PROV: The W3C PROVenance model

Tutorial plan -- 1.5 hours

Part I: The PROV Data Model and PROV Notation (PROV-N)

Part II: Constraints of the Provenance Model

Part III: Known extensions and applications

Paolo Missier, Newcastle University
Khalid Belhajjame, University of Manchester
James Cheney, University of Edinburgh

EDBT 2013, March 20, 2013

PROV: The W3C PROVenance model

Part I of III
The PROV Data Model

Paolo Missier, Newcastle University
Khalid Belhajjame, University of Manchester
James Cheney, University of Edinburgh

Provenance Metadata

Provenance refers to the sources of information, including entities and processes, involving in producing or delivering an artifact (*)

Provenance is a description of how things came to be, and how they came to be in the state they are in today (*)

- Data dependencies -- from process or other data
- Activities that operate on data, and their temporal relationships
- Data integration: which fragments come from which source
- Who played what role in creating the data (responsibilities)
- ...

(*) Definitions proposed by the W3C Incubator Group on provenance:
http://www.w3.org/2005/Incubator/prov/wiki/What_Is_Provenance

Why does provenance matter?

- To establish quality, relevance, trust
- To track information attribution through complex transformations
- To describe one's experiment to others, for understanding / reuse
- To provide evidence in support of scientific claims
- To enable *post hoc* process analysis for debugging, improvement, evolution

...in such areas as:

- Open Information Systems
  - origin of the data, who was responsible for its creation
- Science applications
  - how the results were obtained
- News
  - origins and references of blogs, news items
- Law
  - licensing attribution of documents, data
  - privacy information

Provenance is not a new subject (*)

- There has been lot of work around
  - workflow systems
  - databases
  - knowledge representation
  - information retrieval
- Existing community-grown vocabularies
  - Open Provenance Model (OPM)
  - Dublin Core
  - Provenir ontology
  - Provenance vocabulary
  - SWAN provenance ontology
  - etc.

(*) source: Paul Groth, 2012 http://www.w3.org/2011/prov/wiki/ISWCProvTutorial

The W3C Working Group on Provenance: timeline

http://www.w3.org/2011/prov/wiki/

2009-2010

W3C
Incubator group
on provenance

Chair: Yolanda Gil,
ISI, USC

Main output:
"Provenance XG Final Report"
http://www.w3.org/2005/Incubator/prov/XGR-prov/
- provides an overview of the various existing approaches, vocabularies
- proposes the creation of a dedicated W3C Working Group

April, 2011

W3C
working group
approved

Chairs:
Luc Moreau,
Paul Groth

April, 2013

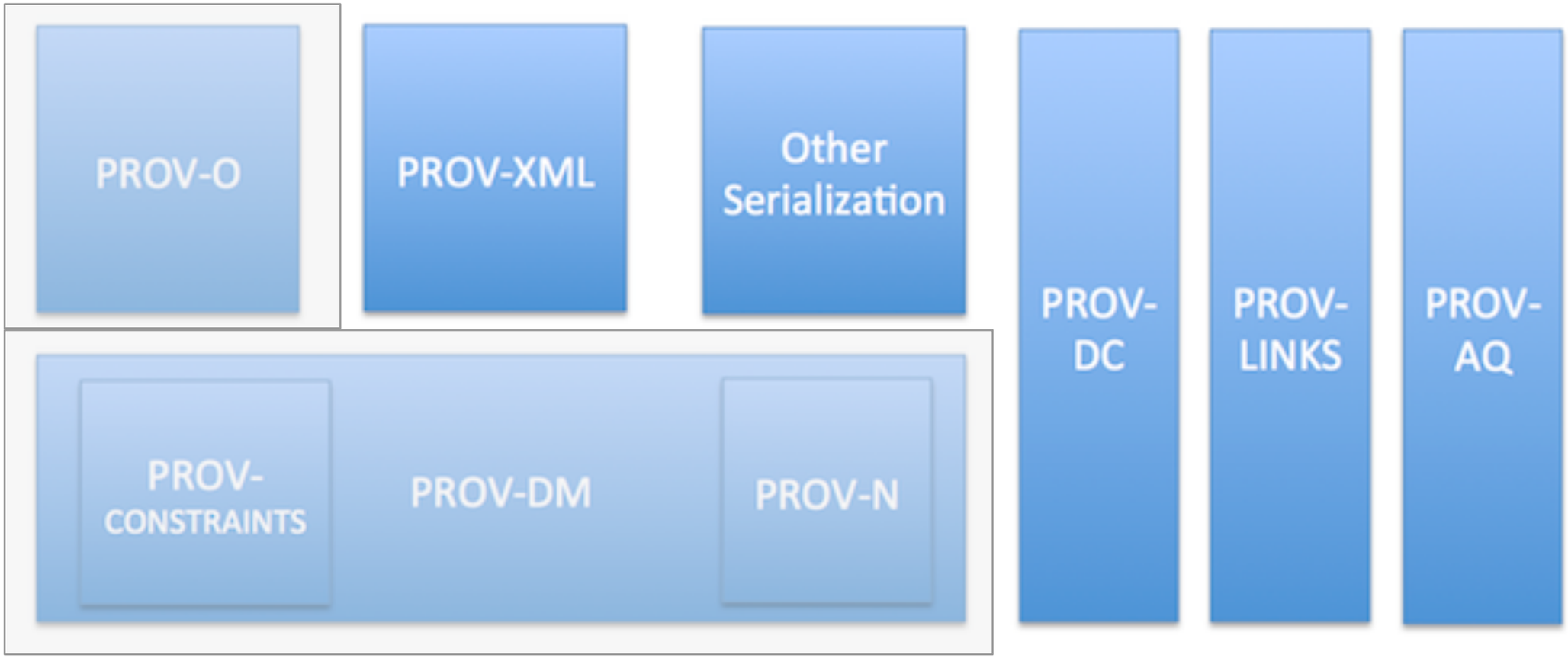
Proposed
Recommendations
finalised

prov-dm: Data Model
prov-o: OWL ontology, RDF encoding
prov-n: prov notation
prov-constraints

...plus a number of non-prescriptive Notes

PROV: scope and structure

Recommendation track

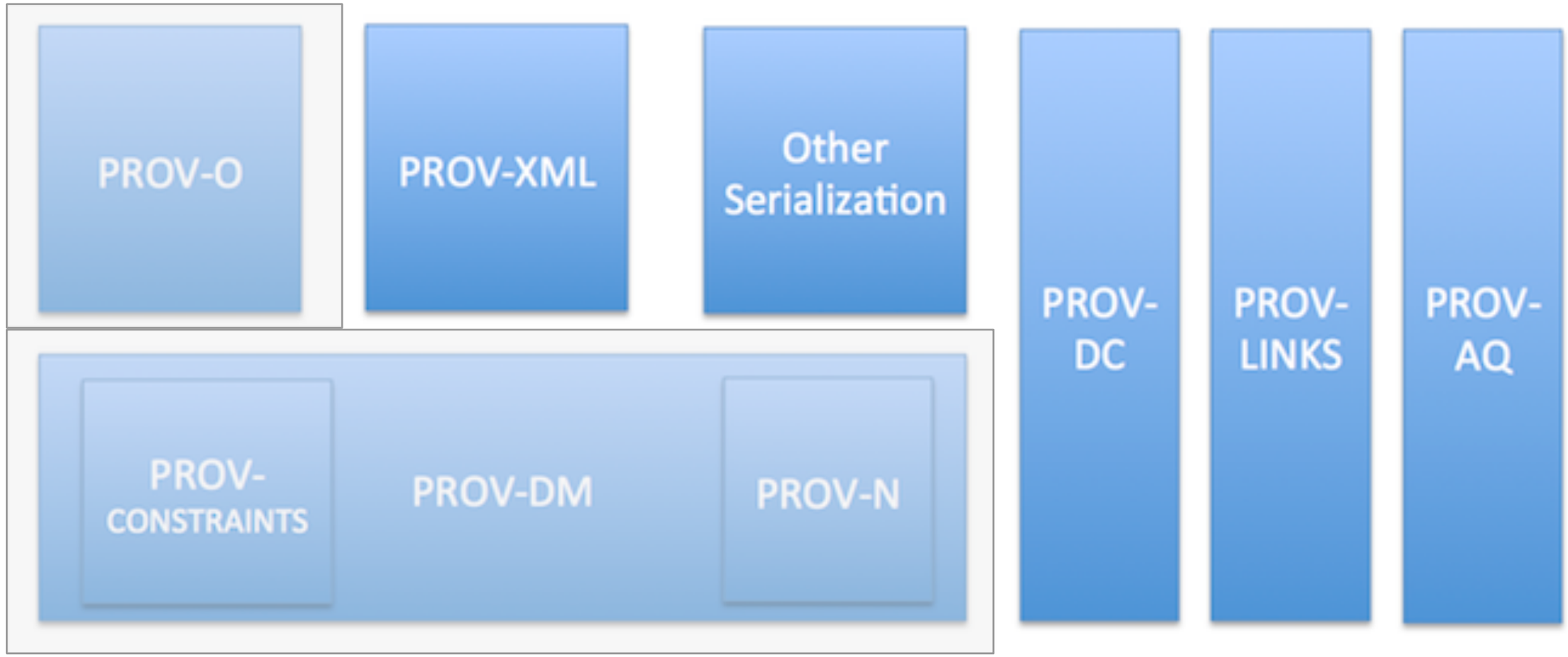


plus: Prov-dictionary

source: http://www.w3.org/TR/prov-overview/

PROV: scope and structure

Recommendation track



plus: Prov-dictionary

This tutorial

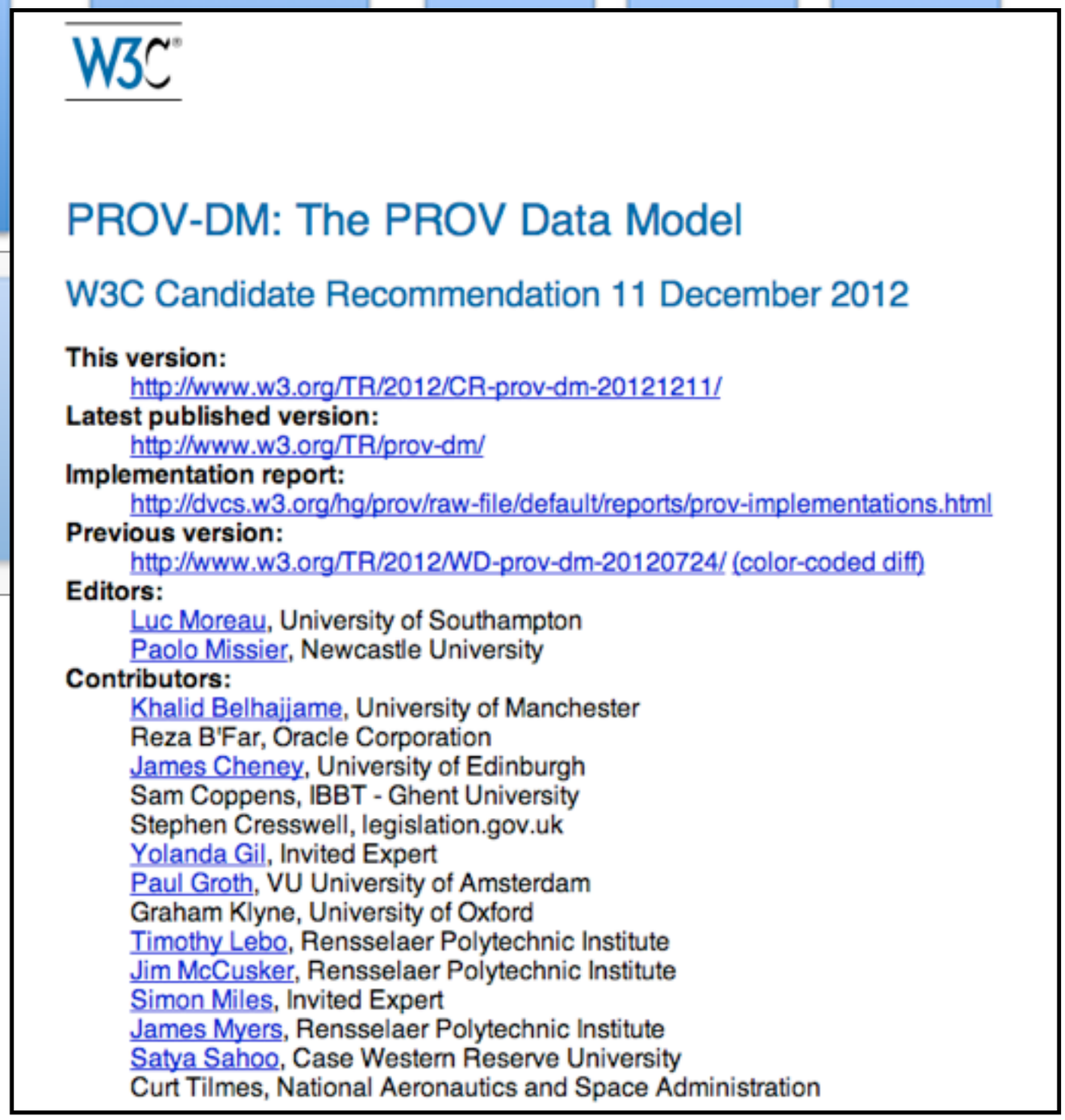
source: http://www.w3.org/TR/prov-overview/

PROV: scope and structure

Recommendation track

plus: Prov-dictionary

This tutorial



W3C

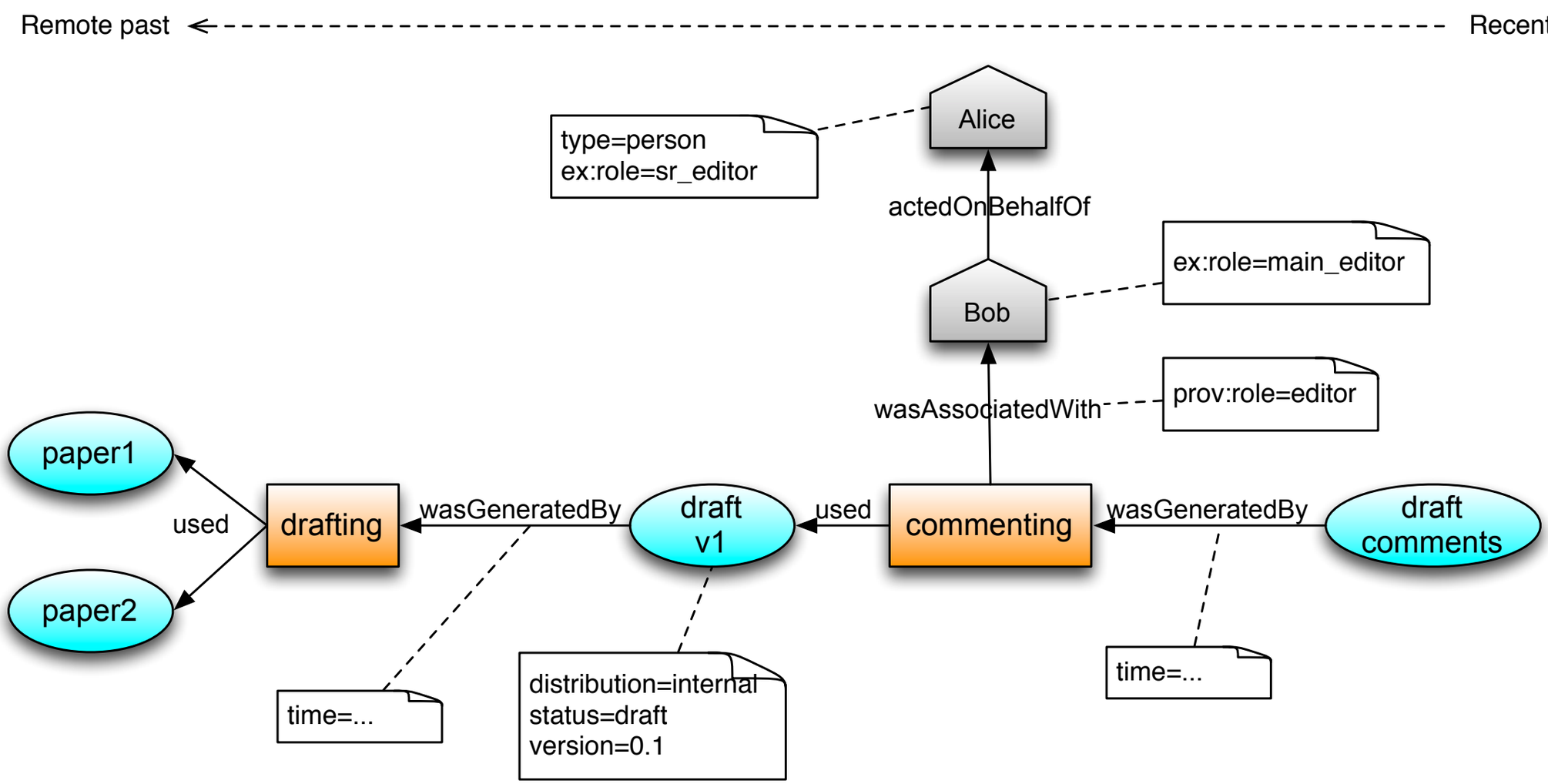
PROV-DM: The PROV Data Model

W3C Candidate Recommendation 11 December 2012

This version:
http://www.w3.org/TR/2012/CR-prov-dm-20121211/
Latest published version:
http://www.w3.org/TR/prov-dm/
Implementation report:
http://dvcs.w3.org/hg/prov/raw-file/default/reports/prov-implementations.html
Previous version:
http://www.w3.org/TR/2012/WD-prov-dm-20120724/ (color-coded diff)
Editors:
Luc Moreau, University of Southampton
Paolo Missier, Newcastle University
Contributors:
Khalid Belhajjame, University of Manchester
Reza B'Far, Oracle Corporation
James Cheney, University of Edinburgh
Sam Coppens, IBBT - Ghent University
Stephen Cresswell, legislation.gov.uk
Yolanda Gil, Invited Expert
Paul Groth, VU University of Amsterdam
Graham Klyne, University of Oxford
Timothy Lebo, Rensselaer Polytechnic Institute
Jim McCusker, Rensselaer Polytechnic Institute
Simon Miles, Invited Expert
James Myers, Rensselaer Polytechnic Institute
Satya Sahoo, Case Western Reserve University
Curt Tilmes, National Aeronautics and Space Administration

source: http://www.w3.org/TR/prov-overview/

PROV Core Elements (graph depiction)



Entity

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

Activity

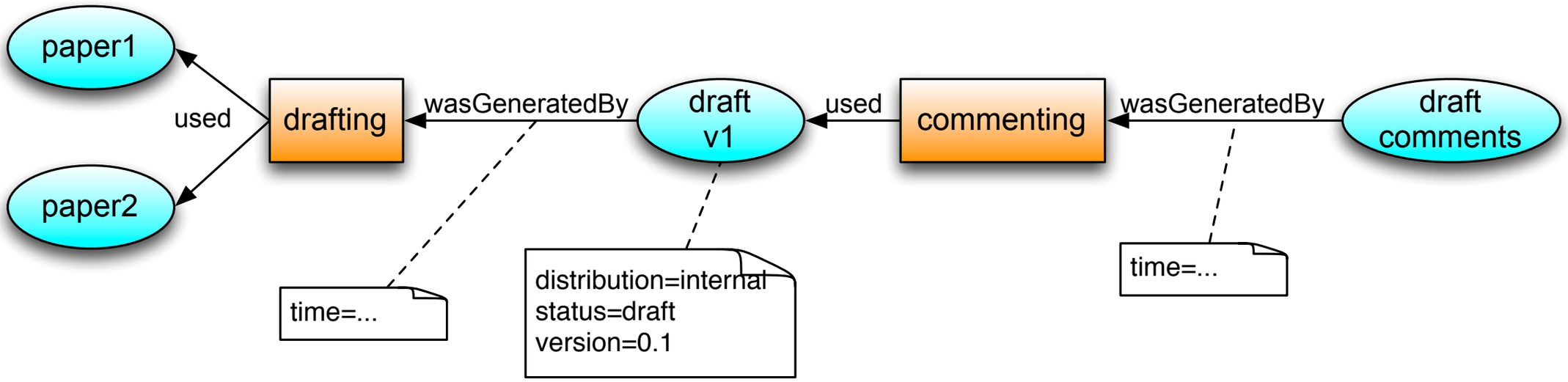
*An **activity** is something that occurs over a period of time and acts upon or with entities; it may include consuming, processing, transforming, ..., **using, or generating** entities.*

Agent

*An **agent** is something that bears some form of responsibility for an activity taking place, for the existence of an entity, or for another agent's activity.*

Jump to alternate

Generation, Usage



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.*

Usage *is the* ***beginning of utilizing*** *an entity by an activity. Before usage, the activity had not begun to utilize this entity*

PROV is based on a notion of ***instantaneous event***s, that mark transitions in the world

- generation, usage (and others)

Ordering constraints amongst events: **-- see part II of this tutorial**

“generation of e must precede each of usages”

“a can only use / generate e after it has started and before it has ended”

Concepts and relations

Generation of “draft v1” expressed as relation:

wasGeneratedBy(“draft v1”, ...)

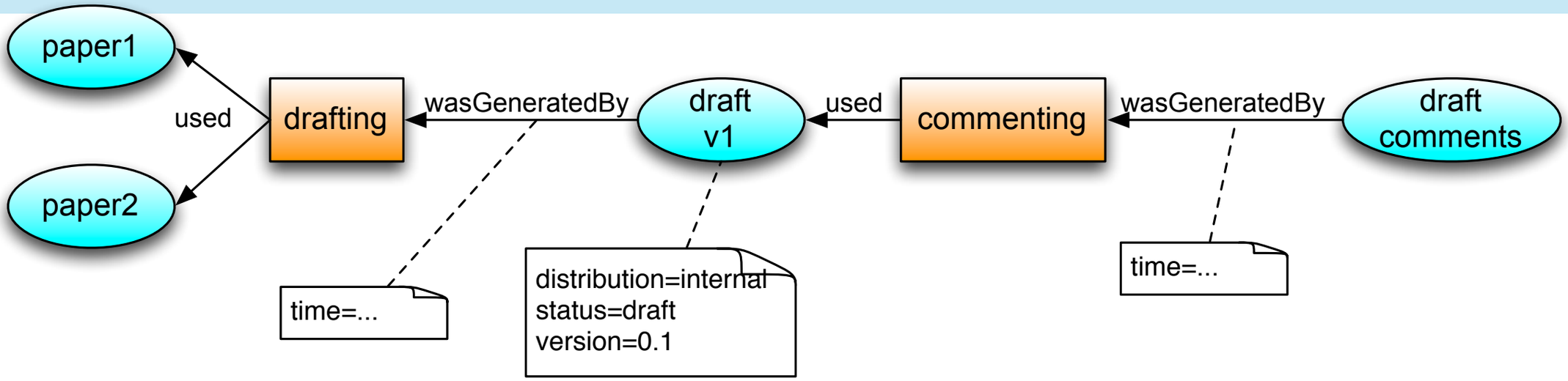
Usage of “draft v1” by “commenting” expressed as relation:

used(“commenting, “draft v1”,...)

Table 2◊: Mapping of PROV core concepts to types and relations

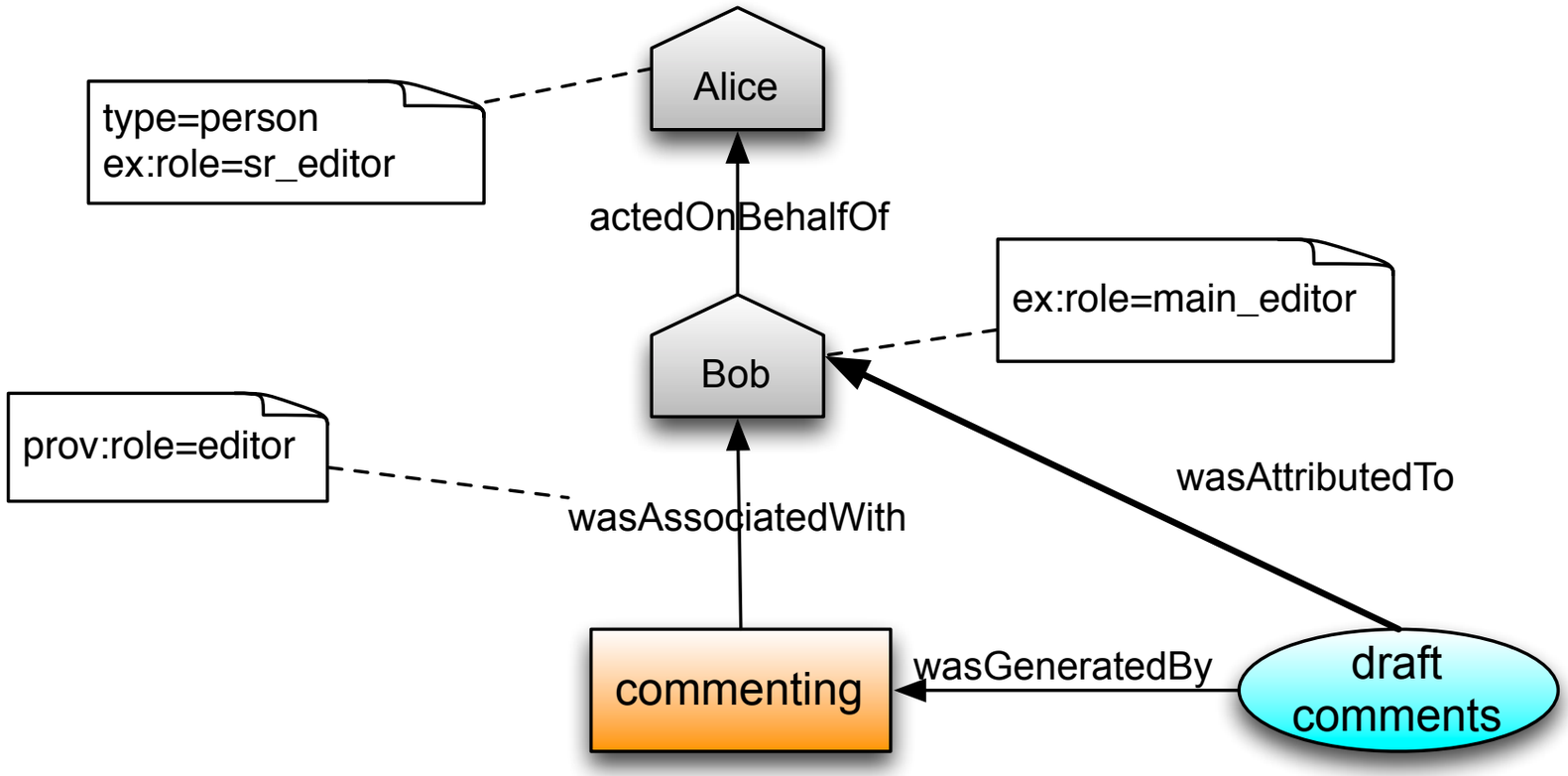
PROV Concepts	PROV-DM types or relations	Name	Overview
Entity	PROV-DM Types	Entity	Section 2.1.1
Activity		Activity	Section 2.1.1
Agent		Agent	Section 2.1.3
Generation	PROV-DM Relations	WasGeneratedBy	Section 2.1.1
Usage		Used	Section 2.1.1
Communication		WasInformedBy	Section 2.1.1
Derivation		WasDerivedFrom	Section 2.1.2
Attribution		WasAttributedTo	Section 2.1.3
Association		WasAssociatedWith	Section 2.1.3
Delegation		ActedOnBehalfOf	Section 2.1.3

PROV notation




```
document

prefix prov <http://www.w3.org/ns/prov#>
prefix ex <http://www.example.com/>

entity(ex:draftComments)
entity(ex:draftV1, [ ex:distr='internal', ex:status = "draft"])
entity(ex:paper1)
entity(ex:paper2)

activity(ex:commenting)
activity(ex:drafting)
wasGeneratedBy(ex:draftComments, ex:commenting, 2013-03-18T11:10:00)
used(ex:commenting, ex:draftV1, -)
wasGeneratedBy(ex:draftV1, ex:drafting, -)
used(ex:drafting, ex:paper1, -)
used(ex:drafting, ex:paper2, -)

endDocument
```

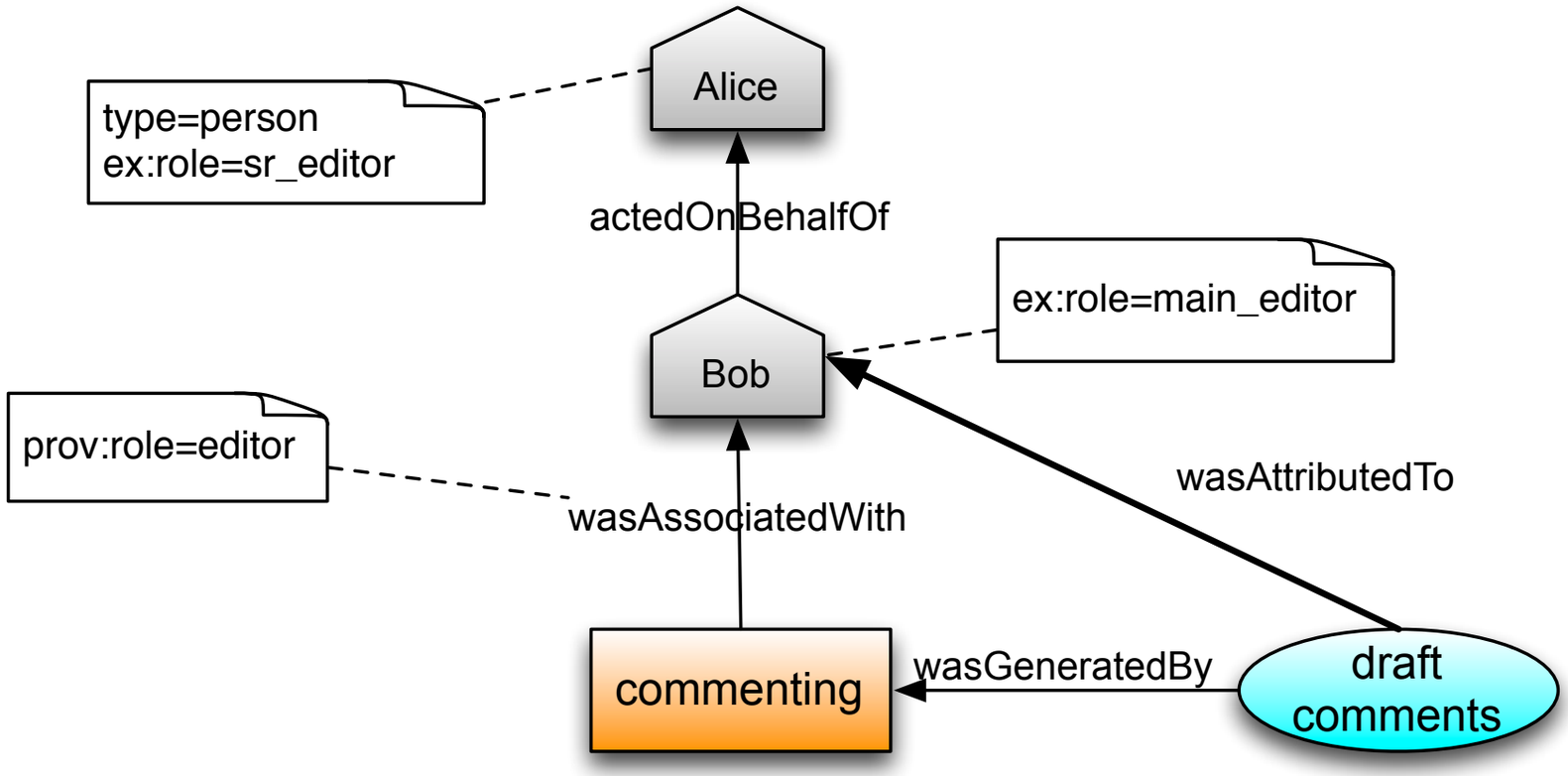
Association, Attribution, Delegation: who did what?



*An activity **association** is an assignment of responsibility to an agent for an activity, indicating that the agent had a role in the activity.*

***Attribution** is the ascribing of an entity to an agent.*

Association, Attribution, Delegation: who did what?



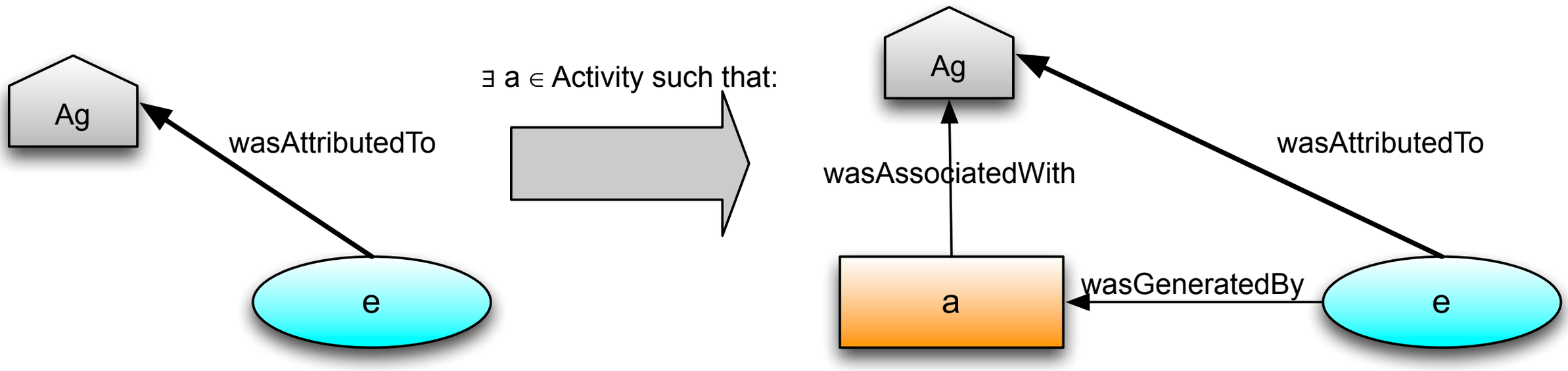
*An activity **association** is an assignment of responsibility to an agent for an activity, indicating that the agent had a role in the activity.*

***Attribution** is the ascribing of an entity to an agent.*

```
entity(ex:draftComments, [ ex:distr='internal' ])
activity(ex:commenting)
agent(ex:Bob,   [prov:type = "mainEditor"] )
agent(ex:Alice, [prov:type = "srEditor"])

wasAssociatedWith(ex:commenting, Bob, -, [prov:role = "editor"])
actedOnBehalfOf(Bob, Alice)
wasAttributedTo(ex:draftComments, ex:Bob)
```

Association and Attribution

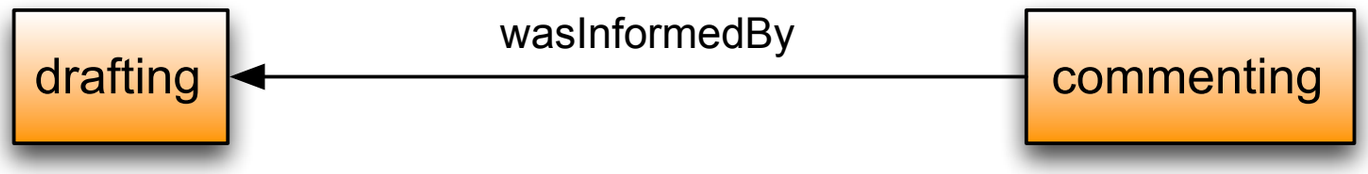
Q.: what is the relationship between attribution and association?



This is inference rule 13 in the PROV-CONSTR document
See part II of this tutorial

```
entity(e)
agent(Ag)
activity(a)

wasAttributedTo(e, Ag)
wasGeneratedBy(e, a)
wasAssociatedWith(a, Ag)
```

Communication amongst activities

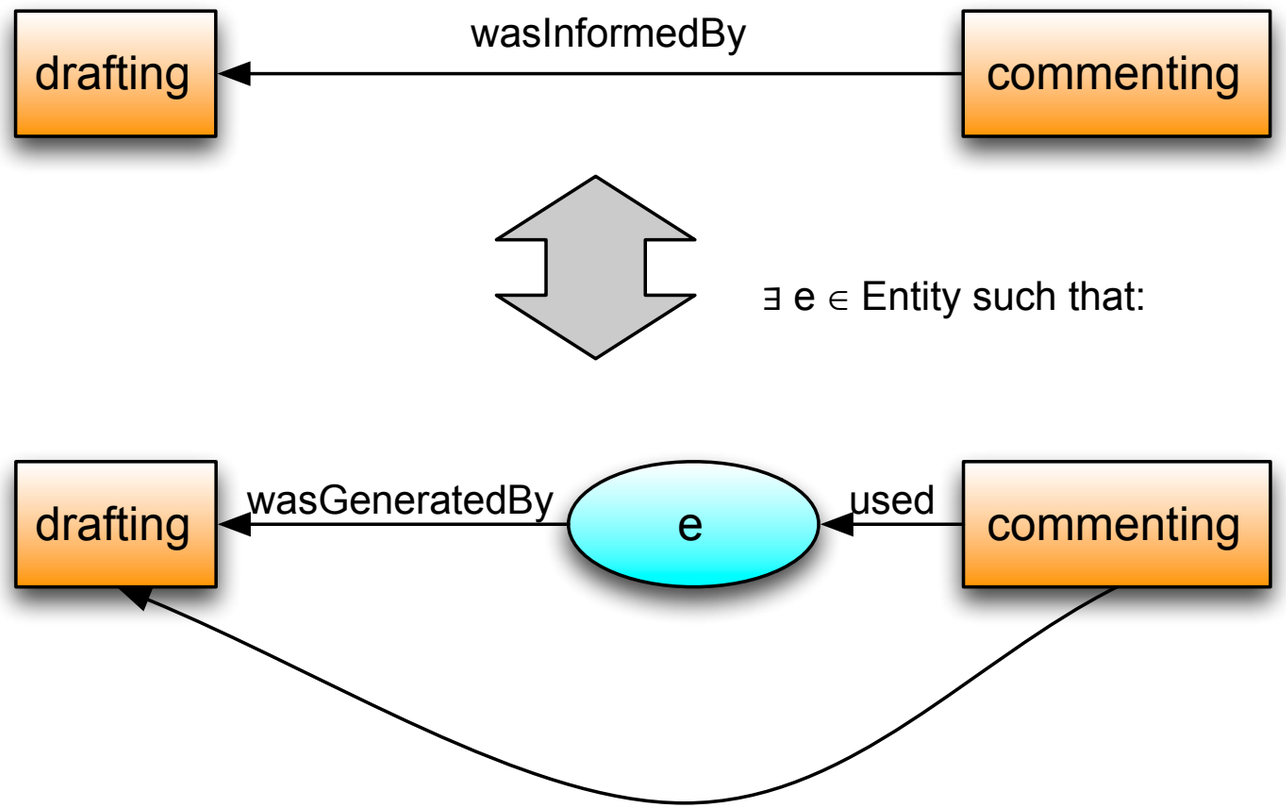


Communication *is the exchange of some unspecified entity by two activities, one activity using some entity generated by the other.*

```
activity(ex:commenting)
activity(ex:drafting)

wasInformedBy(ex:commenting, ex:drafting)
```

Communication, generation, usage

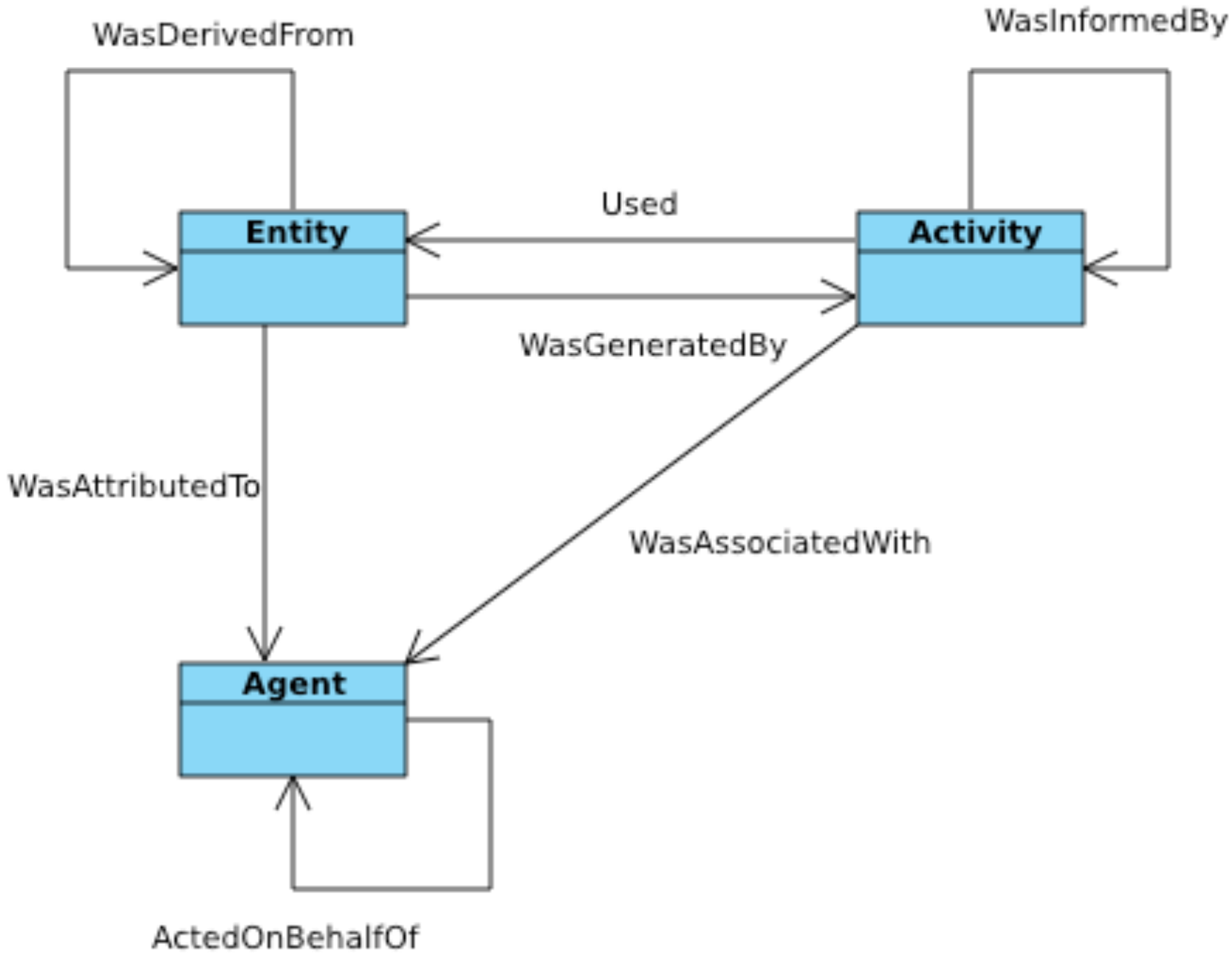
Q.: what is the relationship between communication, generation, and usage?




```
activity(ex:commenting)
activity(ex:drafting)
entity(e)
wasInformedBy(ex:commenting, ex:drafting)
wasGeneratedBy(e,ex:drafting)
used(ex:commenting, e)
```

This are inference rules 5 and 6 in the PROV-CONSTR document

Summary of the PROV Core model



Derivation amongst entities

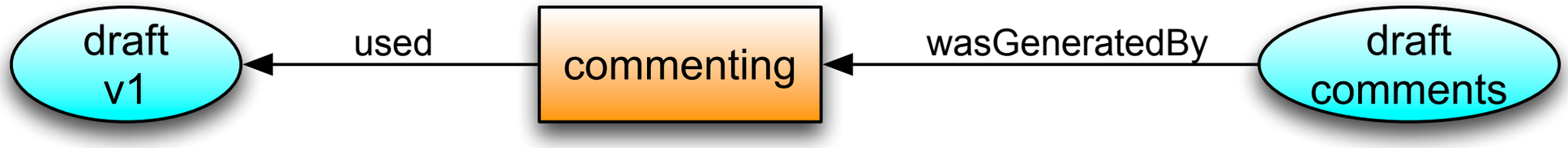
*A **derivation** is a transformation of an entity into another, an update of an entity resulting in a new one, or the construction of a new entity based on a pre-existing entity.*

```
entity(ex:draftV1)
entity(ex:draftV2)
wasDerivedFrom(ex:draftComments, ex:draftV1)
```

Q.: what is the relationship between derivation, generation, and usage?

Relations may be given identifiers

Relation IDs make it possible to refer to relations in other relations




```
entity(ex:draftComments)
entity(ex:draftV1)
activity(ex:commenting)
wasGeneratedBy(gen1; ex:draftComments, ex:commenting, -)
used(use1; ex:commenting, ex:draftV1, -)
```

gen1 denotes a generation event

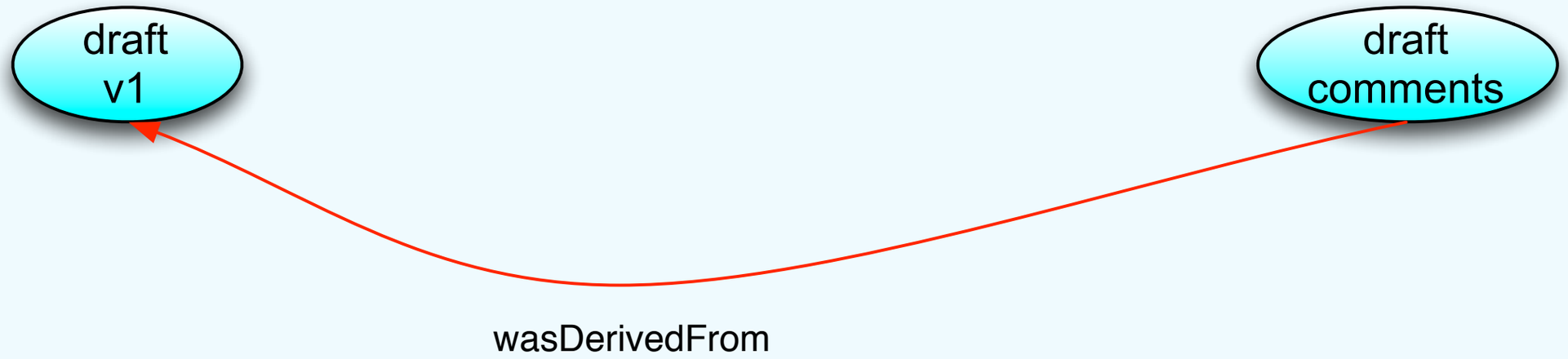
use1 denotes a usage event

General derivation relation:

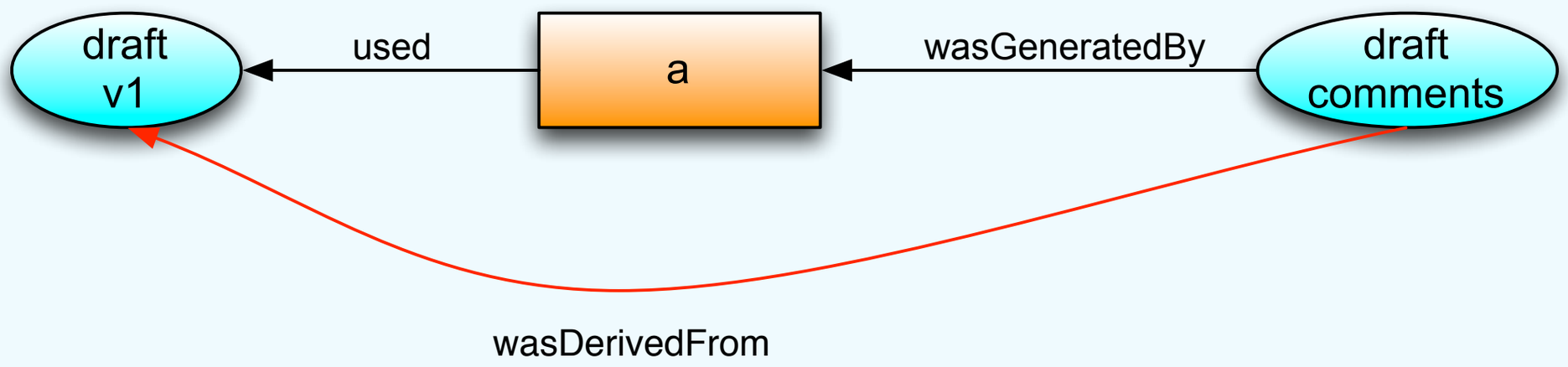
optional references to gen, usage events

```
wasDerivedFrom(id; e2, e1, a, g2, u1, attrs)
```

Derivation, Generation, Usage




```
wasDerivedFrom(draftV1, draftComments, _,_,_)
```




```
wasDerivedFrom(draftV1, draftComments, a, use1, gen1)
```

```
⇒ wasGeneratedBy(gen1; draftComments, a)
```

```
⇒ used(use1; a, draftV1)
```

Derivation, Generation, Usage

Given:

```
wasDerivedFrom(id; e2, e1, a, g2, u1, attrs)
```

where a, g2, u1 are not placeholders ‘-’, we can infer:

```
used(use1; a,e1,_t1,[])
```

and

```
wasGeneratedBy(gen2; e2,a,_t2,[]).
```

for some time stamps _t1, _t2.

If any of the optionals are placeholders, then no inference occurs, i.e.:

```
wasDerivedFrom(id; e2, e1, -, -, -, attrs)
```

PROV-DM relations at a glance

		Object				
		Entity		Activity		Agent
Subject	Entity	**WasDerivedFrom** Revision Quotation PrimarySource AlternateOf SpecializationOf HadMember		**WasGeneratedBy** WasInvalidatedBy	*R* *T* *L*	**WasAttributedTo**
	Activity	**Used** WasStartedBy WasEndedBy	*R* *T* *L*	**WasInformedBy**		**WasAssociatedWith** *R*
	Agent	—		—		**ActedOnBehalfOf**

PROV-DM relations at a glance

		Object				
		Entity		Activity		Agent
Subject	Entity	**WasDerivedFrom** Revision Quotation PrimarySource AlternateOf SpecializationOf HadMember		**WasGeneratedBy** WasInvalidatedBy	*R* *T* *L*	**WasAttributedTo**
	Activity	**Used** WasStartedBy WasEndedBy	*R* *T* *L*	**WasInformedBy**		**WasAssociatedWith** *R*
	Agent	—		—		**ActedOnBehalfOf**

Secondary optional elements in PROV-DM Relations

		Secondary Object		
		Entity	Activity	Agent
Subject	Entity	—	WasDerivedFrom (activity)	—
	Activity	WasAssociatedWith (plan)	WasStartedBy (starter) WasEndedBy (ender)	—
	Agent	—	ActedOnBehalfOf (activity)	—

Ternary relations - plans

Most relation types have two arguments which are { Entity, Activity, Agent}

Derivation is one exception:

```
wasDerivedFrom(id; e2, e1, a, g2, u1, attrs)
```

Two other notable exceptions:

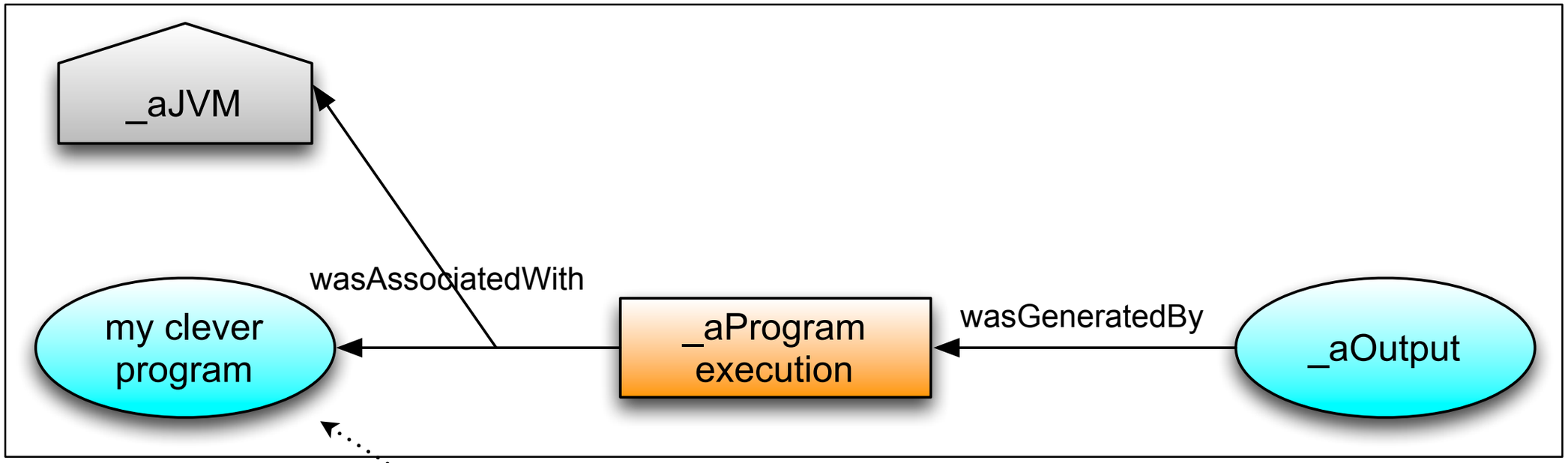
- Associations with a plan

- Delegation with an activity scope

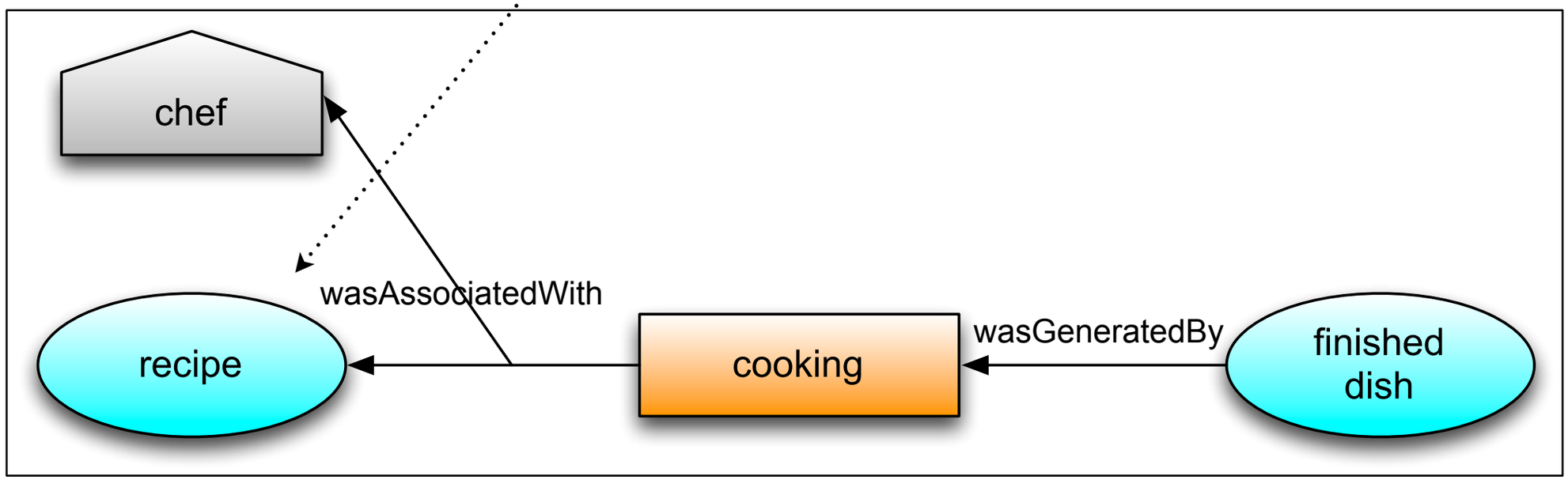
```
wasAssociatedWith(id; a, ag, pl, attrs)
```

A **plan** is an entity that represents a set of actions or steps intended by one or more agents to achieve some goal

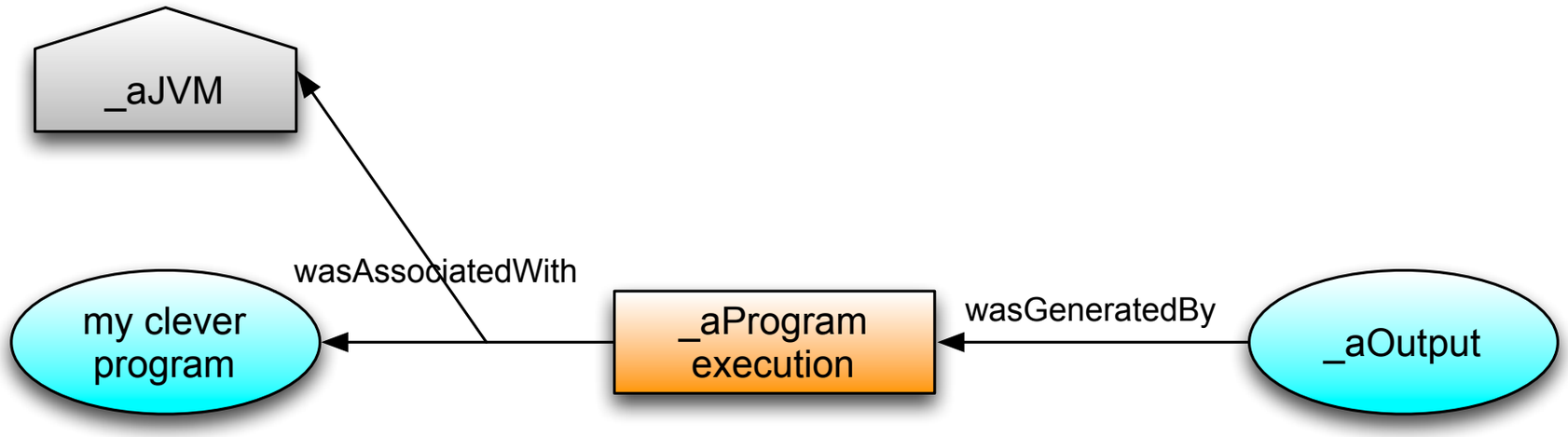
Association with a plan



A **plan** plays a role in an association



Plans are typed entities



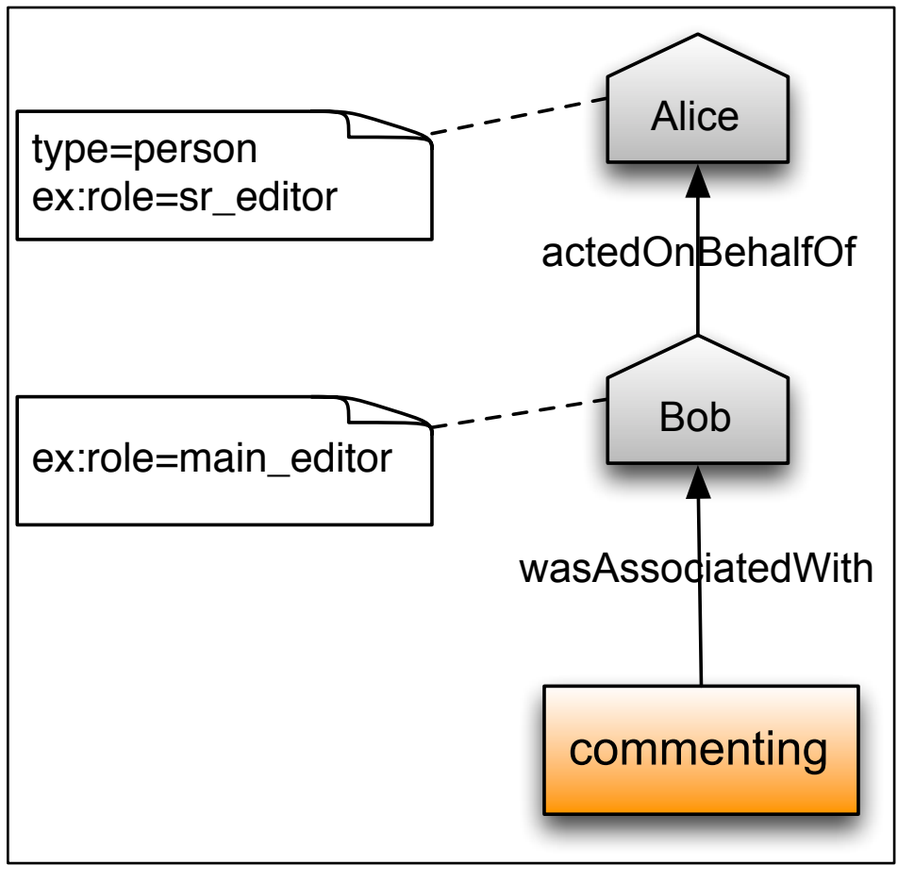
A plan is an entity having **prov:type “prov:plan”**

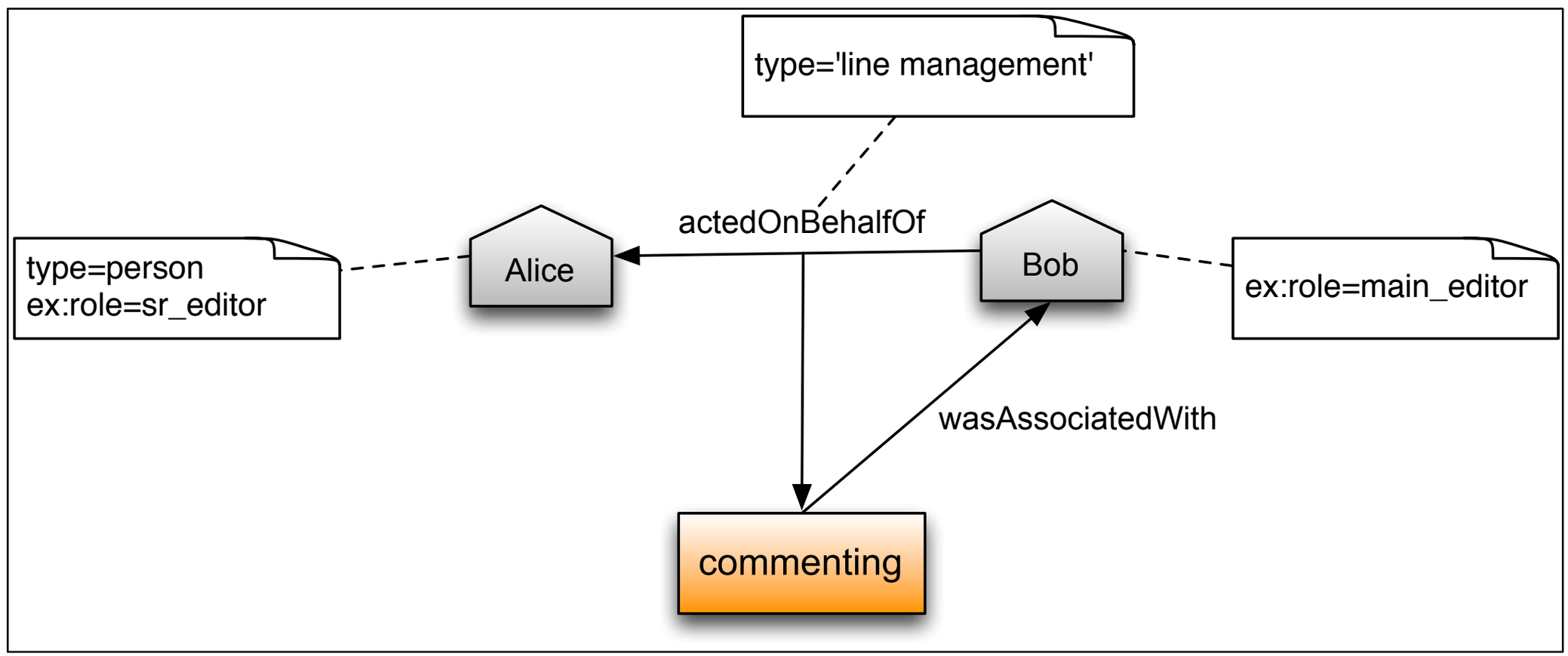
```
activity(ex:_aProgramExecution, [ex:execTime="22.5sec"])
agent(ex:_aJVM, [prov:type = “JVM-6.0”])
entity(ex:myCleverProgram,
        [prov:type='prov:Plan', ex:label="Program 1"])

wasAssociatedWith(ex:_aProgramExecution, ex:_aJVM,
                  ex:myCleverProgram,
                  [prov:role="defaultRuntime",
                   ex:accessPath="webapp" ])
```

Plans are optional.

Constraint 50.13: if a plan argument pl is specified, then pl must be of type Entity

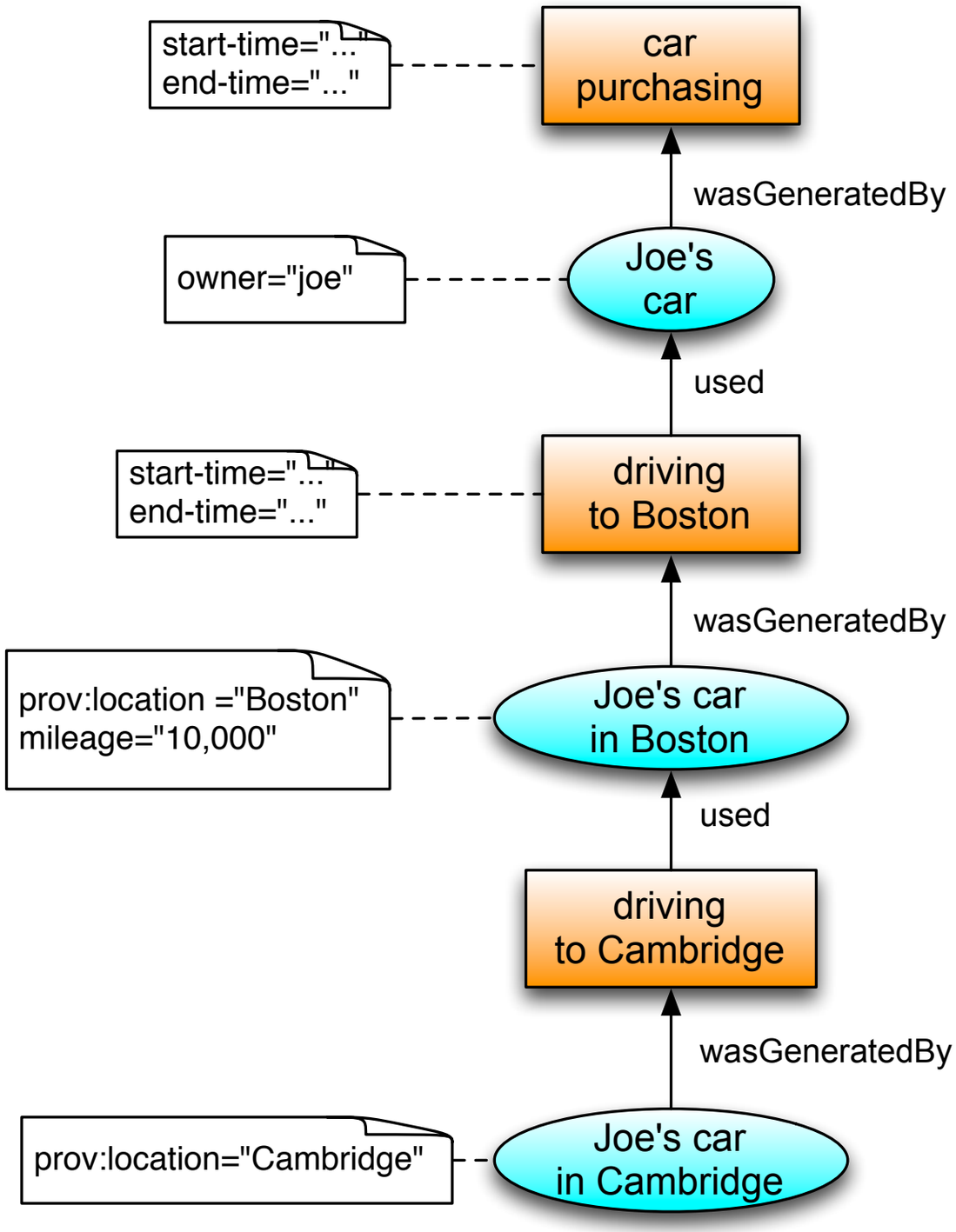
Delegation within an activity scope





Real-world artifacts vs provenance entities

"What do I know about the car I see in this Cambridge street today?"



- It was bought by Joe in 2011
- Joe drove it to Boston on March 16th, 2013. The car has now got 10,000 miles on it
- Joe drove it to Cambridge on March 18th, 2013.

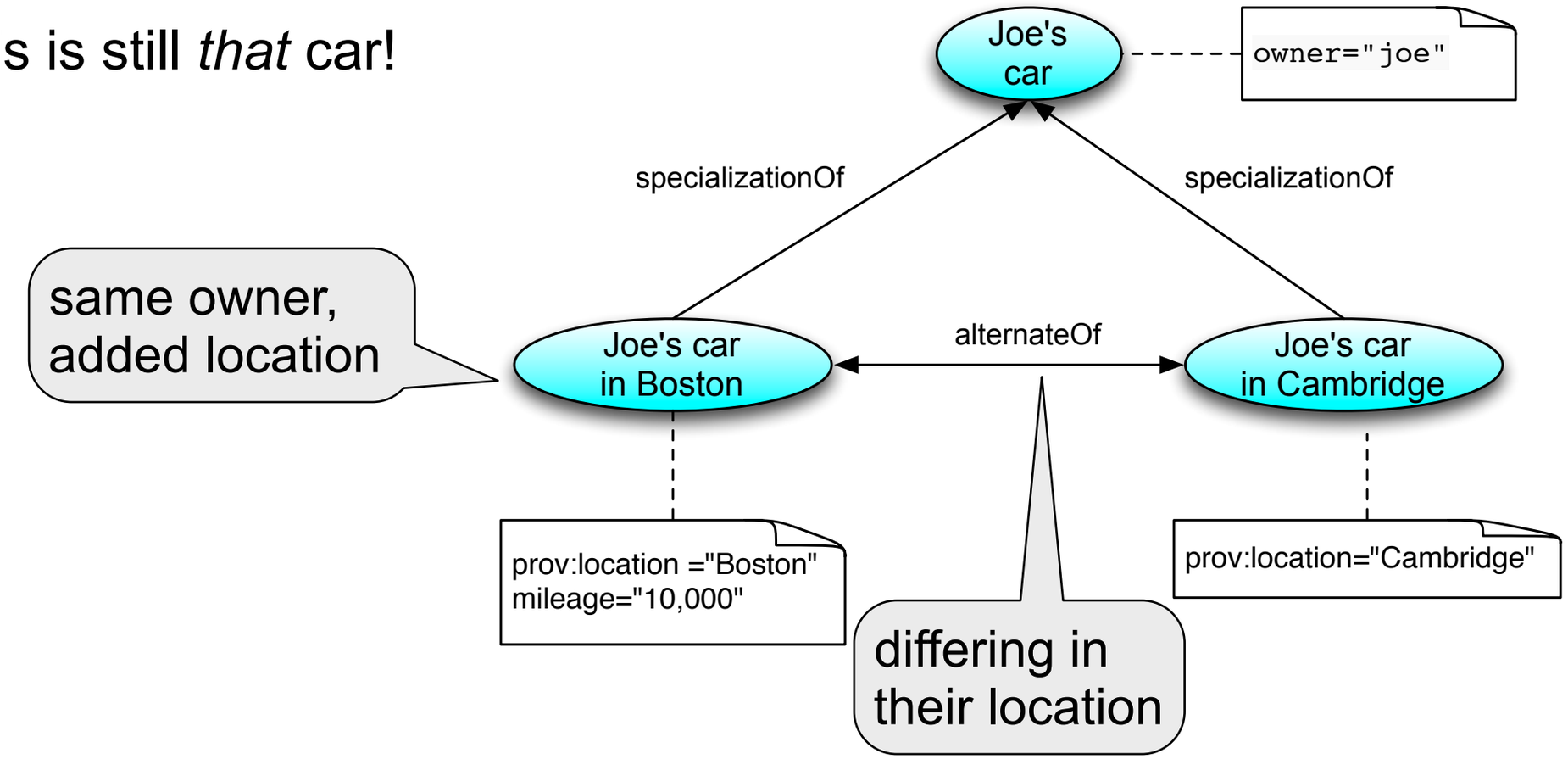
"Same" car, but different provenance at each stage of its evolution

ref: http://www.w3.org/2001/sw/wiki/PROV-FAQ#Examples_of_Provenance

To Core Elements

Alternate-specialization pattern

...But, this is still *that* car!



*An entity that is a **specialization** of another shares all aspects of the latter, and additionally presents more specific aspects of the same thing as the latter.*

*Two **alternate** entities present aspects of the same thing. These aspects may be the same or different, and the alternate entities may or may not overlap in time.*

Semantic notes:

1. Specialization implies alternate: IF specializationOf(e1,e2) THEN alternateOf(e1,e2).

2. Alternate is symmetric: IF alternateOf(e1,e2) THEN alternateOf(e2,e1)

3. Specialization is transitive: IF specializationOf(e1,e2) and specializationOf(e2,e3) THEN specializationOf(e1,e3).

To Core Elements

Reserved attributes and types

A small set of reserved attributes, with some usage restrictions

Attribute	Allowed In
prov:label	*any construct*
prov:location	Entity, Activity, Agent, Usage, Generation, Invalidation, Start, and End
prov:role	Usage, Generation, Invalidation, Association, Start, and End
prov:type	*any construct*
prov:value	Entity

Table 9 ◊: PROV-DM Predefined Types

Type	Specification	Core concept
`prov:Bundle`	Section 5.4.1	Entity
`prov:Collection`	Section 5.6.1	Entity
`prov:EmptyCollection`	Section 5.6.1	Entity
`prov:Organization`	Section 5.3.1	Agent
`prov:Person`	Section 5.3.1	Agent
`prov:Plan`	Section 5.1.1	Entity
`prov:PrimarySource`	Section 5.2.4	Derivation
`prov:Quotation`	Section 5.2.3	Derivation
`prov:Revision`	Section 5.2.2	Derivation
`prov:SoftwareAgent`	Section 5.3.1	Agent

Reserved attributes and types

A small set of reserved attributes, with some usage restrictions

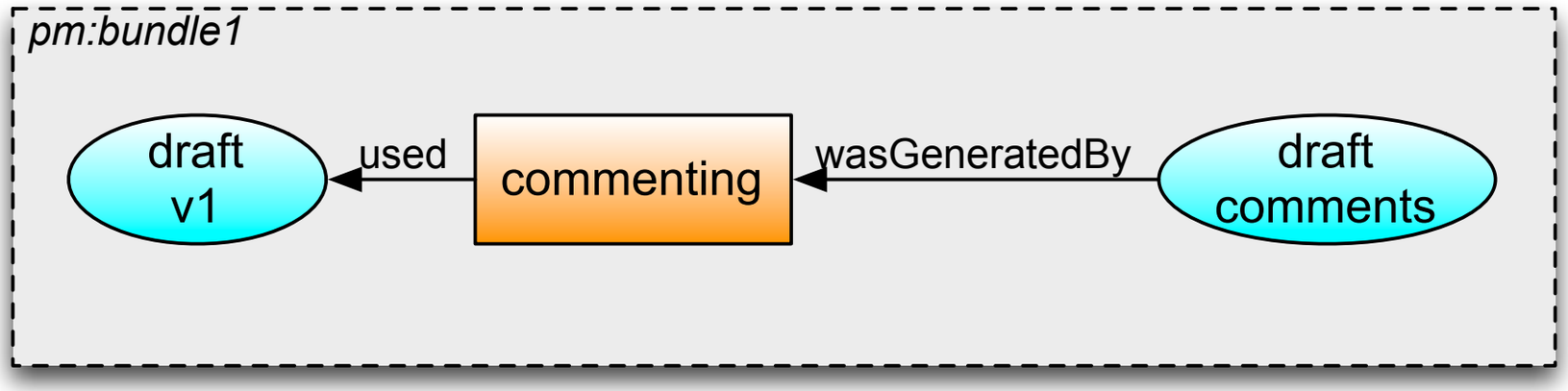
Attribute	Allowed In
prov:label	*any construct*
prov:location	Entity, Activity, Agent, Usage, Generation, Invalidation, Start, and End
prov:role	Usage, Generation, Invalidation, Association, Start, and End
prov:type	*any construct*
prov:value	Entity

Table 9 ◊: PROV-DM Predefined Types

Type	Specification	Core concept
`prov:Bundle`	Section 5.4.1	Entity
`prov:Collection`	Section 5.6.1	Entity
`prov:EmptyCollection`	Section 5.6.1	Entity
`prov:Organization`	Section 5.3.1	Agent
`prov:Person`	Section 5.3.1	Agent
`prov:Plan`	Section 5.1.1	Entity
`prov:PrimarySource`	Section 5.2.4	Derivation
`prov:Quotation`	Section 5.2.3	Derivation
`prov:Revision`	Section 5.2.2	Derivation
`prov:SoftwareAgent`	Section 5.3.1	Agent

Bundles, provenance of provenance

*A **bundle** is a named set of provenance descriptions, and is itself an entity, so allowing provenance of provenance to be expressed.*




```
bundle pm:bundle1

entity(ex:draftComments)
entity(ex:draftV1)

activity(ex:commenting)
wasGeneratedBy(ex:draftComments, ex:commenting,-)
used(ex:commenting, ex:draftV1, -)
endBundle
...
entity(pm:bundle1, [ prov:type='prov:Bundle' ])
wasGeneratedBy(pm:bundle1, -, 2013-03-20T10:30:00)
wasAttributedTo(pm:bundle1, ex:Bob)
```

Time, Events

- Provenance statements are combined by different systems

- An application may not be able to align the times involved to a single global timeline

Therefore, PROV minimizes assumptions about time

> Instead, the PROV data model is implicitly based on a notion of ***instantaneous event***s, that mark transitions in the world (*)

Events:

- activity start, activity end,

- entity generation , entity usage, entity invalidation

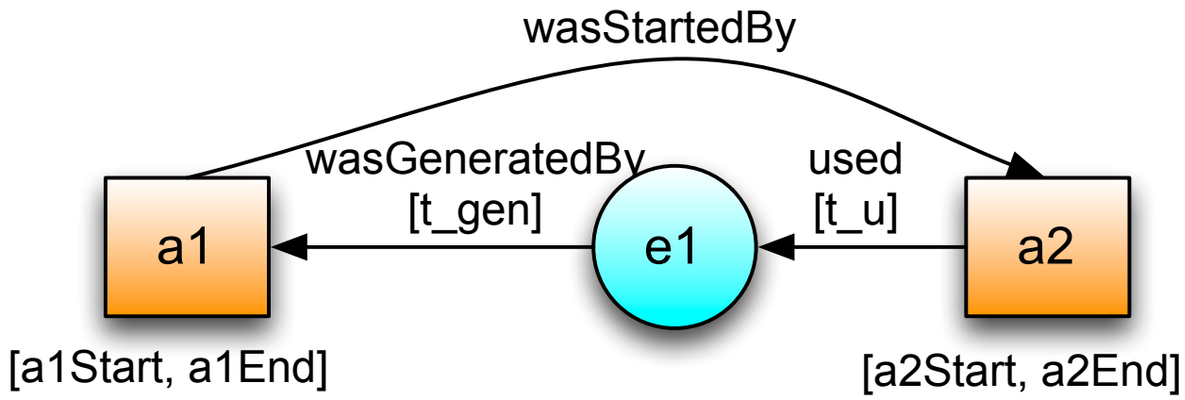
```
wasStartedBy(id; a2, e, a1, t, attrs)
```

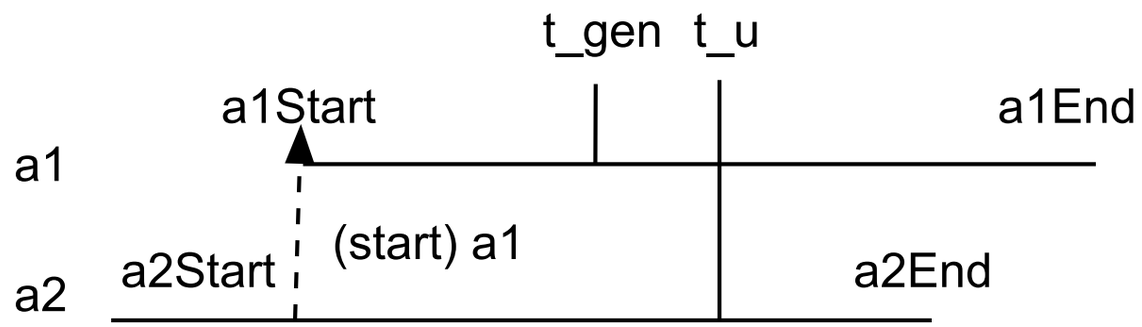
```
wasEndedBy(id; a2, e, a1, t, attrs)
```


(*) PROV-CONSTR http://www.w3.org/TR/prov-constraints/#events (non-normative)

From “scruffy” provenance to “valid” provenance

- Are all possible temporal partial ordering of events equally acceptable?
- How can we specify the set of all valid orderings?





More generally, how do we formally define what it means for a set of provenance statements to be *valid*?

On to part II...