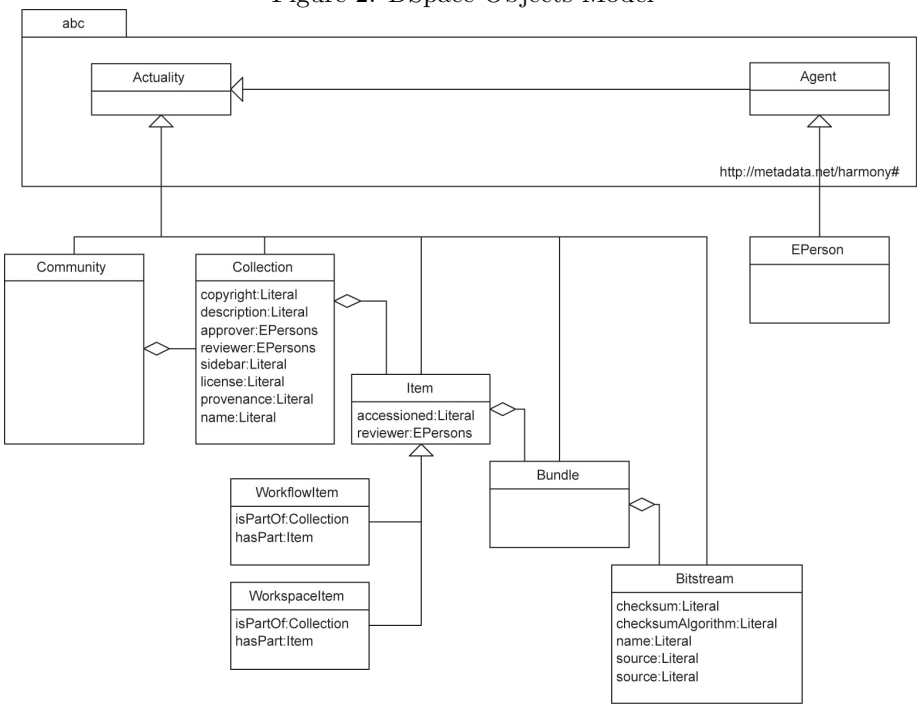
An *EPerson* represents a user of DSpace.

3.2.2 History Objects

The following classes represent internal object types that will be used as extensions to the Harmony ABC schema to represent DSpace-specific state transition information.

DSpaceEvent

Figure 2: DSpace Objects Model



A *DSpaceEvent* is a sub-class of the Harmony ABC class `Event`. All DSpace events that the History System maintains are sub-classes of this class.

Created

A *DSpaceEvent* that signifies the creation of a DSpace object.

Modified

A *DSpaceEvent* that signifies the modification of a DSpace object.

Deleted

A *DSpaceEvent* that signifies the deletion of a DSpace object.

DSpaceAction

A *DSpaceAction* is a sub-class of the Harmony ABC class `Action`. All DSpace actions that the History System maintains are sub-classes of this class.

Creates

A *DSpaceAction* that causes the creation of a DSpace object.

Modifies

A *DSpaceAction* that causes the modification of a DSpace object.

Deletes

A *DSpaceAction* that causes the deletion of a DSpace object.

EPerson

An *EPerson* represents a user of DSpace. In the context of history objects, an *EPerson* can act as a Harmony ABC *Agent*.

3.3 Properties

3.3.1 Dublin Core Elements [1]

`created`

Date of creation of the resource.

`modified`

Date on which the resource was changed.

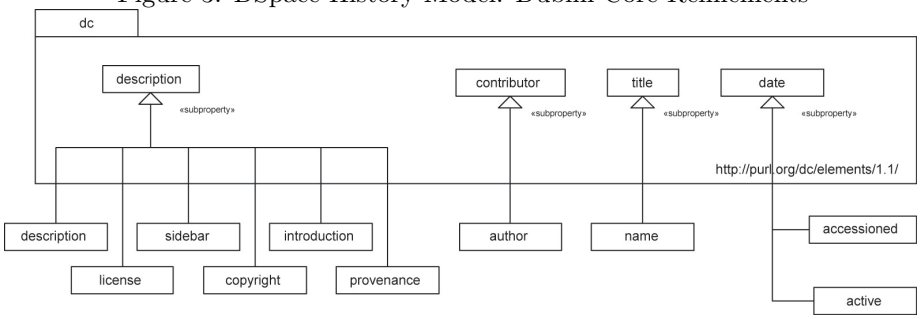
`description`

An account of the content of the resource.

`hasPart`

The described resource includes the referenced resource either physically or logically.

Figure 3: DSpace History Model: Dublin Core Refinements



`format`

The physical or digital manifestation of the resource.

3.3.2 Harmony ABC Elements [2]

`precedes`

Binds a *Situation* and the *Actualities* within its context as existing before an *Event*.

`follows`

Binds a *Situation* and the *Actualities* within its context as existing after an *Event*.

`hasParticipant`

Refines hasPresence to associate an Agent as an active participant in an Event or Action.

`destroys`

A specialization of `hasPatient` that indicates that the value *Actuality* ceases to exist in *Situation*(s) that follow the *Event*.

`atTime`

Associates a time with an entity that is a subcategory of a *Temporality*.

`phaseOf`

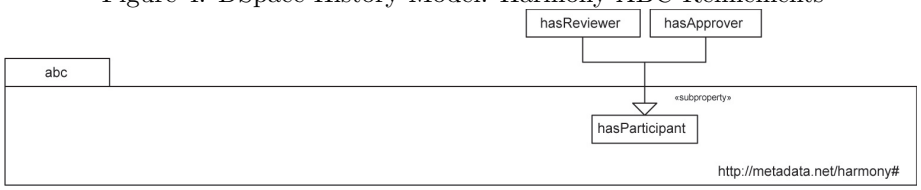
Establishes the relationship between an existential facet and a universal facet of an *Actuality*.

3.3.3 History System Elements

`accessioned`

Date on which the item was introduced into DSpace.

Figure 4: DSpace History Model: Harmony ABC Refinements



`active`

Date on which the user became active in the DSpace installation.

`hasApprover`

Person or group that participated in an approval by certifying the content of an item.

`author`

Person or group that participated in the creation of the content of an item.

`checksum`

String that represents the sequence of bits in a *Bitstream*.

`checksumAlgorithm`

Algorithm that was used to generate the `checksum`

`copyright`

Description of the copyright terms of the item.

`description`

Brief description to be used for display.

`email`

Email address of a person or group.

`firstname`

Given name of a person.

`generated`

Indicates a component that was generated by a software agent. Inverse of `hasGenerator`.

`hasGenerator`

Software agent that generated the change model.

`introduction`

Brief description that introduces the content.

`lastname`

Family name of a person.

`license`

Description of the license terms governing the use of the item.

`name`

Full name of a person.

`phone`

Telephone number where a person can be reached.

`provenance`

Textual description of the history of possession of the item.

`hasReviewer`

Person or group that participated in an approval by reviewing the content of an item.

`sidebar`

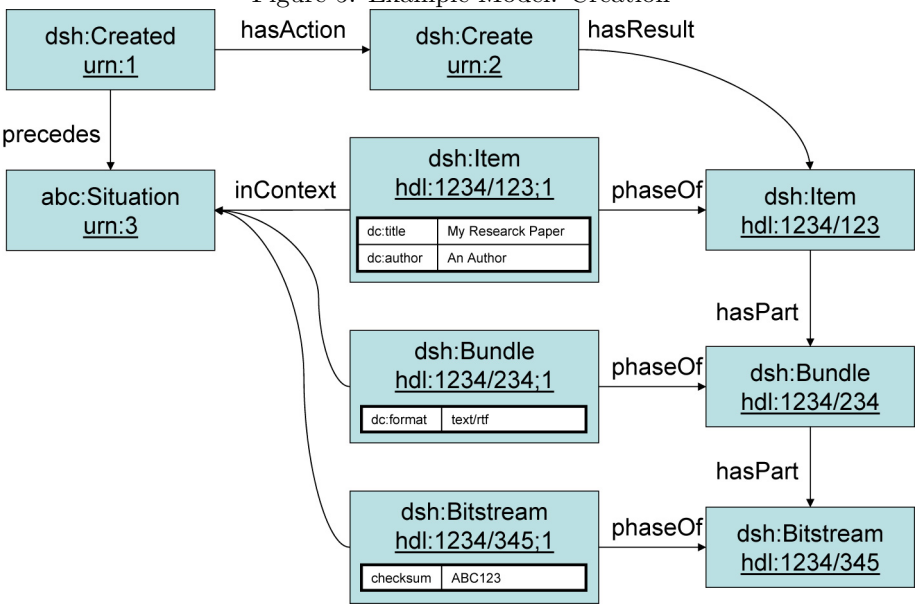
Information that is to be displayed in parallel with the `description` or `introduction` content.

A Example

Consider a fictional research paper that is submitted to DSpace. When the paper is submitted to DSpace, a *Created* event occurs with the associated *Create* action that indicates what has been created. In particular, the result of the *Create* action is a complete *Item* with associated *Bundle* and *Bitstream* (see figure 5). All of these created items have associated properties that are represented in the History System model of the event. The resting state of the model is that there is an *Item*, *Bundle*, and *Bitstream* (the result of the *Create* action) as well as associated *Item*, *Bundle*, and *Bitstream* revision instances that represent the current state of the properties of these created objects. Each of these revision instances are in the context of the initial check-in *Situation*. Note that the revision instances are not related via the `hasPart` relationship, as the resources themselves are. This type of property is referred to in Harmony ABC as a universal facet. This model decouples the changes in one object from the changes in another object. It also allows for the case that a new *Item* is created that represents specific revisions of the *Bundle* or *Bitstream* (for example, an edition of a work).

The example so far reveals only a static example. Consider what would happen to this model if a property of the *Item* were to change. The History System represents this change as additional objects and property values in this model (see figure 6). The change requires the *Item* to have a new revision instance

Figure 5: Example Model: Creation



created (indicated in the figure by the URI, `hdl:1234/123;2`) that contains the new value for the Dublin Core `title` property. An administrative property, `phaseOf`, links this new revision instance back to the root *Item* resource. It is important to note that since the `hasPart` property has not changed, it is not represented in the change model.

B Example Queries

Based on the example above, the model can be queried for useful information. Due to the rich type information stored in the model, these queries can be very precise. If the query mechanism supports inferencing via the RDF-Schema (see appendix C), then simplified queries can also provide useful results. Namespaces have been replaced with common prefixes in the output results for clarity.

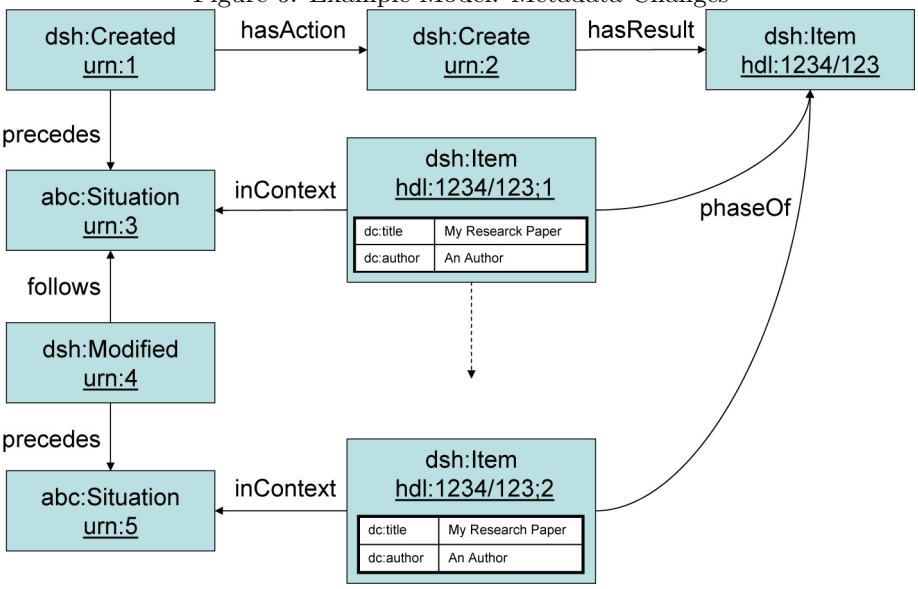
1. **Question**

   What is the entire history of every resource in the system?

   Query

```
select ?revision, ?property, ?value
where (?revision, <abc:inContext>, ?situation),
      (?revision, <abc:phaseOf>, ?actuality),
      (?revision, ?property, ?value)
and   ?property !~ /http:\/\/www.metadata.net\/harmony#/
```

Figure 6: Example Model: Metadata Changes




```
and   ?property !~ /http:\/\/www.w3.org\/1999\/02\/22-rdf-syntax-ns#/
using abc for <http://www.metadata.net/harmony#>
      rdf for <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
```

Results

```
revision           | property     | value
=======================================================
<hdl:1234/123;2>  | <dc:author>  | "An author"
<hdl:1234/123;1>  | <dc:author>  | "An author"
<hdl:1234/123;2>  | <dc:title>   | "My Research Paper"
<hdl:1234/123;1>  | <dc:title>   | "My Researck Paper"
<hdl:1234/123;2>  | <dc:hasPart> | <hdl:1234/234>
<hdl:1234/123;1>  | <dc:hasPart> | <hdl:1234/234>
```

2. **Question**

   What is temporal order of all of the *Situations* for a given *Actuality*?

   Query

```
select ?first, ?second
where (?event, <abc:follows>, ?first),
      (?event, <abc:precedes>, ?second),
      (?event, <abc:involves>, ?actuality)
and   ?actuality eq <hdl:1234/123>
using abc for <http://www.metadata.net/harmony#>
      rdf for <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
```

Results

```
first   | second
================
<urn:3> | <urn:5>
```

NB: Because of the structure of the Harmony ABC model, this result represents a linked list with one arc per result item.

3. **Question**

   Who participated in any action for any resource?

   Query

```
select ?actuality, ?participant
where (?event, <abc:involves>, ?actuality),
      (?event, <abc:hasAction>, ?action),
      (?action, <rdf:type>, <abc:Action>),
      (?action, <abc:hasParticipant>, ?participant)
using abc for <http://www.metadata.net/harmony#>
      rdf for <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
      rdfs for <http://www.w3.org/2000/01/rdf-schema#>
      dsh for <http://dspace.org/history/1.0#>
```

   Results

```
actuality      | participant
============================
<hdl:1234/123> | <eperson:123>
<hdl:1234/234> | <eperson:123>
<hdl:1234/345> | <eperson:123>
```

   NB: This query requires inferencing support given the example model and the query parameters. The URI <`abc:hasParticipant`> could be replaced with an appropriate, more specific property if inferencing is not supported by the query engine.

C Schema Definition

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF
  xml:base="http://dspace.org/history/1.0#"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
  <!--
       DSpace History Classes
    -->
  <rdfs:Class rdf:ID="Collection">
```

```
  <rdfs:subClassOf
    rdf:resource="http://metadata.net/harmony#Actuality"/>
</rdfs:Class>
<rdfs:Class rdf:ID="Item">
  <rdfs:subClassOf
    rdf:resource="http://metadata.net/harmony#Actuality"/>
</rdfs:Class>
<rdfs:Class rdf:ID="Bundle">
  <rdfs:subClassOf
    rdf:resource="http://metadata.net/harmony#Actuality"/>
</rdfs:Class>
<rdfs:Class rdf:ID="Bitstream">
  <rdfs:subClassOf
    rdf:resource="http://metadata.net/harmony#Actuality"/>
</rdfs:Class>
<rdfs:Class rdf:ID="Community">
  <rdfs:subClassOf
    rdf:resource="http://metadata.net/harmony#Actuality"/>
</rdfs:Class>
<rdfs:Class rdf:ID="EPerson">
  <rdfs:subClassOf
    rdf:resource="http://metadata.net/harmony#Agent"/>
</rdfs:Class>
<rdfs:Class rdf:ID="WorkspaceItem">
  <rdfs:subClassOf
    rdf:resource="#Item"/>
</rdfs:Class>
<rdfs:Class rdf:ID="WorkflowItem">
  <rdfs:subClassOf
    rdf:resource="#Item"/>
</rdfs:Class>
<!--
     DSpace History Properties
  -->
<rdf:Property rdf:ID="copyright">
  <rdfs:comment>
    The copyright notice for the
    work or manifestation.
  </rdfs:comment>
  <rdfs:subPropertyOf
    rdf:resource="http://purl.org/dc/elements/1.0/description"/>
</rdf:Property>
<rdf:Property rdf:ID="introduction">
  <rdfs:comment>
    Introductory text for a document.
  </rdfs:comment>
```

```
  <rdfs:subPropertyOf
    rdf:resource="http://purl.org/dc/elements/1.0/description"/>
</rdf:Property>
<rdf:Property rdf:ID="name">
  <rdfs:comment>
    The name of a work or collection.
  </rdfs:comment>
  <rdfs:subPropertyOf
    rdf:resource="http://purl.org/dc/elements/1.0/title"/>
</rdf:Property>
<rdf:Property rdf:ID="logo"/>
<rdf:Property rdf:ID="provenance">
  <rdfs:subPropertyOf
    rdf:resource="http://purl.org/dc/elements/1.0/description"/>
</rdf:Property>
<rdf:Property rdf:ID="license">
  <rdfs:comment>
    A textual description of the usage rights granted to the
    recipient of the license.
  </rdfs:comment>
  <rdfs:subPropertyOf
    rdf:resource="http://purl.org/dc/elements/1.0/description"/>
</rdf:Property>
<rdf:Property rdf:ID="sidebar">
  <rdfs:subPropertyOf
    rdf:resource="http://purl.org/dc/elements/1.0/description"/>
</rdf:Property>
<rdf:Property rdf:ID="description">
  <rdfs:comment>A brief description of the work.</rdfs:comment>
  <rdfs:subPropertyOf
    rdf:resource="http://purl.org/dc/elements/1.0/description"/>
</rdf:Property>
<rdf:Property rdf:ID="isGeneratedBy">
  <rdfs:comment>
    Asserts that a given subject generated a given
    statement or graph.
  </rdfs:comment>
</rdf:Property>
<rdf:Property rdf:ID="hasGenerator">
  <rdfs:comment>
    The entity that generated the current instance.
  </rdfs:comment>
</rdf:Property>
  <rdf:Property rdf:ID="checksum">
  <rdfs:range
    rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
```

```
  </rdf:Property>
  <rdf:Property rdf:ID="checksumAlgorithm">
    <rdfs:range
      rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
  </rdf:Property>
  <rdf:Property rdf:ID="email"/>
  <rdf:Property rdf:ID="firstname"/>
  <rdf:Property rdf:ID="lastname"/>
  <rdf:Property rdf:ID="phone"/>
  <rdf:Property rdf:ID="active">
    <rdfs:subPropertyOf
      rdf:resource="http://purl.org/dc/elements/1.1/date"/>
  </rdf:Property>
  <rdf:Property rdf:ID="hasApprover">
    <rdfs:subPropertyOf
      rdf:resource="http://metadata.net/harmony#hasParticipant"/>
  </rdf:Property>
  <rdf:Property rdf:ID="hasReviewer">
    <rdfs:subPropertyOf
      rdf:resource="http://metadata.net/harmony#hasParticipant"/>
  </rdf:Property>
  <rdf:Property rdf:ID="author">
    <rdfs:subPropertyOf
      rdf:resource="http://purl.org/dc/elements/1.1/contributor"/>
  </rdf:Property>
  <rdf:Property rdf:ID="accessioned">
    <rdfs:comment>
      The date on which the resource was initially checked
      into DSpace.
    </rdfs:comment>
    <rdfs:subPropertyOf
      rdf:resource="http://purl.org/dc/elements/1.1/date"/>
  </rdf:Property>
</rdf:RDF>
```

D Example RDF-XML Serialization

The following RDF-XML serialization represents the example data from the diagrams above. It was used to perform RDQL queries for the example queries.

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF
  xml:base="http://dspace.org/harmony/1.0#"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
```

```
  xmlns:abc="http://www.metadata.net/harmony#"
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:dsh="http://www.dspace.org/history/1.0#"
>
  <dsh:Created rdf:about="urn:1">
    <abc:hasAction>
      <dsh:Create rdf:about="urn:2">
        <abc:hasResult rdf:resource="hdl:1234/123"/>
        <dsh:hasReviewer rdf:resource="eperson:123"/>
      </dsh:Create>
    </abc:hasAction>
    <abc:involves rdf:resource="hdl:1234/123"/>
    <abc:involves rdf:resource="hdl:1234/234"/>
    <abc:involves rdf:resource="hdl:1234/345"/>
    <abc:precedes rdf:resource="urn:3"/>
  </dsh:Created>
  <abc:Situation rdf:about="urn:3">
  </abc:Situation>
  <dsh:Item rdf:about="hdl:1234/123;1">
    <abc:inContext rdf:resource="urn:3"/>
    <dc:author>An author</dc:author>
    <dc:title>My Researck Paper</dc:title>
    <dc:hasPart>
      <dsh:Bundle rdf:about="hdl:1234/234"/>
    </dc:hasPart>
    <abc:phaseOf rdf:resource="hdl:1234/123" />
  </dsh:Item>
  <dsh:Bundle rdf:about="hdl:1234/234;1">
    <dc:hasPart>
      <dsh:Bitstream rdf:about="hdl:1234/345" />
    </dc:hasPart>
    <abc:phaseOf rdf:resource="hdl:1234/234" />
  </dsh:Bundle>
  <dsh:Bitstream rdf:about="hdl:1234/345;1">
    <dsh:checksum>ABC123</dsh:checksum>
    <abc:phaseOf rdf:resource="hdl:1234/345" />
  </dsh:Bitstream>
  <dsh:Modified rdf:about="urn:4">
    <abc:involves rdf:resource="hdl:1234/123"/>
    <abc:follows rdf:resource="urn:3"/>
    <abc:precedes rdf:resource="urn:5"/>
  </dsh:Modified>
  <abc:Situation rdf:about="urn:5">
  </abc:Situation>
  <dsh:Item rdf:about="hdl:1234/123;2">
    <abc:inContext rdf:resource="urn:5"/>
```

```
    <dc:title>My Research Paper</dc:title>
    <abc:phaseOf rdf:resource="hdl:1234/123" />
    <dc:author>An author</dc:author>
    <dc:hasPart>
      <dsh:Bundle rdf:about="hdl:1234/234"/>
    </dc:hasPart>
  </dsh:Item>
</rdf:RDF>
```

References

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

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

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

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

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

[6] Jason A. Kinner. DSpace History System Descriptive Note. http://web.mit.edu/simile/www/resources/history-harmony/descriptive-not%e.pdf.

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

[8] Sam X. Sun and Larry Lannom. Handle system overview (internet draft). http://www.ietf.org/internet-drafts/draft-sun-handle-system-10.txt.