Leibniz-Institut für
Astrophysik Potsdam

Provenance Data Model

RAVE Use case

Provenance Meeting in Paris, April 2016

Kristin Riebe, AIP, GAVO

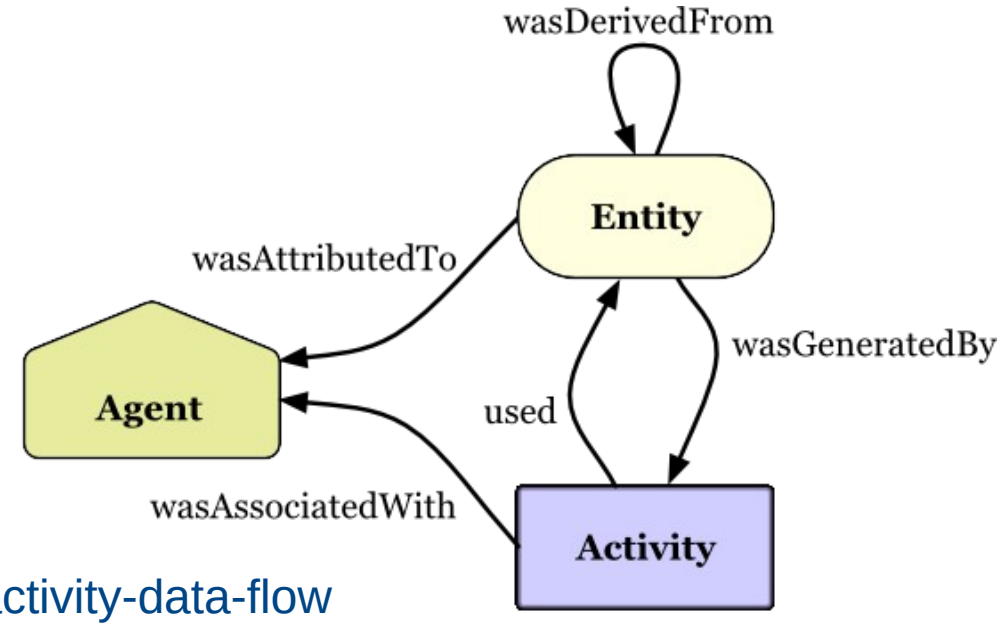
Provenance DM from W3C

http://www.w3.org/TR/prov-dm/

- 3 core classes:
  - Agent
  - Activity
  - Entity
- core relations:
  - used
  - wasGeneratedBy

  } activity-data-flow
  - wasDerivedFrom → data-flow
  - wasAttributedTo
  - wasAssociatedWith

  } responsibility view
- + many more classes and relations



Example: Reduced RAVE-fits file

- PROV-N notation
- 2 files
- 2 agents
- 2 activities

- relations

entity(rave:0645m522l0049.wav.fits, [prov:type = 'std:fits']
entity(rave:0645m522l0049.fits, [prov:type = 'std:fits']

Example: Reduced RAVE-fits file

- PROV-N notation
- 2 files
- 2 agents
- 2 activities

entity(rave:0645m522l0049.fits, [prov:type = 'std:fits']
entity(rave:0645m522l0049.wav.fits, [prov:type = 'std:fits']

agent(aao:Paul_Cass, [prov:type='prov:Person'])
agent(rave:Alessandro_Siviero, [prov:type='prov:Person'])

- relations

Example: Reduced RAVE-fits file

- PROV-N notation
- 2 files
- 2 agents
- 2 activities
- relations

```
entity(rave:0645m522l0049.fits, [prov:type = 'std:fits']
entity(rave:0645m522l0049.wav.fits, [prov:type = 'std:fits']

agent(aao:Paul_Cass, [prov:type='prov:Person'])
agent(rave:Alessandro_Siviero, [prov:type='prov:Person'])

activity(rave:act_observation, 2008-02-16T13:25:24, -,
      [ prov:type = 'obs:Observation'  ])
activity(rave:act_irafReduction, 2008-03-04T09:46:57, -,
      [ prov:type = 'std:reduction' ])
```

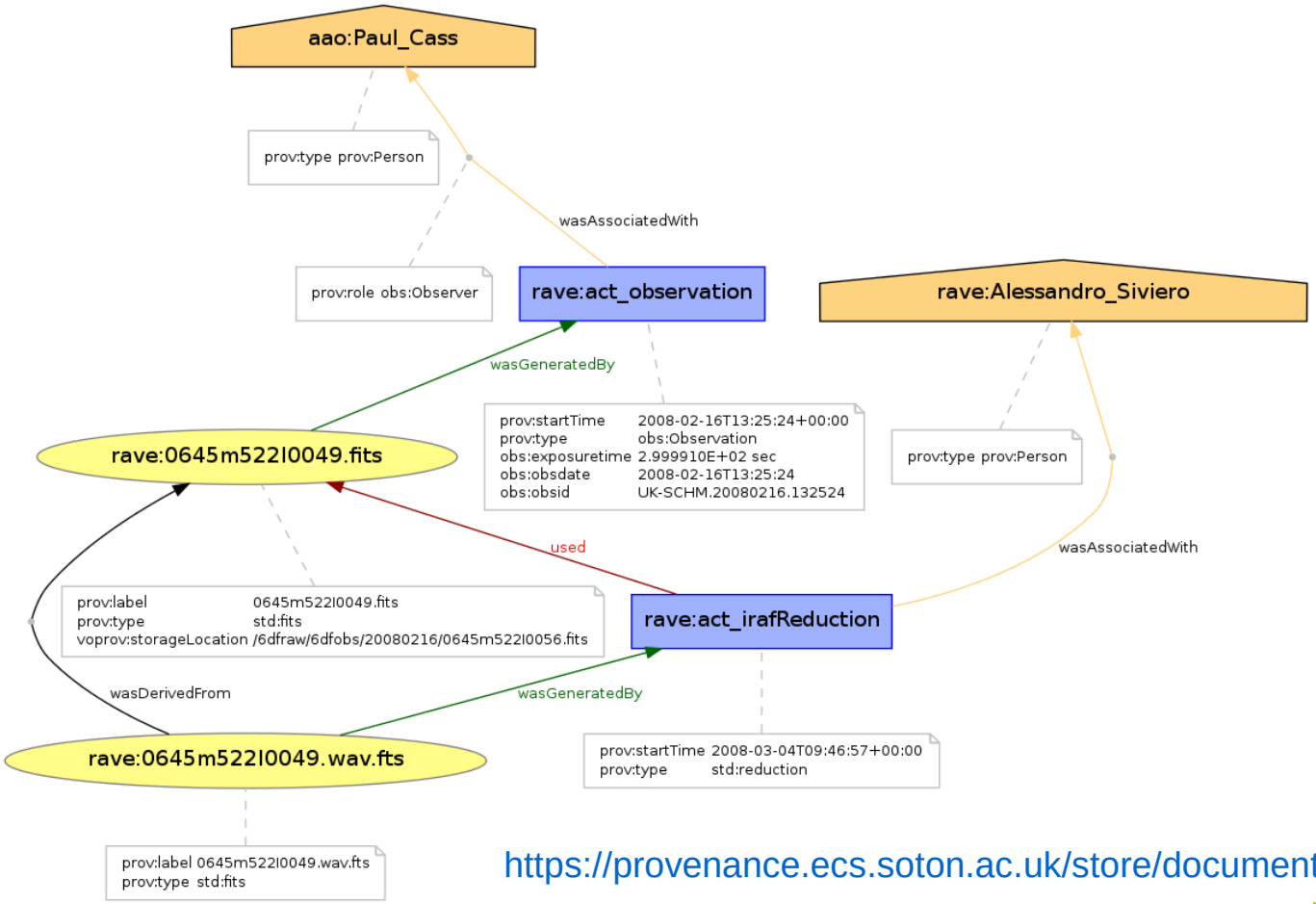
Example: Reduced RAVE-fits file

- PROV-N notation
- 2 files
- 2 agents
- 2 activities
- relations

```
entity(rave:0645m522I0049.fits, [prov:type = 'std:fits']
entity(rave:0645m522I0049.wav.fits, [prov:type = 'std:fits']

agent(aao:Paul_Cass, [prov:type='prov:Person'])
agent(rave:Alessandro_Siviero, [prov:type='prov:Person'])

activity(rave:act_observation, 2008-02-16T13:25:24, -,
      [ prov:type = 'obs:Observation'  ])
activity(rave:act_irafReduction, 2008-03-04T09:46:57, -,
      [ prov:type = 'std:reduction' ])

wasAssociatedWith(rave:act_observation, aao:Paul_Cass, -,
                        [ prov:role = 'obs:Observer' ])
wasAssociatedWith(rave:act_irafReduction, rave:Alessandro_Siviero, -)

wasGeneratedBy(rave:0645m522I0049.fits, rave:act_observation, -)
used(rave:act_irafReduction, rave:0645m522I0049.fits, -)
wasGeneratedBy(rave:0645m522I0049.wav.fts, rave:act_irafReduction, -)
wasDerivedFrom(rave:0645m522I0049.wav.fts, rave:0645m522I0049.fits)
```

Example: Reduced RAVE-fits file

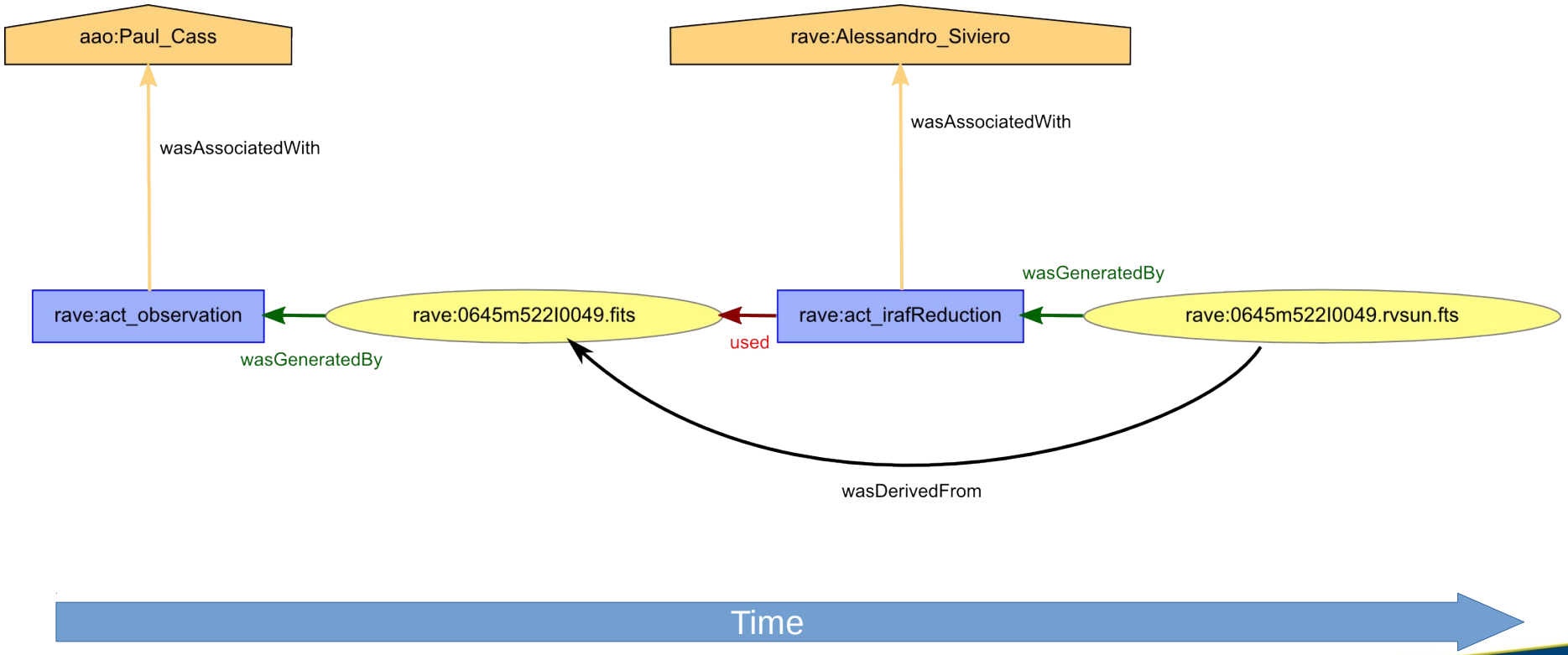
- Graph produced with ProvStore (using GraphViz):



https://provenance.ecs.soton.ac.uk/store/documents/84420/

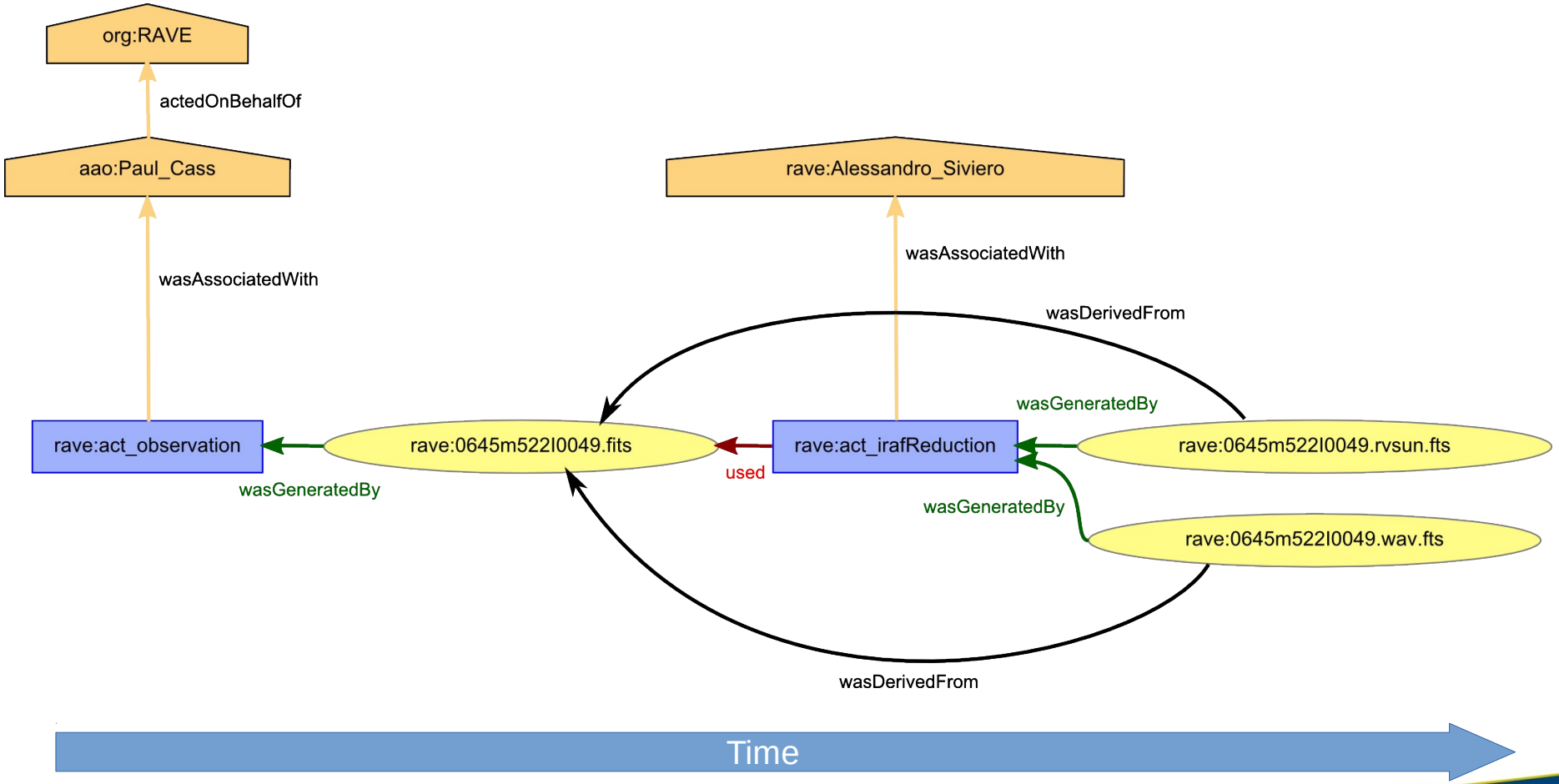
Example: Reduced RAVE fits-file

- Graph reordered, attributes hidden:

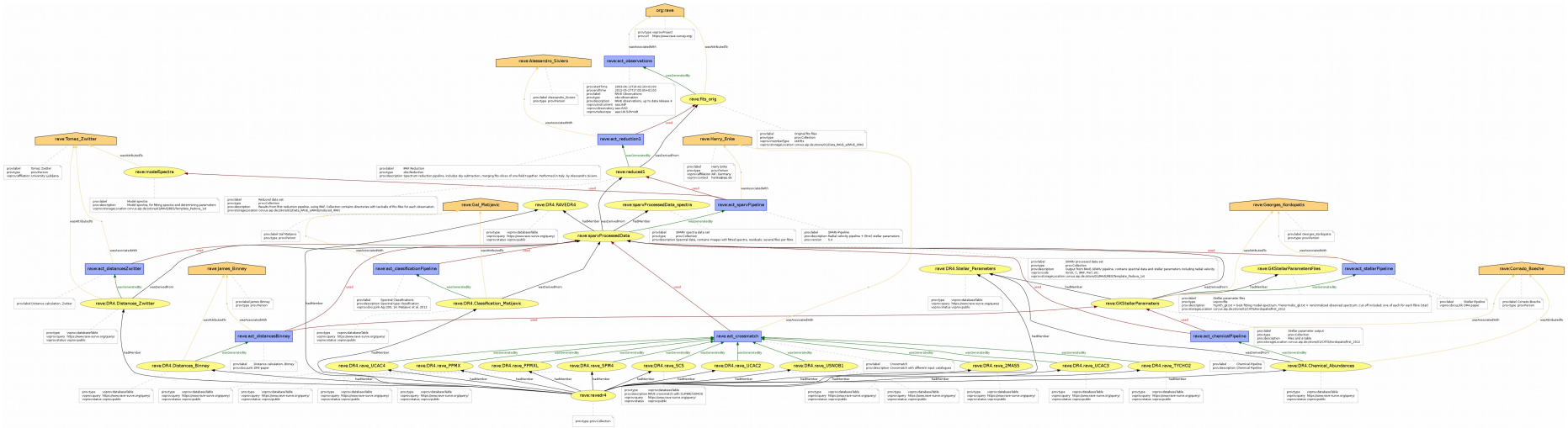


Example: Reduced RAVE fits-file

- Graph reordered, attributes hidden:

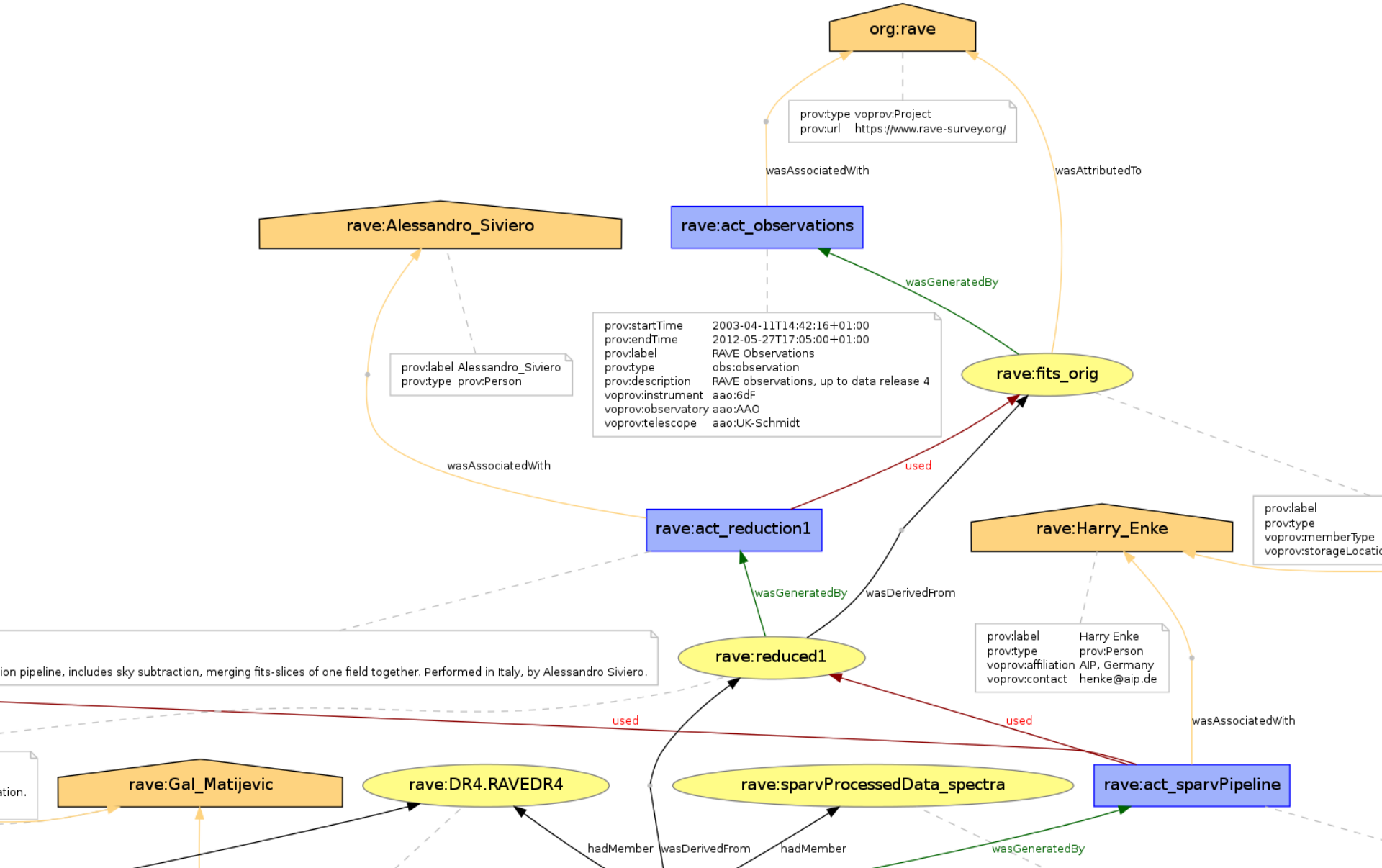


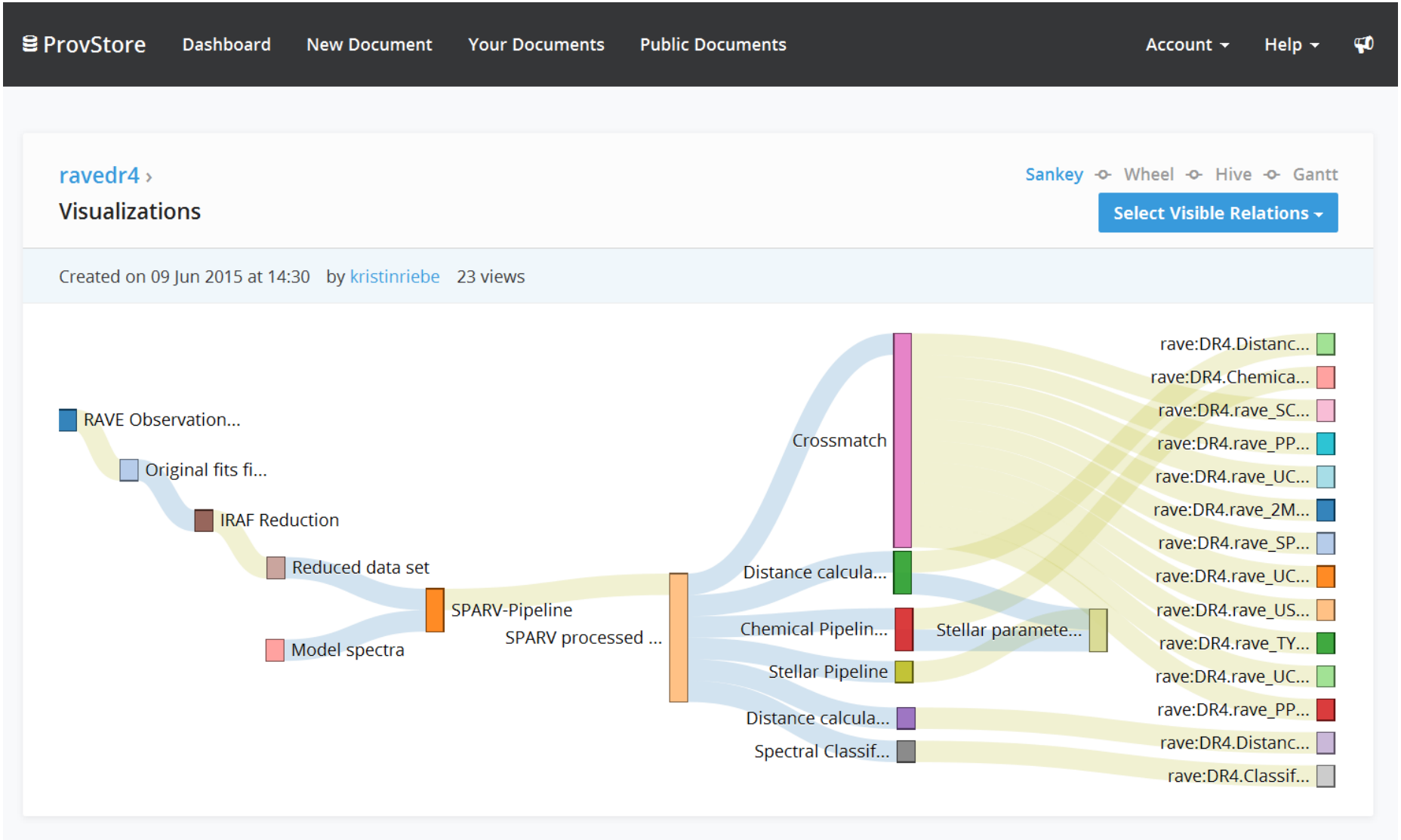
Example: RAVE database tables
(nearly complete history)



https://provenance.ecs.soton.ac.uk/store/documents/84064/

Example: RAVE database tables




ProvStore
Dashboard
New Document
Your Documents
Public Documents
Account
Help
ravedr4 ›
Visualizations
Sankey
Wheel
Hive
Gantt
Select Visible Relations
Created on 09 Jun 2015 at 14:30 by kristinriebe 23 views
RAVE Observation...
Original fits fi...
IRAF Reduction
Reduced data set
Model spectra
SPARV-Pipeline
SPARV processed ...
Crossmatch
Distance calcula...
Chemical Pipelin...
Stellar Pipeline
Distance calcula...
Spectral Classif...
Stellar paramete...
rave:DR4.Distanc...
rave:DR4.Chemica...
rave:DR4.rave_SC...
rave:DR4.rave_PP...
rave:DR4.rave_UC...
rave:DR4.rave_2M...
rave:DR4.rave_SP...
rave:DR4.rave_UC...
rave:DR4.rave_US...
rave:DR4.rave_TY...
rave:DR4.rave_UC...
rave:DR4.rave_PP...
rave:DR4.Distanc...
rave:DR4.Classif...

Webapp for RAVE provenance

- Testing how to implement the data model myself
- Simple setup using Django Framework with SQlite3 database
- Define classes “as is”, main provenance classes, one DB table for each:
  - **entity**
  - **activity**
  - **agent**
  - used -- foreign keys to activity, entity
  - wasGeneratedBy -- foreign keys to entity, activity
  - wasAssociatedWith -- foreign keys to entity, agent
  - hadMember -- foreign keys to entities (one with type collection)
  - wasDerivedFrom -- foreign keys to entities

Webapp for RAVE provenance

- Create views to show e.g.
  - provn-serialisation of the complete provenance
  - graph-representation
    - could also divide between the 3 different views:
      - data flow (entities)
      - process flow (activities)
      - responsibility view (agents)
  - list of activities, entities, agents
  - details for individual elements
- Provide detailed information for individual observations
  - given an obsId, return file names and locations of intermediate and raw files
- More use cases?

Use cases

- There are different types of users:
  - project manager: interested mainly in coarse data flow, involved processes (activities), not very detailed
  - “pipeline writer” (e.g. scientist from the project): interested in redoing parts of the pipeline, using different algorithms, testing influence of different parameters
  - “other scientist”: usually interested in science-ready data only (no need for raw observation files), quality assessment, error-bars, applicability of data, error tracking

Use cases

- Example: Project management:
  - Give me a visualisation of the data flow and the work flow, showing all involved activities, agents and resulting entities.
    - Interesting for PI of the project, someone writing a report, a funding agency
- Example: Pipeline analysis
  - Where are the raw fits-files? The flat-fields? Can I access the extracted spectrum for each fiber? Which processes were involved and where are they described?

Use cases

- Examples: Scientist:
  - Who created the stellar_parameters-table?
    - i.e.: get the agent associated with this entity, thus: retrieve details for this entity
  - Where do the values in column Teff_K come from? In which paper are the methods described? The uncertainties?
    - errors are in additional columns "e..."-something. Are things like this described in any other data model/standard?
  - Are intermediate files (spectrum png/ascii) for a given obsId available? How could I get them?
    - Or: who do I need to ask for them?
    - Need: permission/accessibility flag, contact details
  -

Use cases

- Examples: Scientist (continued):
  - How are values (for a given star) changing for each data release? What's the difference in processing?
    - First part can be answered with published data alone, provenance only needed for second question.
  - Are there multiple observations of the same star? If the derived heliocentric radial velocity differs more than the error bars suggest: what was causing this difference? (Which processing step(s)?)
  - What is the coverage of this survey? Compare intended/actual coverage for studies of completeness/selection effects.
    - Needs additional information on failed fibers per field

Implementation details: attributes

- common attributes for activity/entity/agent:
  - id
  - label
  - type
  - description
  - W3C: id is a qualified name, e.g. a string like: rave:DR4 as id for an entity
- additional attributes for each major class

Implementation details: attributes

- activity:
  - type: observation, reduction, classification, crossmatch, chemical pipeline, distances, other
  - docuLink: link to documentation, e.g. paper, webpage, ...

- entity:
  - type: prov:Collection, voprov:dataset, voprov:image
  - status (better: accessibility?): voprov:public, voprov:restricted, maybe also: "unavailable"
  - dataType: voprov:database, voprov:databaseTable,
  - voprov:directory, std:fits, std:votable

Implementation details: attributes

- different attributes needed for different types of activity/entity/agent
  - agent, type=”project”: webpage
  - agent, type=”person”: first name, last name, affiliation, email
- could use subclasses instead of 'type'-attribute
- or separate attributes from the tables
  - e.g. done in PROV-Man implementation of PROV-DM, http://nl.sharp-sys.com/provman/PROVman.html
  - could even use one common class for all of them, and make distinction between activity/entity/agent by another table field
- use one common class/table for all relations?

Discussions