(2) When do we say PROV identifiers are QNames + say PROV convention is that QNames map to URIs. + note restriction on QNames.

(3) examples miss prefix or default namespace.

PROV-XML: The PROV XML Schema

W3C Working Group Note 30 April 2013

(4) Bundle element should be choice.

This version:
http://www.w3.org/TR/2013/NOTE-prov-xml-20130430/
Latest published version:
http://www.w3.org/TR/prov-xml/
Latest editor's draft:
http://dvcs.w3.org/hg/prov/raw-file/default/xml/prov-xml.html
Previous version:
http://www.w3.org/TR/2013/WD-prov-xml-20130312/ (color-coded diff)

Editors:
(In alphabetical order)
Hook Hua, Invited Expert
Curt Tilmes, National Aeronautics and Space Administration
Stephan Zednik, Rensselaer Polytechnic Institute
Author:
Luc Moreau, University of Southampton

Abstract

Provenance is information about entities, activities, and people involved in producing a piece of data or thing, which can be used to form assessments about its quality, reliability or trustworthiness. PROV-DM is the conceptual data model that forms a basis for the W3C provenance (PROV) family of specifications. It defines a concepts for expressing provenance information enabling interchange. This document introduces an XML schema for the PROV data model (PROV-DM), allowing instances of the PROV data model to be serialized in XML.

The PROV Document Overview describes the overall state of PROV, and should be read before other PROV documents.

(1) ⚠ element → PROV-N "element" or → XML-element

attribute → PROV-N "attribute"

type → PROV type / XML type → XML-attribute.

Status of This Document

This section describes the status of this document at the time of its publication. Other documents may supersede this document. A list of current W3C publications and the latest revision of this technical report can be found in the W3C technical reports index at http://www.w3.org/TR/.

PROV Family of Documents

This document is part of the PROV family of documents, a set of documents defining various aspects that are necessary to achieve the vision of inter-operable interchange of provenance information in heterogeneous environments such as the Web. These documents are listed below. Please consult the [PROV-OVERVIEW] for a guide to reading these documents.

- PROV-OVERVIEW (Note), an overview of the PROV family of documents [PROV-OVERVIEW];
- PROV-PRIMER (Note), a primer for the PROV data model [PROV-PRIMER];
- PROV-O (Recommendation), the PROV ontology, an OWL2 ontology allowing the mapping of the PROV data model to RDF [PROV-O];
- PROV-DM (Recommendation), the PROV data model for provenance [PROV-DM];
- PROV-N (Recommendation), a notation for provenance aimed at human consumption [PROV-N];
- PROV-CONSTRAINTS (Recommendation), a set of constraints applying to the PROV data model [PROV-CONSTRAINTS];
- PROV-XML (Note), an XML schema for the PROV data model (this document);
- PROV-AQ (Note), mechanisms for accessing and querying provenance [PROV-AQ];
- PROV-DICTIONARY (Note) introduces a specific type of collection, consisting of key-entity pairs [PROV-DICTIONARY];
- PROV-DC (Note) provides a mapping between PROV-O and Dublic Core Terms [PROV-DC];
- PROV-SEM (Note), a declarative specification in terms of first-order logic of the PROV data model [PROV-SEM];
- PROV-LINKS (Note) introduces a mechanism to link across bundles [PROV-LINKS].

Implementations Encouraged

The Provenance Working Group encourages implementation of the material defined in this document. Although work on this document by the Provenance Working Group is complete, errors may be recorded in the errata or and these may be addressed in future revisions.

Please Send Comments

This document was published by the Provenance Working Group as a Working Group Note. If you wish to make comments regarding this document, please send them to public-prov-comments@w3.org (subscribe, archives). All comments are welcome.

Publication as a Working Group Note does not imply endorsement by the W3C Membership. This is a draft document and may be updated, replaced or obsoleted by other documents at any time. It is inappropriate to cite this document as other than work in progress.

Table of Contents

1. Introduction

According to Prov-DM

~~For the purpose of this specification~~, ***provenance*** is defined as a record that describes the people, institutions, entities, and activities involved in producing, influencing, or delivering a piece of data or a thing. In particular, the provenance of information is crucial in deciding whether information is to be trusted, how it should be integrated with other diverse information sources, and how to give credit to its originators when reusing it. In an open and inclusive environment such as the Web, where users find information that is often contradictory or questionable, provenance can help those users to make trust judgements.

Family of PROV-DM

The PROV data model, PROV-DM, presents a generic data model for provenance that allows domain and application specific representations of provenance to be translated into such a data model and *interchanged* between systems. Thus, heterogeneous systems can export their native provenance into such a core data model, and applications that need to make sense of provenance can then import it, process it, and reason over it.

This enables PROV-DM to be independent of specific domains.

The PROV data model distinguishes *core structures* from *extended structures*: core structures form the essence of provenance information, and are commonly found in various domain-specific vocabularies that deal with provenance or similar kinds of information [Mappings]. Extended structures enhance and refine core structures with more expressive capabilities to cater for more advanced uses of provenance. The PROV data model, comprising both core and extended structures, is a domain-agnostic model, but with clear extensibility points allowing further domain-specific and application-specific extensions to be defined.

The PROV data model has a modular design and is structured according to six components covering various facets of provenance:

- component 1: entities and activities, and the time at which they were created, used, or ended;
- component 2: derivations of entities from others;
- component 3: agents bearing responsibility for entities that were generated and activities that happened;
- component 4: bundles, a mechanism to support provenance of provenance;
- component 5: properties to link entities that refer to a same thing;
- component 6: collections forming a logical structure for its members.

This specification goal is to provide a succinct definition of the XML form of PROV-DM, thus, we refer the reader to the PROV-DM to provide overall justification and context to the definitions presented here.

1.1 PROV Namespace

The PROV namespace is http://www.w3.org/ns/prov#.

All the concepts, reserved names and attributes introduced in this specification belong to the PROV namespace.

1.2 Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2. XML Schema Design

Several general design principles and patterns were used in the construction of the PROV XML Schema.

2.1 Schema Modularization

The PROV-XML schema have been modularized so that extension elements defined in Working Group Notes can be defined in separate schemas. Elements corresponding to terms defined in the PROV-DM are defined in the **prov-core.xsd** schema and elements corresponding to terms defined in notes are defined in extension schemas (e.g. **prov-dictionary.xsd**, **prov-links.xsd**). The default schema, **prov.xsd**, imports **prov-core.xsd** and all extension schemas developed by the Working Group. With this modeling all PROV elements, even those defined in Notes, are defined from the default schema. If the user wishes to leverage a schema that does not include extension elements the user can use schemaLocation to directly reference **prov-core.xsd**.

The default schema - **prov.xsd**

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema targetNamespace="http://www.w3.org/ns/prov#"
           xmlns:xs="http://www.w3.org/2001/XMLSchema"
           xmlns:prov="http://www.w3.org/ns/prov#"
           elementFormDefault="qualified"
           attributeFormDefault="unqualified">

        <xs:include schemaLocation="prov-core.xsd"/>
        <xs:include schemaLocation="prov-dictionary.xsd"/>
        <xs:include schemaLocation="prov-links.xsd"/>
```

```
</xs:schema>
```

Extension schemas import the **prov-core.xsd** schema and make use of a substitution group on the `prov:internalElement` to add extension-defined elements to the list of valid PROV elements in a bundle or document.

> NOTE
>
> This schema design leveraging substitutionGroups on an abstract element may result in sub-optimal binding classes being generated by OXM frameworks such as JAXB, JiBX, etc. See the PROV FAQ entry at How should I generate JAXB classes from the PROV-XML schemas? for a JAXB-specific discussion on using OXM technologies with the PROV schemas.

The **prov-links.xsd** extension schema.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
        targetNamespace="http://www.w3.org/ns/prov#" xmlns:prov="http://www.w3.org/ns/prov#"
        elementFormDefault="qualified">

        <xs:include schemaLocation="prov-core.xsd" />

        <xs:complexType name="Mention">
                <xs:sequence>
                        <xs:element name="specificEntity" type="prov:IDRef" />
                        <xs:element name="generalEntity" type="prov:IDRef" />
                        <xs:element name="bundle" type="prov:IDRef" />
                </xs:sequence>
        </xs:complexType>

        <xs:element name="mentionOf" type="prov:Mention" substitutionGroup="prov:internalElemer

</xs:schema>
```

All schemas developed by the PROV WG utilize the PROV namespace.

2.2 *Salami Slice* Design Pattern

The general design pattern for the XML schema has been called *Salami Slice Design*. With this design, the individual components are each defined at the top level as separate elements with distinct types. This allows the types to be easily reusable for domain specific extensions.

and elements

The `prov:document` element has been defined to act as a convenient root element for a PROV-XML document, but its use as the root element is not required. The schema follows the Salami Slice pattern to ensure PROV XML elements can be integrated with mixed-schema XML documents which require a different document root.

2.3 Elements vs. Attributes

The general PROV-N syntax patterns for expressing provenance concepts are:

```
thing(id, elem1, elem2, ..., [attr1=val1, attr2=val2])
concept(id; elem1, elem2, ... [attr1=val2, attr2=val2])
```

In both cases (required id or optional id), the PROV-N id is treated as an XML attribute (`prov:id`), the PROV-N "elements" are treated as XML elements, always with the same required order (position) as the PROV-DM/PROV-N description, and optional PROV-N "attributes", if allowed, always follow and are also represented by XML elements. As in PROV-N, the "attributes" can be specified multiple times, but unlike PROV-N the

attributes have a fixed (alphabetical) order. The PROV-N "attribute" elements are always defined at the end of the encompassing sequence after all PROV-N "elements". Elements defined in namespaces other than the PROV namespace may be included in an element after all PROV elements.

Wherever an "id" is referenced from a later concept, the id is referenced as a prov:ref attribute of the element within the concept.

This transformation technique yields a general XML pattern:

```
<prov:thing prov:id="id">
  <prov:elem1 />
  <prov:elem2 />
  ...
  <ex:attr1>val1</ex:attr1>
  <ex:attr2>val2</ex:attr2>
  ...
</prov:thing>
```

Most of the concepts described below follow this general pattern.

2.4 Type Conventions

2.4.1 PROV Type Attribute

The PROV-DM states type information is described using the prov:type PROV attribute and may occur multiple times for a given entity, activity, agent, or relation.

PROV-XML uses the element `prov:type` to represent the prov:type PROV attribute. This element can be used to represent both PROV and non-PROV type information. The following examples shows type information encoded using the `prov:type` element.

EXAMPLE 1: type information using prov:type PROV attribute

```
<prov:document
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:prov="http://www.w3.org/ns/prov#"
    xmlns:ex="http://example.com/ns/ex#"
    xmlns:tr="http://example.com/ns/tr#">

  <prov:entity prov:id="tr:WD-prov-dm-20111215">
    <prov:type xsi:type="xsd:QName">prov:Plan</prov:type>
    <prov:type xsi:type="xsd:QName">ex:Workflow</prov:type>
  </prov:entity>

</prov:document>
```

The `prov:type` element can be used in conjunction with schema-defined PROV types (see examples 2-5).

2.4.2 Extension Types

PROV-XML defines complexTypes to match the PROV defined type values. These types provide a more native XML representation of PROV types. The following example is considered equivalent to the previous example because the element `prov:plan` has type `prov:Plan`. All complexTypes representing a PROV type which is defined as a subclass of another PROV type are defined in PROV-XML as extensions of their parent PROV type's complexType. For example, `prov:Plan` is defined as an

extension of the complexType `prov:Entity` and may be referenced by either `prov:plan` or `prov:entity`.

↳ is it the right word?

EXAMPLE 2: type information using schema defined types

```
<prov:document
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:prov="http://www.w3.org/ns/prov#"
    xmlns:ex="http://example.com/ns/ex#"
    xmlns:tr="http://example.com/ns/tr#">

  <prov:plan prov:id="tr:WD-prov-dm-20111215">
    <prov:type xsi:type="xsd:QName">ex:Workflow</prov:type>
  </prov:plan>

</prov:document>
```

→ which? relation

When an extended type is used, a PROV type attribute relation may be inferred for the current and any parent type of the declared type.

Stating all type information using the PROV type attribute assists in interoperability with non-PROV-XML encoding of PROV.

EXAMPLE 3: type information using schema defined types - type inferences

```
<prov:document
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:prov="http://www.w3.org/ns/prov#"
    xmlns:ex="http://example.com/ns/ex#"
    xmlns:tr="http://example.com/ns/tr#">

  <prov:plan prov:id="tr:WD-prov-dm-20111215">
    <prov:type xsi:type="xsd:QName">ex:Workflow</prov:type>
    <prov:type>prov:Plan</prov:type> <!-- inferred -->
    <prov:type>prov:Entity</prov:type> <!-- inferred -->
  </prov:plan>

</prov:document>
```

This is confusing!

Is it prov:type xsd-element?

2.4.3 XSI Type

Because the `prov:Plan` complexType is defined as an extension of the complexType `prov:Entity`, the following example using `xsi:type` is valid and considered equivalent to the two previous examples. The attribute `xsi:type` tells an XML parser the complexType of the element. The value of `xsi:type` must be a complexType derived from the default element type in a schema with known location (referenced through `xsi:schemaLocation`).

EXAMPLE 4: type information using xsi:type

```
<prov:document
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:prov="http://www.w3.org/ns/prov#"
    xmlns:ex="http://example.com/ns/ex#"
    xmlns:tr="http://example.com/ns/tr#">

  <prov:entity prov:id="tr:WD-prov-dm-20111215" xsi:type="prov:Plan">
    <prov:type xsi:type="xsd:QName">ex:Workflow</prov:type>
```

```
    </prov:entity>

  </prov:document>
```

again, which relation do you refer to?

A PROV type attribute relation may be inferred by the use of the `xsi:type` XML attribute.

EXAMPLE 5: type information using xsi:type - type inferences

```
<prov:document
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:prov="http://www.w3.org/ns/prov#"
    xmlns:ex="http://example.com/ns/ex#"
    xmlns:tr="http://example.com/ns/tr#">

  <prov:entity prov:id="tr:WD-prov-dm-20111215" xsi:type="prov:Plan">
    <prov:type xsi:type="xsd:QName">ex:Workflow</prov:type>
    <prov:type>prov:Plan</prov:type> <!-- inferred -->
    <prov:type>prov:Entity</prov:type> <!-- inferred -->
  </prov:entity>

</prov:document>
```

2.5 Naming Conventions

XML - Element names are aligned with [PROV-N] record names (e.g. `prov:wasGeneratedBy`, `prov:actedOnBehalfOf`) and record parameter roles (e.g. `prov:delegate`, `prov:responsible` on a Delegation). Elements are named in camelCase which also conforms with [PROV-N] naming conventions.

ComplexType names are aligned with [PROV-DM] type names (e.g. `prov:Generation`, `prov:Delegation`). ComplexTypes are named in PascalCase which conforms to [PROV-DM] naming conventions and differentiates complexTypes from elements in the schema.

XML -

3. PROV XML Schema

refer to DM

Provenance concepts, expressed as PROV-DM types and relations, are organized according to six components that are defined in this section.

- **Component 1: entities and activities.** The first component consists of entities, activities, and concepts linking them, such as generation, usage, start, end. The first component is the only one comprising time-related concepts.
- **Component 2: derivations.** The second component is formed with derivations and derivation subtypes.
- **Component 3: agents, responsibility, and influence.** The third component consists of agents and concepts ascribing responsibility to agents.
- **Component 4: bundles.** The fourth component is concerned with bundles, a mechanism to support provenance of provenance.
- **Component 5: alternate.** The fifth component consists of relations linking entities referring to the same thing.
- **Component 6: collections.** The sixth component is about collections.

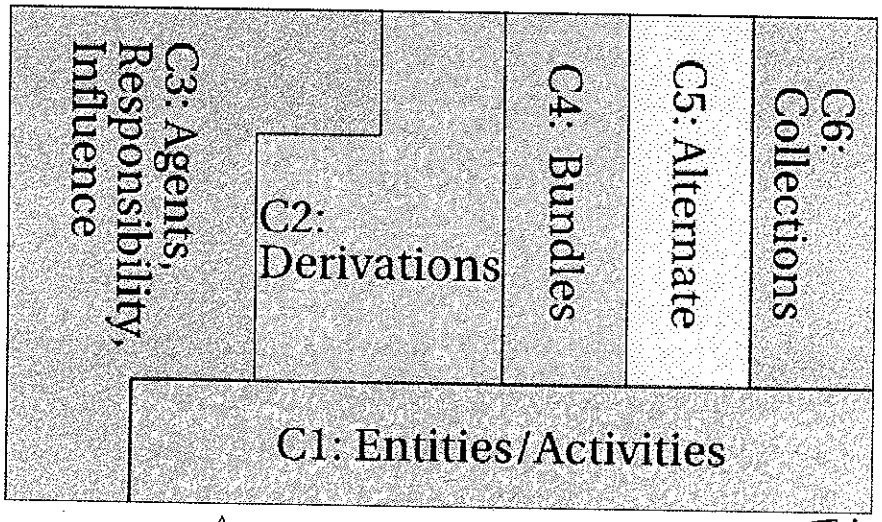

Figure 1◊: PROV-DM Components (Informative) Taken from DM

Table 1 is a mapping of PROV-DM types and relations ~~in~~ to? PROV-XML schema XML types and elements.

Table 1◊: PROV-DM Types and Relations to XML Schema Mapping

Type or Relation Name	XML Schema ComplexType	XML Schema Referencing Element	Component
Entity	prov:Entity	prov:entity	Component 1: Entities/Activities
Activity	prov:Activity	prov:activity	
Generation	prov:Generation	prov:wasGeneratedBy	
Usage	prov:Usage	prov:used	
Communication	prov:Communication	prov:wasInformedBy	
Start	prov:Start	prov:wasStartedBy	
End	prov:End	prov:wasEndedBy	
Invalidation	prov:Invalidation	prov:wasInvalidatedBy	
Derivation	prov:Derivation	prov:wasDerivedFrom	Component 2: Derivations
Revision	prov:Revision	prov:wasRevisionOf	
Quotation	prov:Quotation	prov:wasQuotedFrom	
Primary Source	prov:PrimarySource	prov:hadPrimarySource	
Agent	prov:Agent	prov:agent	Component 3: Agents, Responsibility, Influence
Attribution	prov:Attribution	prov:wasAttributedTo	
Association	prov:Association	prov:wasAssociatedWith	
Delegation	prov:Delegation	prov:actedOnBehalfOf	
Plan	prov:Plan	prov:plan	
Person	prov:Person	prov:person	
Organization	prov:Organization	prov:organization	
Software Agent	prov:SoftwareAgent	prov:softwareAgent	
Influence	prov:Influence	prov:wasInfluencedBy	
Bundle	prov:Bundle	prov:bundle	Component 4: Bundles
Bundle Constructor	prov:BundleConstructor	prov:bundleContent	

Can they be made linkable?

Alternate	prov:Alternate	prov:alternateOf	Component 5: Alternate
Specialization	prov:Specialization	prov:specializationOf	
Collection	prov:Collection	prov:collection	Component 6: Collections
Empty Collection	prov:EmptyCollection	prov:emptyCollection	
Membership	prov:Membership	prov:hadMember	

In the rest of the section, each type is defined, in English initially, followed by its XML schema definition and some example.

3.1 Component 1: Entities and Activities

The first component of PROV-DM is concerned with entities and activities, and their inter-relations: Usage, Generation, Start, End, and Communication. and Invalidation?

3.1.1 Entity

An entity is a physical, digital, conceptual, or other kind of thing with some fixed aspects; entities may be real or imaginary.

Type definition in XML Schema:

```
<xs:complexType name="Entity">
  <xs:sequence>
    <!-- prov attributes -->
    <xs:element ref="prov:label" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="prov:location" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="prov:type" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="prov:value" minOccurs="0"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute ref="prov:id"/>
</xs:complexType>
```

→ is it the right word? to denote?

The element `prov:entity` is used to reference a prov:Entity from within a prov:Document or prov:BundleConstructor.

→ why not typewriter

Element definition in XML Schema:

```
<xs:element xmlns:xs="http://www.w3.org/2001/XMLSchema" name="entity" type="prov:Entity"/>
```

→ can we get rid of this everywhere? This would make the schema more readable.

EXAMPLE 6: prov:entity

```
<prov:document
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:prov="http://www.w3.org/ns/prov#"
    xmlns:ex="http://example.com/ns/ex#"
    xmlns:tr="http://example.com/ns/tr#">

  <prov:entity prov:id="tr:WD-prov-dm-20111215">
    <prov:type xsi:type="xsd:QName">document</prov:type>
    <ex:version>2</ex:version>
  </prov:entity>

</prov:document>
```

Pedagogically, it's better to either
- define a default namespace
- add a meta

3.1.2 Activity

An activity is something that occurs over a period of time and acts upon or with entities; it may include consuming, processing, transforming, modifying, relocating, using, or generating entities.

Type definition in XML Schema:

```
<xs:complexType name="Activity">
  <xs:sequence>
    <xs:element name="startTime" type="xs:dateTime" minOccurs="0"/>
    <xs:element name="endTime" type="xs:dateTime" minOccurs="0"/>
    <!-- prov attributes -->
    <xs:element ref="prov:label" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="prov:location" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="prov:type" minOccurs="0" maxOccurs="unbounded"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute ref="prov:id"/>
</xs:complexType>
```

see above:

The element `prov:activity` is used to reference a prov:Activity from within a prov:Document or prov:BundleConstructor.

↳ why not in typecasting?

Element definition in XML Schema:

```
<xs:element xmlns:xs="http://www.w3.org/2001/XMLSchema" name="activity" type="prov:Activity"/>
```

EXAMPLE 7: prov:activity

```
<prov:document
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:prov="http://www.w3.org/ns/prov#"
    xmlns:ex="http://example.com/ns/ex#">

  <prov:activity prov:id="a1">
    <prov:startTime>2011-11-16T16:05:00</prov:startTime>
    <prov:endTime>2011-11-16T16:06:00</prov:endTime>
    <prov:type xsi:type="xsd:QName">ex:edit</prov:type>
    <ex:host>server.example.org</ex:host>
  </prov:activity>

</prov:document>
```

3.1.3 Generation

Generation is the completion of production of a new entity by an activity. This entity did not exist before generation and becomes available for usage after this generation.

Type definition in XML Schema:

```
<xs:complexType name="Generation">
  <xs:sequence>
    <xs:element name="entity" type="prov:IDRef"/>
    <xs:element name="activity" type="prov:IDRef" minOccurs="0"/>
    <xs:element name="time" type="xs:dateTime" minOccurs="0"/>
    <!-- prov attributes -->
    <xs:element ref="prov:label" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="prov:location" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="prov:role" minOccurs="0" maxOccurs="unbounded"/>
```

```
      <xs:element ref="prov:type" minOccurs="0" maxOccurs="unbounded"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute ref="prov:id"/>
  </xs:complexType>
```

to denote?

The element prov:wasGeneratedBy is used to reference a prov:Generation from within a prov:Document or prov:BundleConstructor.

Element definition in XML Schema:

```
<xs:element xmlns:xs="http://www.w3.org/2001/XMLSchema" name="wasGeneratedBy" type="prov:Genera
```

EXAMPLE 8: prov:wasGeneratedBy

```
<prov:document
    xmlns:prov="http://www.w3.org/ns/prov#"
    xmlns:ex="http://example.com/ns/ex#">

  <prov:entity prov:id="e1"/>

  <prov:activity prov:id="a1"/>

  <prov:wasGeneratedBy>
    <prov:entity prov:ref="e1"/>
    <prov:activity prov:ref="a1"/>
    <prov:time>2001-10-26T21:32:52</prov:time>
    <ex:port>p1</ex:port>
  </prov:wasGeneratedBy>

  <prov:entity prov:id="e2"/>

  <prov:wasGeneratedBy>
    <prov:entity prov:ref="e2"/>
    <prov:activity prov:ref="a1"/>
    <prov:time>2001-10-26T10:00:00</prov:time>
    <ex:port>p2</ex:port>
  </prov:wasGeneratedBy>

</prov:document>
```

3.1.4 Usage

Usage is the beginning of utilizing an entity by an activity. Before usage, the activity had not begun to utilize this entity and could not have been affected by the entity.

Type definition in XML Schema:

```
<xs:complexType name="Usage">
  <xs:sequence>
    <xs:element name="activity" type="prov:IDRef"/>
    <xs:element name="entity" type="prov:IDRef" minOccurs="0"/>
    <xs:element name="time" type="xs:dateTime" minOccurs="0"/>
    <!-- prov attributes -->
    <xs:element ref="prov:label" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="prov:location" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="prov:role" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="prov:type" minOccurs="0" maxOccurs="unbounded"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
<xs:attribute ref="prov:id"/>
</xs:complexType>
```

The element prov:used is used to reference a prov:Usage from within a prov:Document

or prov:BundleConstructor.

Element definition in XML Schema:

```
<xs:element xmlns:xs="http://www.w3.org/2001/XMLSchema" name="used" type="prov:Usage"/>
```

EXAMPLE 9: prov:used

```
<prov:document
    xmlns:prov="http://www.w3.org/ns/prov#"
    xmlns:ex="http://example.com/ns/ex#">

  <prov:activity prov:id="a1"/>

  <prov:entity prov:id="e1"/>

  <prov:entity prov:id="e2"/>

  <prov:used>
    <prov:activity prov:ref="a1"/>
    <prov:entity prov:ref="e1"/>
    <prov:time>2011-11-16T16:00:00</prov:time>
    <ex:parameter>p1</ex:parameter>
  </prov:used>

  <prov:used>
    <prov:activity prov:ref="a1"/>
    <prov:entity prov:ref="e2"/>
    <prov:time>2011-11-16T16:00:01</prov:time>
    <ex:parameter>p2</ex:parameter>
  </prov:used>

</prov:document>
```

3.1.5 Communication

Communication is the exchange of some unspecified entity by two activities, one activity using some entity generated by the other.

Type definition in XML Schema:

```
<xs:complexType name="Communication">
  <xs:sequence>
    <xs:element name="informed" type="prov:IDRef"/>
    <xs:element name="informant" type="prov:IDRef"/>
    <!-- prov attributes -->
    <xs:element ref="prov:label" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="prov:type" minOccurs="0" maxOccurs="unbounded"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute ref="prov:id"/>
</xs:complexType>
```

The element `prov:wasInformedBy` is used to reference a prov:Communication from within a prov:Document or prov:BundleConstructor.

Element definition in XML Schema:

```
<xs:element xmlns:xs="http://www.w3.org/2001/XMLSchema" name="wasInformedBy" type="prov:Communi
```

EXAMPLE 10: prov:wasInformedBy

```
<prov:document
```

```
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns:xsd="http://www.w3.org/2001/XMLSchema"
      xmlns:prov="http://www.w3.org/ns/prov#">

    <prov:activity prov:id="a1">
      <prov:type xsi:type="xsd:string">traffic regulations enforcing<prov:type>
    </prov:activity>

    <prov:activity prov:id="a2">
      <prov:type xsi:type="xsd:string">fine paying, check writing, and mailing</prov:type>
    </prov:activity>

    <prov:wasInformedBy>
      <prov:informed prov:ref="a2"/>
      <prov:informant prov:ref="a1"/>
    </prov:wasInformedBy>

  </prov:document>
```

3.1.6 Start

Start is when an activity is deemed to have been started by an entity, known as trigger. The activity did not exist before its start. Any usage, generation, or invalidation involving an activity follows the activity's start. A start may refer to a trigger entity that set off the activity, or to an activity, known as starter, that generated the trigger.

Type definition in XML Schema:

```
<xs:complexType name="Start">
  <xs:sequence>
    <xs:element name="activity" type="prov:IDRef"/>
    <xs:element name="trigger" type="prov:IDRef" minOccurs="0"/>
    <xs:element name="starter" type="prov:IDRef" minOccurs="0"/>
    <xs:element name="time" type="xs:dateTime" minOccurs="0"/>
    <!-- prov attributes -->
    <xs:element ref="prov:label" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="prov:location" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="prov:role" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="prov:type" minOccurs="0" maxOccurs="unbounded"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute ref="prov:id"/>
</xs:complexType>
```

The element `prov:wasStartedBy` is used to reference a prov:Start from within a prov:Document or prov:BundleConstructor.

Element definition in XML Schema:

```
<xs:element xmlns:xs="http://www.w3.org/2001/XMLSchema" name="wasStartedBy" type="prov:Start"/>
```

EXAMPLE 11: prov:wasStartedBy

```
<prov:document
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:prov="http://www.w3.org/ns/prov#"
    xmlns:ex="http://example.com/ns/ex#">

  <prov:entity prov:id="e1">
    <prov:type xsi:type="xsd:string">email message</prov:type>
  </prov:entity>
```

```
    <prov:activity prov:id="a1">
      <prov:type xsi:type="xsd:QName">Discuss</prov:type>
    </prov:activity>

    <prov:wasStartedBy>
      <prov:activity prov:ref="a1"/>
      <prov:trigger prov:ref="e1"/>
      <prov:time>2011-11-16T16:05:00</prov:time>
    </prov:wasStartedBy>

    <prov:used>
      <prov:activity prov:ref="a1"/>
      <prov:entity prov:ref="e1"/>
    </prov:used>

    <prov:activity prov:id="a0">
      <prov:type xsi:type="xsd:QName">Write</prov:type>
    </prov:activity>

    <prov:wasGeneratedBy>
      <prov:entity prov:ref="e1"/>
      <prov:activity prov:ref="a0"/>
    </prov:wasGeneratedBy>

    <prov:wasStartedBy>
      <prov:activity prov:ref="a1"/>
      <prov:trigger prov:ref="e1"/>
      <prov:starter prov:ref="a0"/>
      <prov:time>2011-11-16T16:05:00</prov:time>
    </prov:wasStartedBy>

    <prov:wasStartedBy>
      <prov:activity prov:ref="a1"/>
      <prov:starter prov:ref="a0"/>
      <prov:time>2011-11-16T16:05:00</prov:time>
    </prov:wasStartedBy>

  </prov:document>
```

3.1.7 End

End is when an activity is deemed to have been ended by an entity, known as trigger. The activity no longer exists after its end. Any usage, generation, or invalidation involving an activity precedes the activity's end. An end may refer to a trigger entity that terminated the activity, or to an activity, known as ender that generated the trigger.

Type definition in XML Schema:

```
<xs:complexType name="End">
  <xs:sequence>
    <xs:element name="activity" type="prov:IDRef"/>
    <xs:element name="trigger" type="prov:IDRef"  minOccurs="0"/>
    <xs:element name="ender" type="prov:IDRef" minOccurs="0"/>
    <xs:element name="time" type="xs:dateTime" minOccurs="0"/>
    <!-- prov attributes -->
    <xs:element ref="prov:label" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="prov:location" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="prov:role" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="prov:type" minOccurs="0" maxOccurs="unbounded"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute ref="prov:id"/>
</xs:complexType>
```

The element prov:wasEndedBy is used to reference a prov:End from within a prov:Document or prov:BundleConstructor.

Element definition in XML Schema:

```
<xs:element xmlns:xs="http://www.w3.org/2001/XMLSchema" name="wasEndedBy" type="prov:End"/>
```

EXAMPLE 12: prov:wasEndedBy

```
<prov:document
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:prov="http://www.w3.org/ns/prov#"
    xmlns:ex="http://example.com/ns/ex#">

  <prov:entity prov:id="e1">
    <prov:type xsi:type="xsd:string">approval document</prov:type>
  </prov:entity>

  <prov:activity prov:id="a1">
    <prov:type xsi:type="xsd:QName">Editing</prov:type>
  </prov:activity>

  <prov:wasEndedBy>
    <prov:activity prov:ref="a1"/>
    <prov:trigger prov:ref="e1"/>
  </prov:wasEndedBy>

</prov:document>
```

3.1.8 Invalidation

Invalidation is the start of the destruction, cessation, or expiry of an existing entity by an activity. The entity is no longer available for use (or further invalidation) after invalidation. Any generation or usage of an entity precedes its invalidation.

Type definition in XML Schema:

```
<xs:complexType name="Invalidation">
  <xs:sequence>
    <xs:element name="entity" type="prov:IDRef"/>
    <xs:element name="activity" type="prov:IDRef" minOccurs="0"/>
    <xs:element name="time" type="xs:dateTime" minOccurs="0"/>
    <!-- prov attributes -->
    <xs:element ref="prov:label" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="prov:location" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="prov:role" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="prov:type" minOccurs="0" maxOccurs="unbounded"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute ref="prov:id"/>
</xs:complexType>
```

The element prov:wasInvalidatedBy is used to reference a prov:Invalidation from within a prov:Document or prov:BundleConstructor.

Element definition in XML Schema:

```
<xs:element xmlns:xs="http://www.w3.org/2001/XMLSchema" name="wasInvalidatedBy" type="prov:Inva
```

EXAMPLE 13: prov:wasInvalidatedBy

```
<prov:document
```

```
      xmlns:prov="http://www.w3.org/ns/prov#"
      xmlns:ex="http://example.com/ns/ex#"
      xmlns:bbc="http://www.bbc.co.uk/news/">

    <prov:entity prov:id="ex:The-Painter"/>

    <prov:agent prov:id="ex:Picasso"/>

    <prov:wasAttributedTo>
      <prov:entity prov:ref="ex:The-Painter" />
      <prov:agent prov:ref="ex:Picasso" />
    </prov:wasAttributedTo>

    <prov:activity prov:id="ex:crash"/>

    <prov:wasInvalidatedBy>
      <prov:entity prov:ref="ex:The-Painter"/>
      <prov:activity prov:ref="ex:crash"/>
      <prov:time>1998-09-03T01:31:00</prov:time>
      <ex:circumstances>plane accident</ex:circumstances>
    </prov:wasInvalidatedBy>

  </prov:document>
```

3.2 Component 2: Derivations

The second component of PROV-DM is concerned with: derivations of entities from other entities and derivation subtypes WasRevisionOf (Revision), WasQuotedFrom (Quotation), and HasPrimarySource (Primary Source).

3.2.1 Derivation

A derivation is a transformation of an entity into another, an update of an entity resulting in a new one, or the construction of a new entity based on a pre-existing entity.

Type definition in XML Schema:

```
<xs:complexType name="Derivation">
  <xs:sequence>
    <xs:element name="generatedEntity" type="prov:IDRef"/>
    <xs:element name="usedEntity" type="prov:IDRef"/>
    <xs:element name="activity" type="prov:IDRef" minOccurs="0"/>
    <xs:element name="generation" type="prov:IDRef" minOccurs="0"/>
    <xs:element name="usage" type="prov:IDRef" minOccurs="0"/>
    <!-- prov attributes -->
    <xs:element ref="prov:label" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="prov:type" minOccurs="0" maxOccurs="unbounded"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute ref="prov:id"/>
</xs:complexType>
```

The element `prov:wasDerivedFrom` is used to reference a prov:Derivation from within a prov:Document or prov:BundleConstructor.

Element definition in XML Schema:

```
<xs:element xmlns:xs="http://www.w3.org/2001/XMLSchema" name="wasDerivedFrom" type="prov:Deriva
```

EXAMPLE 14: prov:wasDerivedFrom

```
<prov:document
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:prov="http://www.w3.org/ns/prov#">

  <prov:entity prov:id="e1"/>

  <prov:entity prov:id="e2"/>

  <prov:wasDerivedFrom>
    <prov:generatedEntity prov:ref="e2"/>
    <prov:usedEntity prov:ref="e1"/>
  </prov:wasDerivedFrom>

  <prov:wasDerivedFrom>
    <prov:generatedEntity prov:ref="e2"/>
    <prov:usedEntity prov:ref="e1"/>
    <prov:type xsi:type="xsd:string">physical transform</prov:type>
  </prov:wasDerivedFrom>

</prov:document>
```

3.2.2 Revision

A revision is a derivation for which the resulting entity is a revised version of some original.

Type definition in XML Schema:

```
<xs:complexType name="Revision">
  <xs:complexContent>
    <xs:extension base="prov:Derivation">
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

The element `prov:wasRevisionOf` is used to reference a prov:Revision from within a prov:Document or prov:BundleConstructor.

Element definition in XML Schema:

```
<xs:element xmlns:xs="http://www.w3.org/2001/XMLSchema" name="wasRevisionOf" type="prov:Revisic
```

EXAMPLE 15: prov:wasRevisionOf

```
<prov:document
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:prov="http://www.w3.org/ns/prov#"
    xmlns:rec54="http://www.w3.org/2001/02pd/rec54#"
    xmlns:tr="http://example.com/ns/tr#">

  <prov:entity prov:id="tr:WD-prov-dm-20111215">
    <prov:type xsi:type="xsd:QName">rec54:WD</prov:type>
  </prov:entity>

  <prov:entity prov:id="tr:WD-prov-dm-20111018">
    <prov:type xsi:type="xsd:QName">rec54:WD</prov:type>
  </prov:entity>

  <prov:wasRevisionOf>
    <prov:generatedEntity prov:ref="tr:WD-prov-dm-20111215"/>
    <prov:usedEntity prov:ref="tr:WD-prov-dm-20111018"/>
  </prov:wasRevisionOf>
```

```
</prov:document>
```

3.2.3 Quotation

A quotation is the repeat of (some or all of) an entity, such as text or image, by someone who may or may not be its original author.

Type definition in XML Schema:

```
<xs:complexType name="Quotation">
  <xs:complexContent>
    <xs:extension base="prov:Derivation">
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

The element `prov:wasQuotedFrom` is used to reference a prov:Quotation from within a prov:Document or prov:BundleConstructor.

Element definition in XML Schema:

```
<xs:element xmlns:xs="http://www.w3.org/2001/XMLSchema" name="wasQuotedFrom" type="prov:Quotati
```

EXAMPLE 16: prov:wasQuotedFrom

```
<prov:document
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:prov="http://www.w3.org/ns/prov#"
    xmlns:wp="http://thinklinks.wordpress.com/2012/03/07/"
    xmlns:ex="http://example.com/ns/ex#"
    xmlns:dm="http://dvcs.w3.org/hg/prov/raw-file/default/model/prov-dm.html#">

  <prov:entity prov:id="wp:thoughts-from-the-dagstuhl-principles-of-provenance-workshop"/>

  <prov:entity prov:id="dm:bl-dagstuhl"/>

  <prov:person prov:id="ex:Luc"/>

  <prov:person prov:id="ex:Paul"/>

  <prov:wasQuotedFrom>
    <prov:generatedEntity prov:ref="dm:gl-dagstuhl"/>
    <prov:usedEntity prov:ref="wp:thoughts-from-the-dagstuhl-principles-of-provenance-worksh
  </prov:wasQuotedFrom>

  <prov:wasAttributedTo>
    <prov:entity prov:ref="dm:bl-dagstuhl"/>
    <prov:agent prov:ref="ex:Luc"/>
  </prov:wasAttributedTo>

  <prov:wasAttributedTo>
    <prov:entity prov:ref="wp:thoughts-from-the-dagstuhl-principles-of-provenance-workshop"/:
    <prov:agent prov:ref="ex:Paul"/>
  </prov:wasAttributedTo>

</prov:document>
```

3.2.4 Primary Source

A primary source for a topic refers to something produced by some agent with direct