W3C PROV Constraints

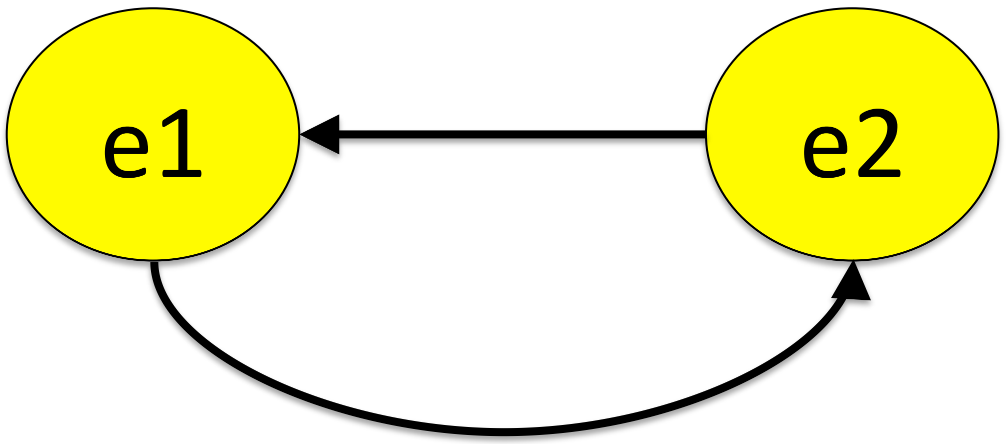
ISWC 2013

Paul Groth

slide help from Ivan Herman

Checking provenance statements ("Constraints")

- Provenance statements can become fairly complicated ☹
- In some applications it may become advantageous to *check* the validity of the provenance structures.



Definition of the constraints

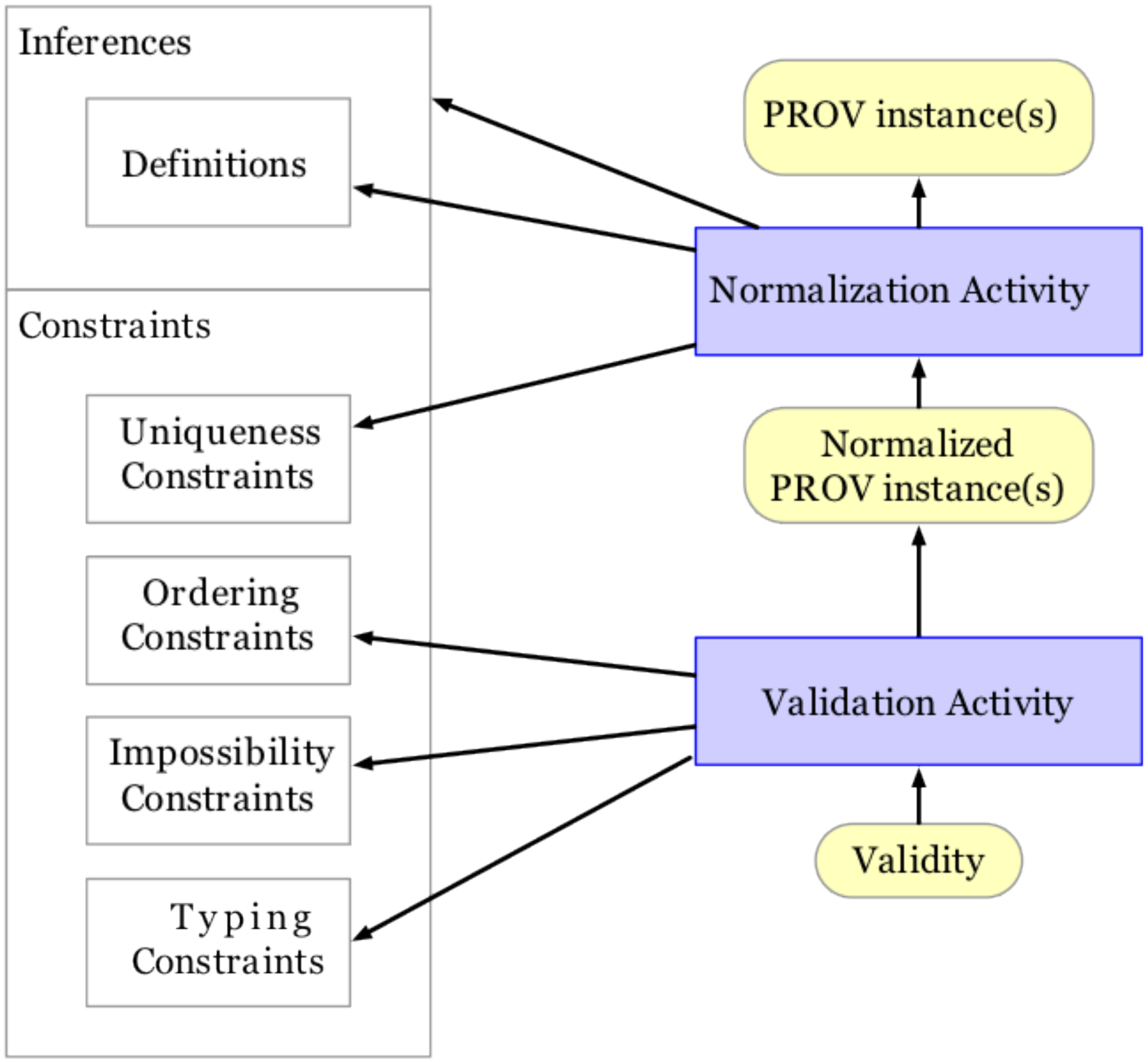
- An abstract data model for provenance (with its own, abstract notation) is also published
- http://www.w3.org/TR/prov-n/

```
entity(<http://.../isbn/000651409X>)
activity(:WritingTheBook)
wasGeneratedBy(<http://.../isbn/000651409X>,:WritingTheBook)
agent(:AmitavGhosh,
          [prov:type='prov:Person',foaf:name='AmitavGhosh'])
wasAttributedTo(<http://.../isbn/000651409X>,:AmitavGhosh,
                   [roles:witRole='roles:author'])
```

Note that the "qualified" versions are unnecessary at that level, relationships are n-ary

Definition of the constraints

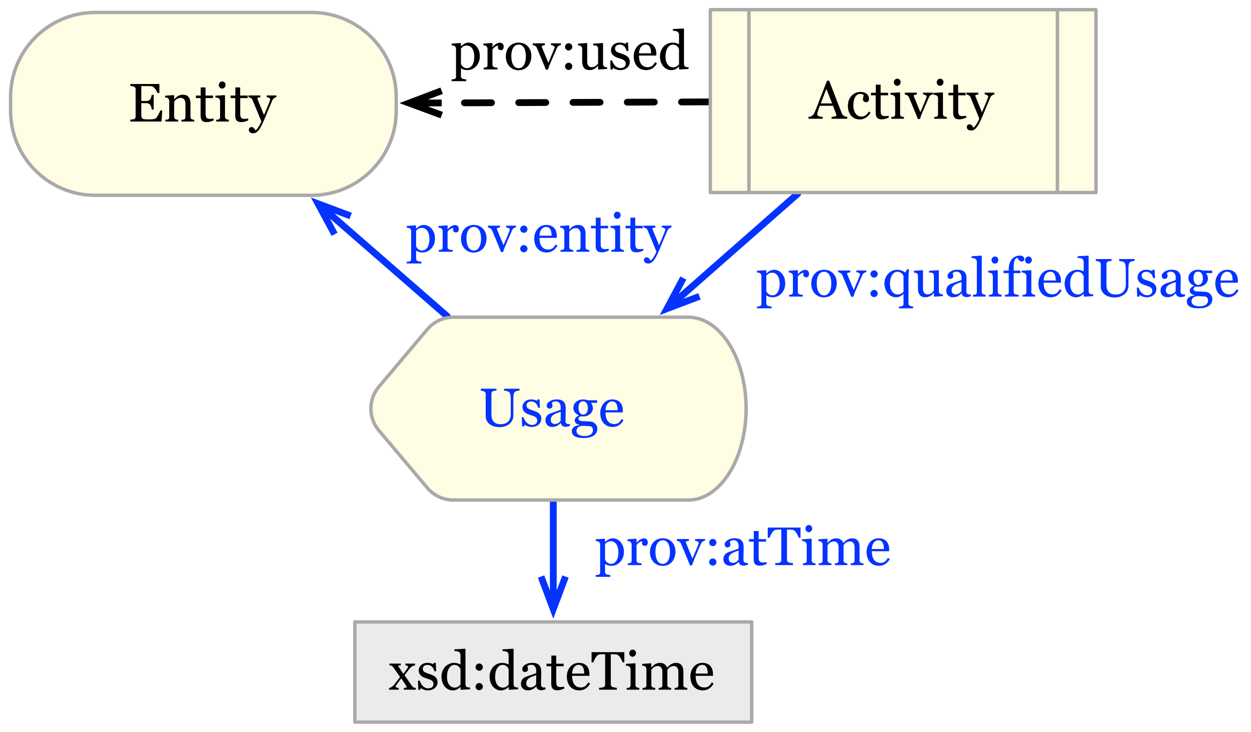
- A separate document defines the constraints on the abstract data model
  - http://www.w3.org/TR/prov-constraints/
- Constraints themselves are defined as a set of abstract rules
  - they may translated into:
    - (partially) into OWL
    - rules, e.g., using SPARQL
  - general constraint checkers on the abstract model are also doable


Inferences
Definitions
Constraints
Uniqueness Constraints
Ordering Constraints
Impossibility Constraints
Typing Constraints
PROV instance(s)
Normalization Activity
Normalized PROV instance(s)
Validation Activity
Validity

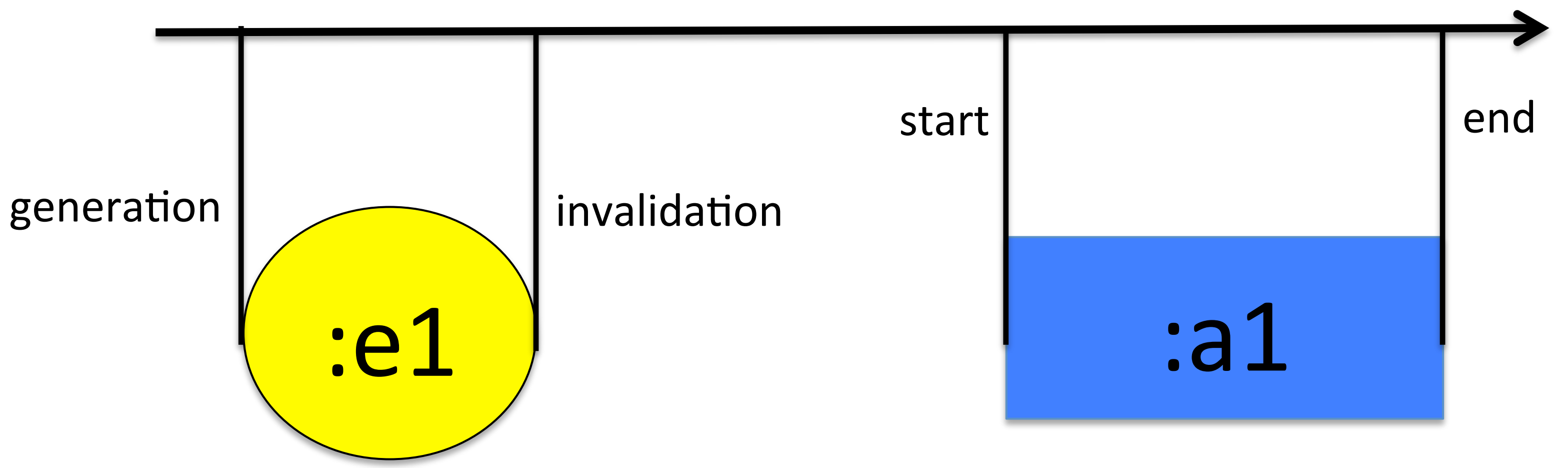
Normalization

From an RDF perspective

1. Expand
2. Everything is already merged by virtue of URIs
    - Blank nodes….

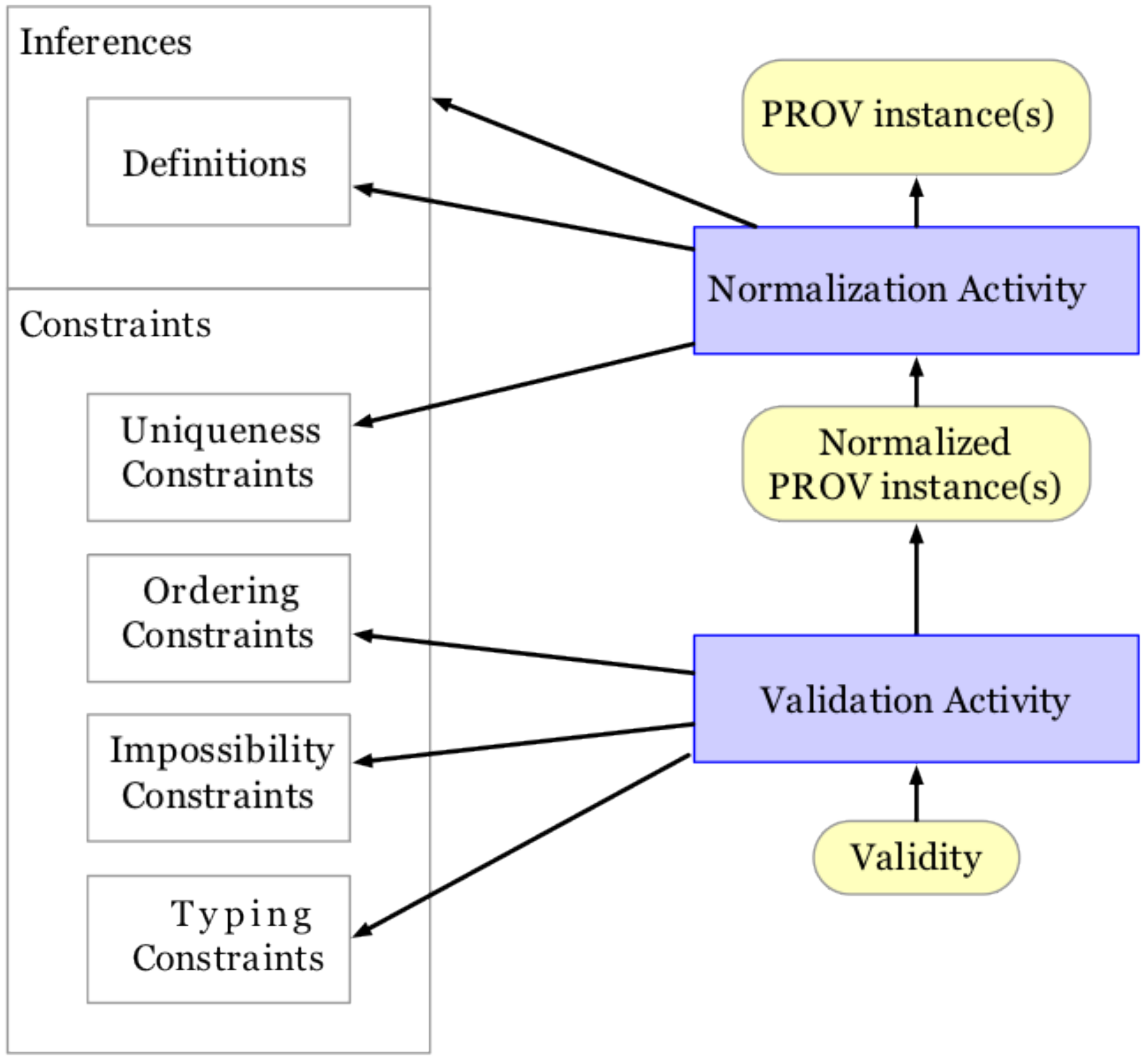


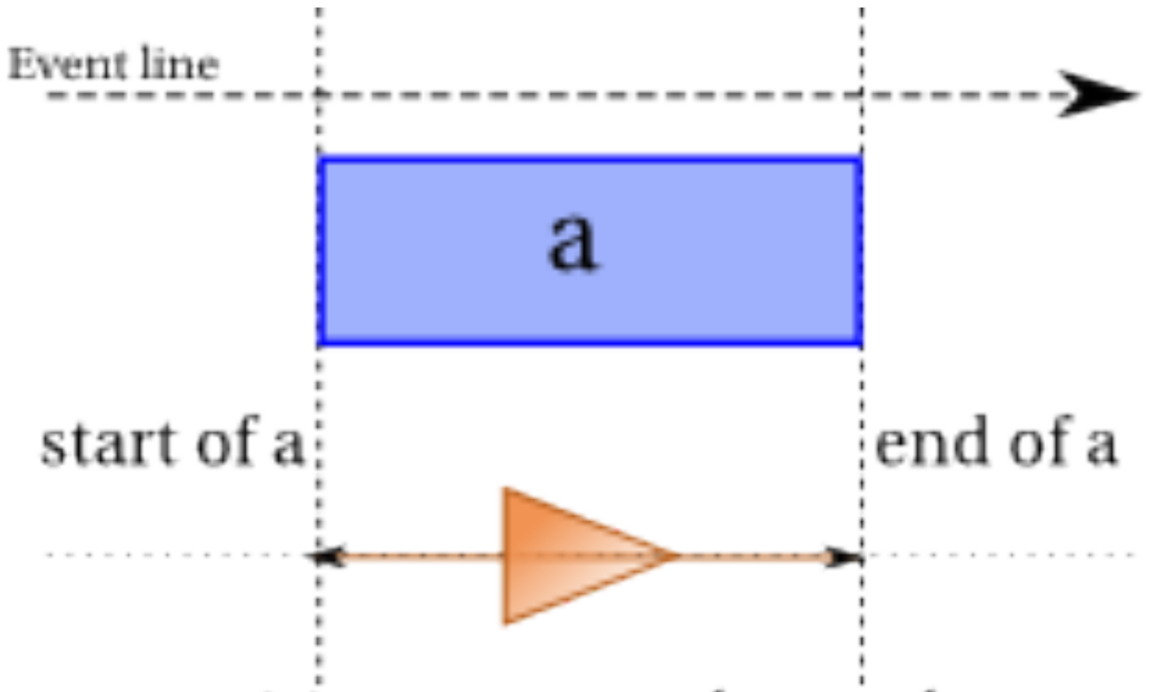
Events on a Lifetime



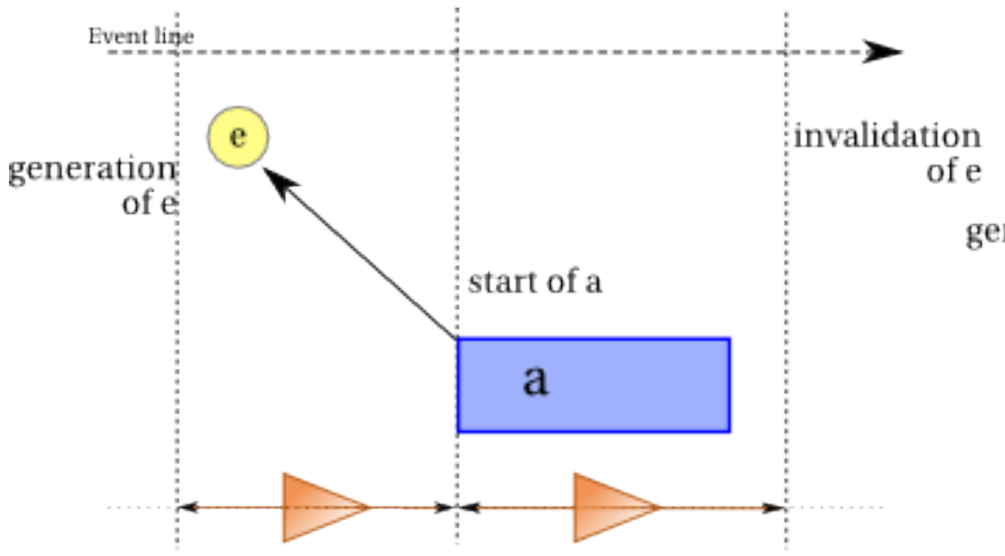
Uniqueness Constraints

```
select ?e where {
        ?e prov:qualifiedGeneration ?gen1 .
        ?gen1 prov:activity ?act .
        ?e prov:qualifiedGeneration ?gen2 .
        ?gen2 prov:activity ?act .
        FILTER (?gen1 != ?gen2)
    }
```

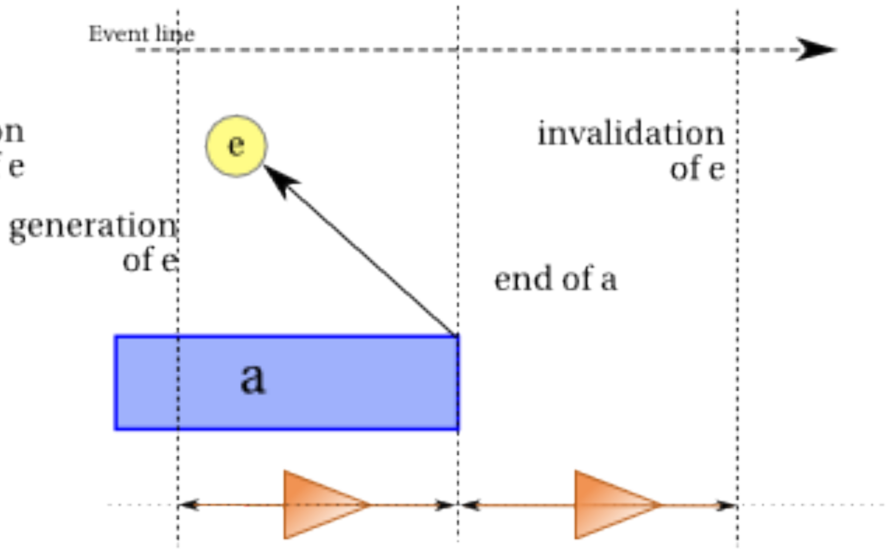

Inferences
Definitions
Constraints
Uniqueness Constraints
Ordering Constraints
Impossibility Constraints
Typing Constraints
PROV instance(s)
Normalization Activity
Normalized PROV instance(s)
Validation Activity
Validity



(a) start-precedes-end

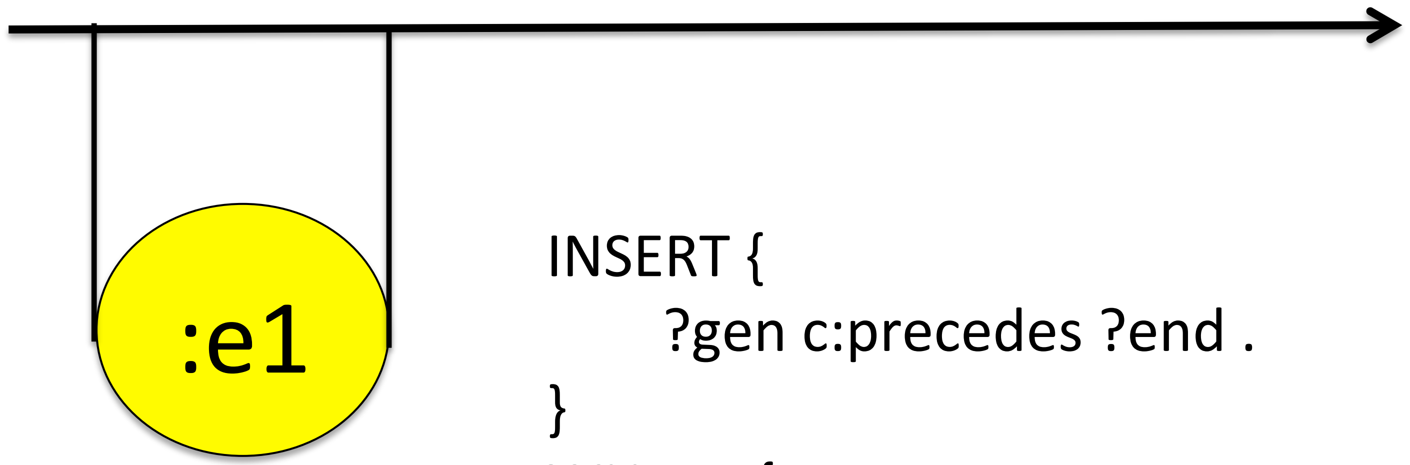


(a) wasStartedBy-ordering



(b) wasEndedBy-ordering

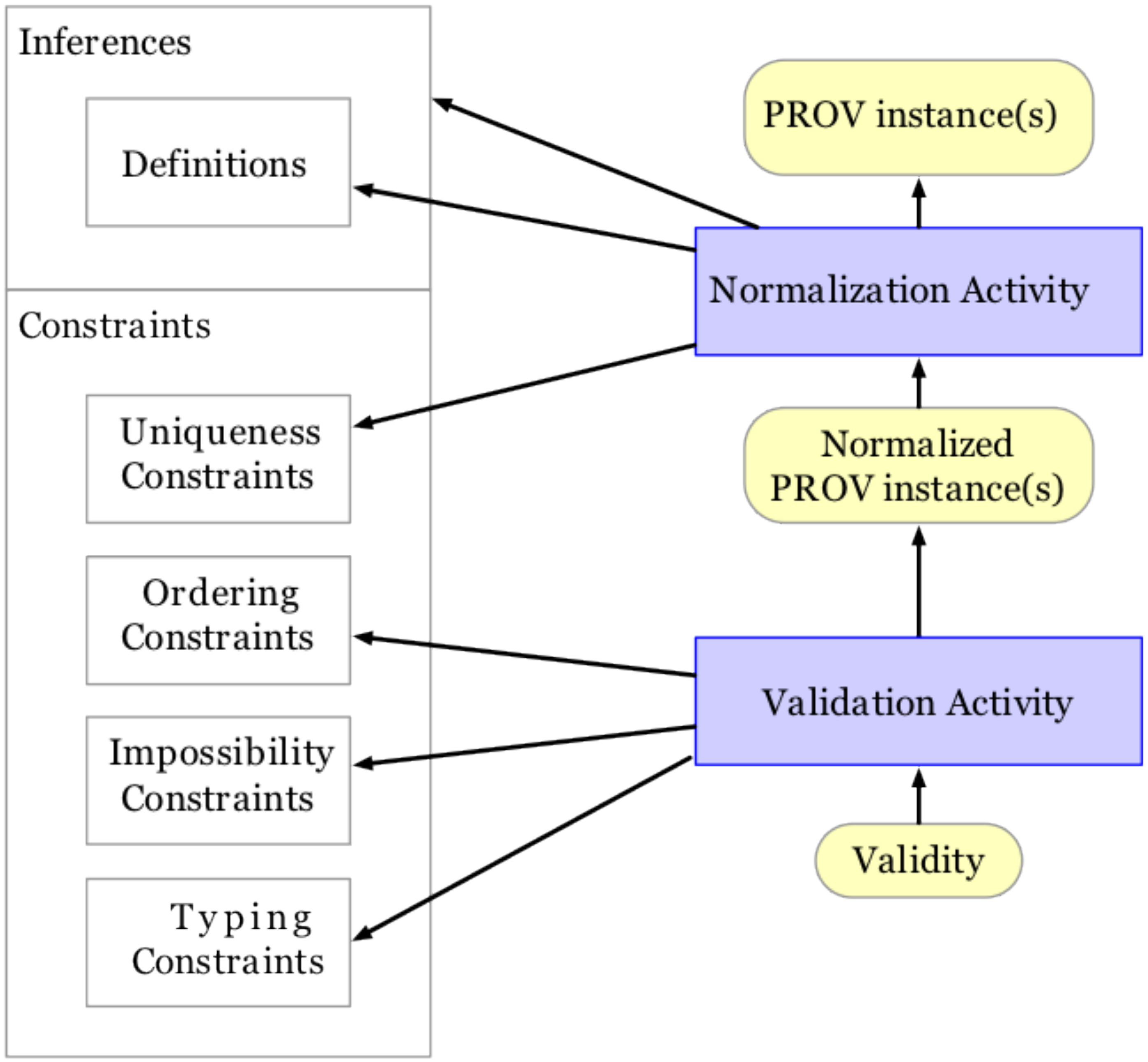
Create the event timeline



```
INSERT {
      ?gen c:precedes ?end .
}
WHERE {
     ?act a prov:Activity .
     ?act prov:qualifiedEnd ?end .
     ?act prov:qualifiedGeneration ?gen .
  }
```

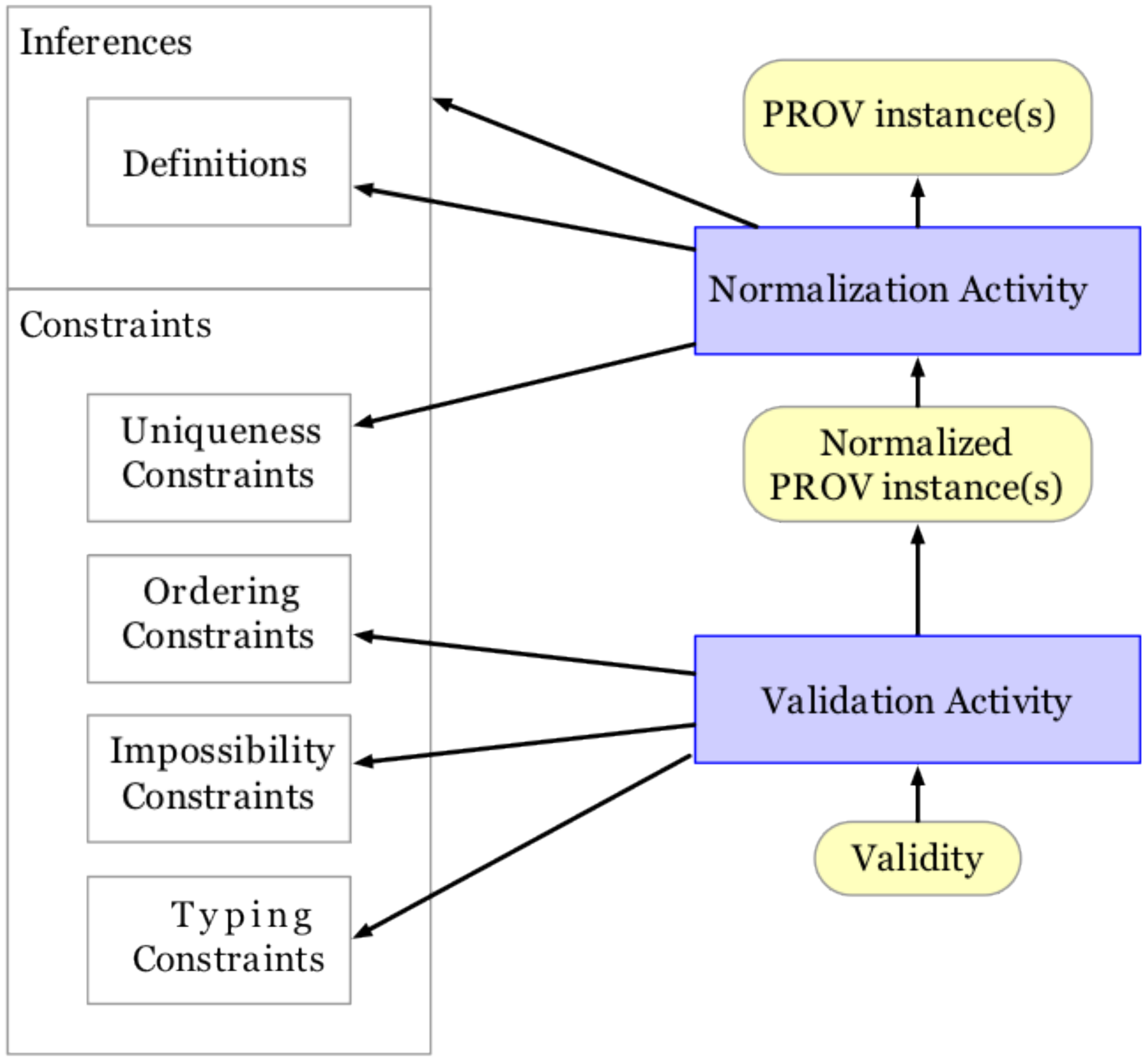
Check for cycles

```
select ?x where {
?x (c:precedes+|c:strictlyPrecedes+)/
    c:strictlyPrecedes ?x .
}
```


Inferences
Definitions
Constraints
Uniqueness Constraints
Ordering Constraints
Impossibility Constraints
Typing Constraints
PROV instance(s)
Normalization Activity
Normalized PROV instance(s)
Validation Activity
Validity

Impossibility Constraint
e.g. Activity & Entity disjoint

```
select ?e where {
    ?e a prov:Entity, prov:Activity .
}
```


Inferences
Definitions
Constraints
Uniqueness Constraints
Ordering Constraints
Impossibility Constraints
Typing Constraints
PROV instance(s)
Normalization Activity
Normalized PROV instance(s)
Validation Activity
Validity

Do you fill all the slots?

```
select ?asc where {
        ?asc a prov:Association .
        FILTER NOT EXISTS {
           ?a prov:qualifiedAssociation ?asc .
    }
}
```

Validators

- Prov-check
  - https://github.com/pgroth/prov-check
- Southampton Provenance Suite
  - https://provenance.ecs.soton.ac.uk