To the W3C PROV working group:

This document summarizes my comments on the PROV family of documents, which I reviewd in July 2012. I reviewed the Primer, Notation, and Data Model specs. I did not have time to complete a review of the Ontology, Constraints, or AQ documentation.

As a newcomer to the spec, I faced a steep learning curve. It took me quite some time to feel like I fully understood the model, caused in large part by my inability to distinguish between activities and relationships that sound like activities. Once I resolved those ambiguities I was able to understand the model more completely. These issues are documented below.

An accompanying document summarizes my proposed changes and additions to the PROV model, most of which are based on the comments below. Please let me know if you have any questions.

Regards,

Robert Freimuth, Ph.D.
Mayo Clinic

Feedback on the PROV family of documents

- Primer (http://www.w3.org/TR/prov-primer/)
  - Section 2
    - The figure is out of sync with the one in the Data Model document
  - Section 2.9
    - The use of the term "specialization" is potentially confusing as it implies that the child entity inherits all of the attributes of the parent entity. In document revision, this is not always the case (content can be deleted as well as added).
    - It is confusing to group versions of an entity (that contain different content) together with copies of an entity (that contain the same content). This makes the definition and interpretation of "alternates" ambiguous.
  - Section 3.9
    - The figure makes clear the ambiguous interpretation of "alternateOf". Both V1 and V2 are different "specializations" of "article", yet they are declared to be alternates. I find this unintuitive.
- Notation (http://www.w3.org/TR/prov-n/)
  - I would strongly prefer to see all attributes in the form of "name = value" pairs rather than relying on positional references. If the purpose of this syntax is to facilitate human readability ("aimed at human consumption", as stated in the doc), that will be achieved more easily when named attributes are used rather than expecting people to remember the order of attributes for each expression. For example:

    ```
    wasDerivedFrom( derivation = $d, drv_entity = $e2, src_entity
    = $e1, activity = $a, generation = $g2, usage = $u1, [
    optional_attributes] )
    ```

    This would also be consistent with how the optional attributes are specified.

- Section 2.1
  - The second example uses "a" in the first line and "a2" in the second line.
- Section 2.2
  - Please verify the definition of "(term|term)*". I believe it should be "matches zero or more occurrences of one of the two terms".
- Section 2.3
  - Is the use of "-" to indicate a missing term a standard convention? It seems unintuitive and potentially error-prone. NULL might be better if positional attributes are used (this issue is moot if named attributes are used).
  - There seems to be ambiguity in the syntax when the first parameter is option. For example, if both "wasDerivedFrom(e2, e1, a)" and "wasDerivedFrom(d, e2, e1)" are valid expressions, how can they be differentiated?
    - (e2, e1, a) is an acceptable form of (e2, e1, a, -, -)
    - (d, e2, e1) is an acceptable form of (d, e2, e1, -, -, -)
    - Without named attributes, it is not possible to unambiguously determine how to parse "wasDerivedFrom(1, 2, 3)"
- Section 2.4
  - The identifier syntax is confusing: wasDerivedFrom(e2, e1) is visually idential to wasDerivedFrom(d, e2), the latter being an invalid expression. If the first parameter of an expression is optional, it would be more clear to use a distinct delimiter between the identifier and the subsequent attributes (just as square brackets are used to deliniate optional attributes). That said, named attributes would be optimal.
  - The text for the second example refers to 3 activities, but the same identifier is used in each line of syntax.
  - The text for the second example is an overstatement. Since attributes can be specified separately, it would be more accurate to say an expression does not specify any attributes rather than saying that the activity has no attributes.
- Section 3
  - The EBNF forms of each expression are helpful, but they are not easily human-readable. Line breaks between attributes would help distinguish the groupings tremendously (this is especially important when trying to determine which terms are optional, as visually identifying matched pairs of parentheses within a long statement can be challenging). Compare:

```
generationExpression ::= 'wasGeneratedBy' '(' (( identifier |
'-' ) ',' ) ? eIdentifier ',' ( aIdentifier | '-' ) ',' ( time
| '-' ) optional-attribute-values ')'

generationExpression ::= 'wasGeneratedBy' '('
      (( identifier | '-' ) ',' ) ?
      eIdentifier ','
      ( aIdentifier | '-' ) ','
      ( time | '-' )
      optional-attribute-values ')'
```

  - It is difficult to visually differentiate between the various identifiers in the EBNF forms since the distinguishing characters are a minor part of the word. For example: (identifier | -) , aIdentifier , agIdentifier , (eIdentifier | -)… it

is very hard to distinguish between the terms, especially when "a" and "ag" are both used. Named attributes would help.
        - The EBNF form is the complete expression, but it lacks documentation to explain the components. To understand what each means, one needs to read the example text, map the text to the example expression, then map the example expression to the EBNF form. Starting with the EBNF form is even more challenging. It would help to have documentation within the EBNF sections.
    - Section 3.1.3
        - "Optional generation time" does not specify whether it is the start time or end time of the generation event.
    - Section 3.1.4
        - "Optional usage time" does not specify whether it is the start time or end time of the usage event.
    - Section 3.5
        - Inconsistent use of terms: key-value in this section compared to attribute-value elsewhere.
        - Why are keys quoted? This notation is for humans so the quotes are extra visual clutter.
    - Section 3.7
        - Recommend retitling the section to "Further Expressions and Reserved Words" to make it easier for people to find the latter.
    - Section 3.7.2
        - The identifier notation and accompanying definitions ("intended to denote…") would be helpful at the start of section 3, as they are used throughout the section without being explicitly defined.
    - Section 3.7.4
        - The encoding system should be explicitly defined (e.g., "#x22").
        - Is the encoding mandatory when provenance is expressed in languages other than HTML or XML (e.g., RDF)?
- Data Model (http://dvcs.w3.org/hg/prov/raw-file/default/model/prov-dm.html)
    - Section 2.1
        - The spec is missing the relation for "activity wasPartOf activity" to support composition. The existing relation "wasInformedBy" is a temporal relation that is distinct from structural composition. If support were added for composition, it would enable two sets of provenance to be integrated when they reference different levels of granularity.
    - Figure 1
        - Since the following sections are organized by "PROV concept" (e.g., Generation), it would help the reader if those terms were included in the relationship labels along with the names (e.g., wasGeneratedBy).
    - Table 2
        - All of the relations are verbs and therefore readers may consider them to be activities. In fact, the temptation to think of the concepts (e.g., generation, usage, communication) as activities is so strong that I was significantly confused by this for some time. It was only after several hours reading the

spec that I was able to start to think of them as stated relationships rather than as activities.

- For example, derivation seems to be a type of activity (e.g., In section 2.1.2, example 6 lists four examples, all of which use the verb "transformation". This term is also part of the definition of "activity" (section 2.1.1) so to those that are not intimately familiar with PROV, it seems quite clear that derivation is a type of activity.
- It may be helpful to explicitly state (and re-state) the distinction between assertions and activities, and/or select different terms when needed. For example, the actual activity that creates an entity is distinct from the expression of the relationship between them, but the term "generation" could be used to describe both concepts.

- Section 2.1.1
  - There seems to be significant semantic conflicts in the definitions of a few terms.
    - Activity: "something that occurs over a period of time … may include … using or generating entities"
    - Generation: "the completion of production"
    - Usage: "the beginning of utilizing"

    Based on the definition for Activity, usage and generation appear to be activities, but the separate definitions for those terms clearly indicate they are discrete points in time (at the beginning or end of an activity) and are not activities themselves. This is further confused by Figure 1 and Table 2, which list usage and generation as relations and not timepoints.
    It is not clear why it is necessary to define terms for discrete points in time within the PROV model. If activities already have start and end times, isn't that sufficient?
    - If these distinctions are important, I would recommend using different terms in the definitions and examples to break the circularity between them.
  - If activities were hierarchical (see comments for 2.1, wasPartOf), it would be possible to specify when a "parent" activity started to use an entity by defining the "usage" period as a separate child activity.
    - The creation (generation) and termination (invalidation) of entities can be handled similarly.
    - Communication can still be used in this example to state that the parent activity was informedBy the child activity.
  - The example of driving a car from Boston to Cambridge is not intuitive to most readers and does not highlight the strengths of the PROV model. The example focuses on the relocation of a physical object, which most people would consider a single entity (not two or more). The examples later in the spec use document editing, which is a straightforward example that will be very intuitive to readers. I suggest replacing the car example with a document example, which would also be more consistent with the rest of the spec.
    - This example should also be contained within an "example" box rather than as part of the text.

- Section 2.1.2
  - The concept of derivation seems to have as a central tenet the creation of a new entity. If this is true, I would recommend stating "the construction of a new entity based on a pre-existing entity" as the first phrase in the definition. The current wording of the definition places emphasis on transformation and updates, which should be qualifers of creation rather than the other way around.
  - It would help to clarify the authors' point of view regarding when entities are created. For example, there are several examples that state physical relocation results in the creation of a new entity. This will not be intuitive to many readers. The melting of ice in example 6 also falls into this category.
  - It could be argued that all entities derive from other entities, as no idea (or physical object) springs into existance without being influenced by pre-existing entities. Therefore, all entities are created through derivation.
- Section 2.2.1.2
  - "Agents may adopt plans". Since plans are entities and agents are related to entities through attribution, it follows that "wasAdoptedBy" is an expanded relation specified by PROV. This relation is missing from the spec.
- Section 2.2.2
  - A bundle (2.2.2) is a set of provenance descriptions, and a set is a type of collection (2.2.3). Therefore, it follows that a bundle is a specific type of collection (where the members of the collection are called "descriptions", as in section 5.4). This should be stated explicitly. Furthermore, since a bundle is a specialized collection, the spec would read better if their order in the document was reversed.
- Section 3
  - "PROV-N is a notation aimed at human consumption … The interpretation of PROV-N arguments is defined according to their position in the list of arguments. This convention allows for a compact notation."
    - IMO, the goal of human consumption trumps compact notation and using positional arguments hinders the former for the benefit of the latter. See also my comments on the PROV-N spec, especially regarding named attributes.
  - Example 15 shows a semicolon used to separate the optional identifier from the rest of the arguments. This is not consistent with the PROV-N spec.
- Section 4.1
  - Fig 2 is out of sync with the text. The figure has author and editor, while the text below uses contributor and editor.
- Tables 4 and 5
  - PROV-O gives the unqualified inverse of wasAssociatedWith as prov:wasAssociateFor, but that association isn't included in these tables. Please verify all docs in the PROV spec are internally consistent and complete, so that someone that reads only the data model spec is not missing information found in the ontology spec.
- Table 5

        - The bolded rows have some attributes listed in bold and some in normal font, presumably to indicate mandatory/optional status. This should be mentioned in the text.
        - The child relationships (e.g., revision) would be easier to see if their name were indented relative to their parent.
        - This table highlights the inconsistent attribute syntax. The combined use of positional attributes and attribute/value pairs (used for context-dependent optional attributes) is a little awkward.
    - Figure 5
        - The figure is a little cluttered. Since Time is a primitive datatype and not an object, it could be removed for clarity. Other datatypes are not shown.
        - The headers of the supporting text in section 5.1 are different than the labels in the figure, so the reader must use table 5 as a cross reference when going between them. Some translations are straightforward, but for readers that are new to PROV it is not obvious that wasInformedBy means Communication (for example).
        - When modeling complex relationships that have mandatory and optional attributes, class associations are a good approach to use. The current version of the spec does not fully model the relationships, however. Each of the gray classes in figure 5 could be modeled as a complete class that follows the requirements of the relationship.
            - For example, the class wasGeneratedBy currently shows only two attributes, although the spec for Generation lists five (four of which are optional, and one of which can be zero to many). All of this information can be represented very concisely in the diagram, which would make it easier for a reader to get a complete view of the spec graphically.
    - Section 5.1.3
        - The term "generation" implies an activity that takes place over time. In fact, the definition acknowledges this, as it specifies the point in time when generation is complete to distinguish use of the term to mean the entire generation activity. This is confusing.
            - Modeling this as an activity would allow users of PROV to represent the time it took to generate a new entity, and it might make the model more intuitive. It would also support users that want to know when entity creation began, not only at what point in time it was fully created.
    - Section 5.1.4
        - Like "generation", the term "usage" implies an activity that takes place over time. See comments for 5.1.3.
            - By defining this as a type of activity, users of PROV can specify the window of time when an entity was used.
    - Section 5.1.5 (Example 21)
        - Suggest changing the type within the statement for activity 2 to simply read "paying fine", as the supporting text enumerates the various activities that might be involved with that.
    - Section 5.1.6

        - Starting an activity does not always take place instantaneously. Like generation and usage, it might be advantageous to add support for "activity wasPartOf activity" so that one can model a startup process that takes time and may reference other entities or activities.
            - This would allow users to consider starting an activity as a separate activity, which has a defined start and end time and which could be ended (perhaps prematurely) by some triggering event.
- Section 5.1.7
    - See comments for 5.1.6.
    - The definition allows for an activity to trigger the end of another activity, but this isn't listed in table 5 or shown in figure 5.
- Section 5.1.8
    - See comments for 5.1.3.
    - Typo: "in the last three cases" should be "in the third case".
    - Invalidating an entity due to a state change is going to be difficult for some people to accept, even though it may not be strictly accurate from a rigorous philosophical point of view (not all adopters of PROV will use the model this way). In fact, if this definition is applied consistently throughout the spec, then all entities will have infinitesimally short lifespans and examples (such as the car relocation example in 2.1.1) will become extremely complex.
        - It might be worthwhile adding a section to discuss topics related to entity state, creation, etc. This would provide a way to retain these more complex points while simplifying the examples used throughout the rest of the spec.
    - Is it possible for entities to become temporarily unavailable (e.g., for usage)? If so, a state model for entities might be helpful.
- Section 5.2.1
    - See comments for 2.1.2, as well as the text that indicates that a derivation is an activity ("underpinning activities performing the necessary actions resulting in such a derivation"). However, it seems the intended concept of a derivation is a summary of information that describes how the creation of one entity was informed by another. If this is correct, is a derivation a type of bundle? Or would a bundle contain statement(s) regarding a derivation? Please clarify the relationship between these concepts.
- Sections 5.2.2 and 5.2.3
    - The semantic distinction between revision and quotation is not clear. To a computer, a document that contains content from another could have been created through either revision or quotation. If these types are to be part of the spec they should be defined more clearly as users arbitrarily choosing one type over another will hinder interoperability.
        - I might suggest that "revision" be used to describe the logical lineage of an entity, whereas "quotation" would pull content from an entity in a different lineage; however, this still isn't completely precise as it requires a common interpretation of how to define "lineage".
- Section 5.2.4
    - The definition of a "primary source" implies that it is an entity when in fact the term qualifies the role that a given entity plays during the creation of a

new entity, not the derivation itself. This might seem to be a minor point, but it is clearly different from both revision and quotation, both of which could be used when deriving a new entity from an entity used as a primary source.
        - It is also important to note that a given entity might be a primary source for one entity but not another ("primary source" is context-dependent).
- Figure 8
    - The figure incorrectly indicates that any child association can be used between and among the entity, activity, and agent classes. It would be more accurate to remove the labels from the solid associations and use the dashed lines to trace each relationship to its associated child association rather than to the parent class.
- Section 5.3.1
    - Given their definitions, Entities (or Activities) act as Agents for Activities. Since Person, Software, and Organization all fit the definition of Entity, I believe they should be specializations of Entity rather than Agent, which is a role that Entities can play in a given context.
- Section 5.3.3
    - By definition, agents can be both entities and activities (section 2.1.3). Can activities be responsible for other activities, or can only entities be assigned responsibility?
    - Similarly, can activities adopt a plan when acting as an agent, or can only entity agents adopt plans?
    - Can associations be valid for a window of time for a given activity? For example, several agents are responsible for the activity of preparing and serving food in a restaurant (server, cook, bartender), but not all of them are responsible for the entire time the activity is active.
        - To support this, start and end times should be added to Associations.
- Section 5.3.4
    - See comments for section 5.3.3. Can activities delegate to other agents, or can only entity agents delegate to other entities? (I assume the intent is to restrict delegation to only entity agents.)
    - Delegations are only valid for a given window of time, so start and end times should be added.
- Section 5.3.5
    - The notion of influence is useful for the PROV model, but it is unclear whether this is intended to represent an extension point for adopters of the spec. How should it be implemented?
    - If all relations inherit from the abstract Influence relationship, then all relations should inherit the attributes of Influence, including ID and attributes (both optional) and the ID of the influencee and influencer. The latter are not required in many relations, so there seems to be a semantic inconsistency within the model.
- Section 5.4.1
    - If bundles are collections (see comments for section 2.2.2), that should be reflected in Figure 9 (Entity – Collection – Bundle).
        - In addition, the members of a collection would be equivalent to the descriptions in a bundle. These terms should be harmonized.

        - If a bundle is an entity, it should require an ID and support a list of optional attributes. These should be added to the definition.
    - Section 5.5.1
        - Through the definition of Influence (figure 8), the relationship "specializationOf" should require an ID and support an optional list of attributes.
    - Section 5.5.2
        - It is not clear how to determine when two entities are considered alternates of each other, or when they are not. Please add more explanation, as this will be important for computational reasoning over provenance information.
        - Through the definition of Influence (figure 8), the relationship "alternateOf" should require an ID and support an optional list of attributes.
    - Section 5.5.3
        - I do not believe Mention is necessary, given the definitions of Entity, Collection, and Bundle. Specifically, bundles are collections (see comments for section 2.2.2), which have Entities as members. Therefore, membership in a bundle can be done as for collections (as described in section 5.6).
    - Section 5.6.1
        - Why is there a need to define EmptyCollection as a separate class?
        - There are no defined attributes for collections. Based on example 47 it appears only an ID is required. As a collection is an Entity, it should require an ID and support a list of optional attributes. It should also support a list of members (entity IDs).
    - Section 5.7.2 (Table 6)
        - The restrictions on when time, location and role can be used should be reviewed after the public feedback period closes and changes are made to the model. In particular, I think there is justification for allowing other relationships, such as Association and Delegation, to take these attributes. The model would be more flexible without these restrictions (which could be circumvented using user-defined optional attributes if needed, at the expense of maintaining a single standard representation for the information).
    - Section 5.7.2.2
        - It might be important to mention that while several objects are allowed to have a Location, it may not make sense to use it in some cases. For example, an activity that describes the relocation of an entity will have start and end locations, as well as every place in between those points.
    - Section 5.7.2.3
        - Role should be allowed for delegation so the relationship between the delegate and the responsible entity can be specified.
- Ontology ()
    - Not reviewed
- Constraints ()
    - Not reviewed
- AQ ()
    - Not reviewed