IVOA Provenance: voprov python module (current status)

IVOA Provenance Group:
François Bonnarel (CDS), Mireille Louys (CDS),
Laurent Michel (OAS), Markus Nullmeier (KAH, GAVO),
Kristin Riebe (AIP, GAVO),
Michèle Sanguillon (LUPM),Mathieu Servillat (LUTH)

Provenance implementation steps

- **Collecting** the information
  - Application dependent
- **Informing** of provenance information existence
  - DATALINK
- **Generating** provenance serialization
  - use of voprov python module
- **Giving** serialized provenance information
  - SODA service
  - In a existing data file (FITS)
- **Selecting** data on provenance criteria
  - Nothing done

DATALINK example

- Example implementation = SSA protocol + DATALINK

http://pollux.graal.univ-montp2.fr/ssaserver/tsap?REQUEST=queryData&teff_min=3000&teff_max=3000&test=1

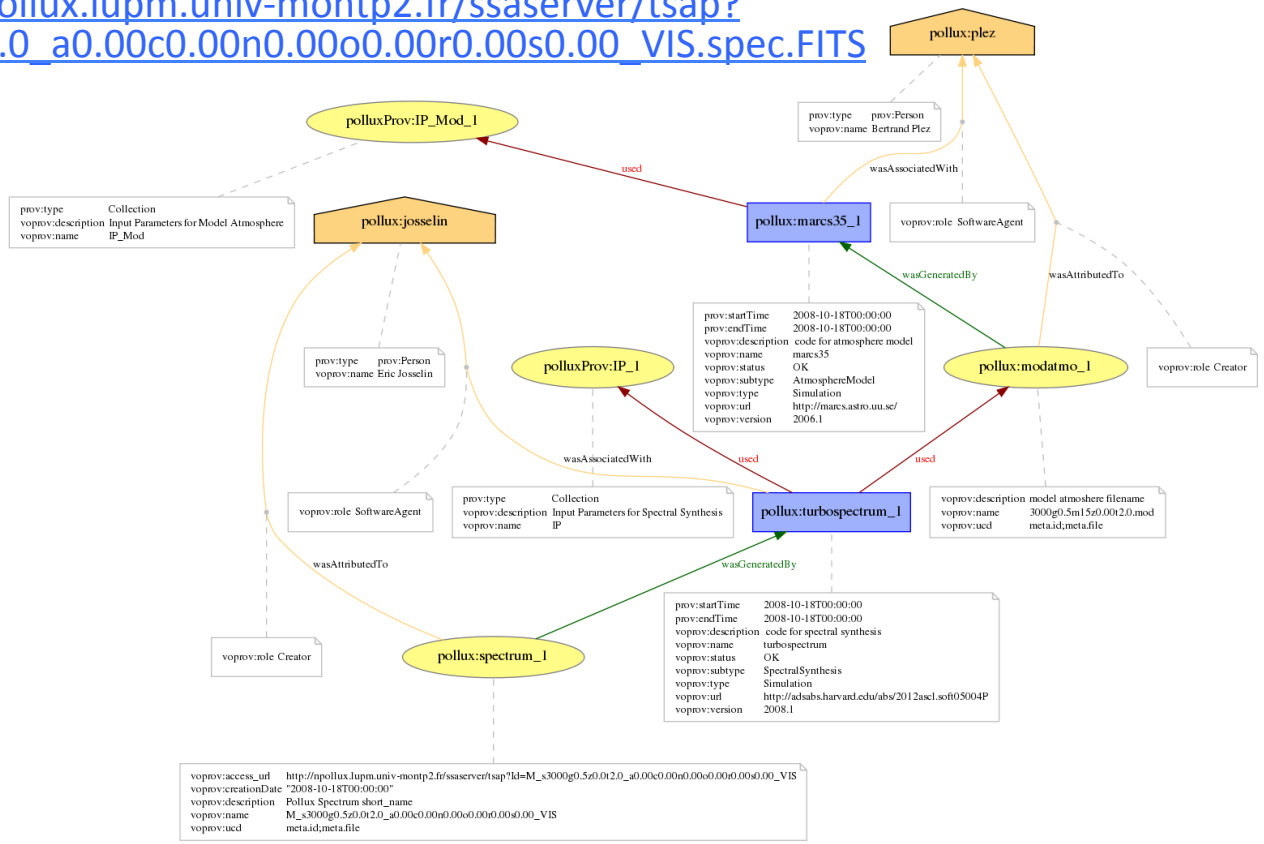
```
– <RESOURCE ID="provenance" type="meta" utype="adhoc:service">
    <PARAM arraysize="*" datatype="char" name="resourceIdentifier" value="ivo://graal.fr/datalink/prov"/>
    <PARAM arraysize="*" datatype="char" name="accessURL" value="http://pollux.graal.univ-montp2.fr/datalink/provenance"/>
  – <GROUP name="inputParams">
      <PARAM arraysize="*" datatype="char" name="uri" ref="Spectrum" value=""/>
    – <PARAM arraysize="*" datatype="char" name="format" value="">
        <DESCRIPTION>Format of the provenance file</DESCRIPTION>
      – <VALUE>
          <OPTION value="JSON"/>
          <OPTION value="PROVN"/>
          <OPTION value="VOTABLE"/>
          <OPTION value="SVG"/>
          <OPTION value="PNG"/>
          <OPTION value="PDF"/>
        </VALUE>
      </PARAM>
    – <PARAM arraysize="*" datatype="char" name="detail" value="">
        <DESCRIPTION>Detail Level of the provenance description</DESCRIPTION>
      – <VALUE>
          <OPTION value="min"/>
          <OPTION value="medium"/>
          <OPTION value="max"/>
        </VALUE>
      </PARAM>
    </GROUP>
  </RESOURCE>
```

SODA service example

- Example SODA provenance service :

  http://pollux.graal.univ-montp2.fr/datalink/provenance?entity_id=http://dev-pollux.lupm.univ-montp2.fr/ssaserver/tsap?Id=M_s3000g0.5z0.0t2.0_a0.00c0.00n0.00o0.00r0.00s0.00_VIS.spec.FITS**&outputFormat**=svg

STILL IN TEST



voprov python module

- Based on prov module (1.5.0 version) developped by Trung Dong Huynh (Southampton University)
  - implementing W3C provenance data model
  - Output formats of serialized data : PROV-N, JSON
  - Graphics output fomats: PDF, PNG, SVG
- voprov:
  - Adapted to IVOA provenance data model (Flow class, hadStep relationship)
  - Addition of the output serialized format: VOTable
  - Beta version: https://github.com/sanguillon/voprov

Use of prov

1 – Import the module

```
from prov.model import ProvDocument
from prov.dot import prov_to_dot
```

2 – Create the provenance document

```
provdoc = ProvDocument()
```

3 – Define the namespaces

```
provdoc.add_namespace('prov', 'http://www.w3.org/ns/prov#')
provdoc .add_namespace('voprov', 'http://www.ivoa.net/documents/dm/provdm/voprov/')
provdoc .add_namespace('example', 'http://example.domain.fr/documents/dm/provdm/ns#')
```

Use of prov

4 – Declare the entities, activities, agents

```
provdoc.entity('example:paramfile_11', {'prov:label':'Param_x3000'})
provdoc.entity('example:calibratedFile_7023', {'prov:label':'M_x3000_7023'})
provdoc.activity('example:calibration_17622', '2008-10-18T00:00:00', '2008-10-18T00:00:00',\
        other_attributes={'prov:label': 'Flatfield Calibration' })
provdoc.agent('example:AG_124', {'prov:name': 'Michel Dupont'})
```

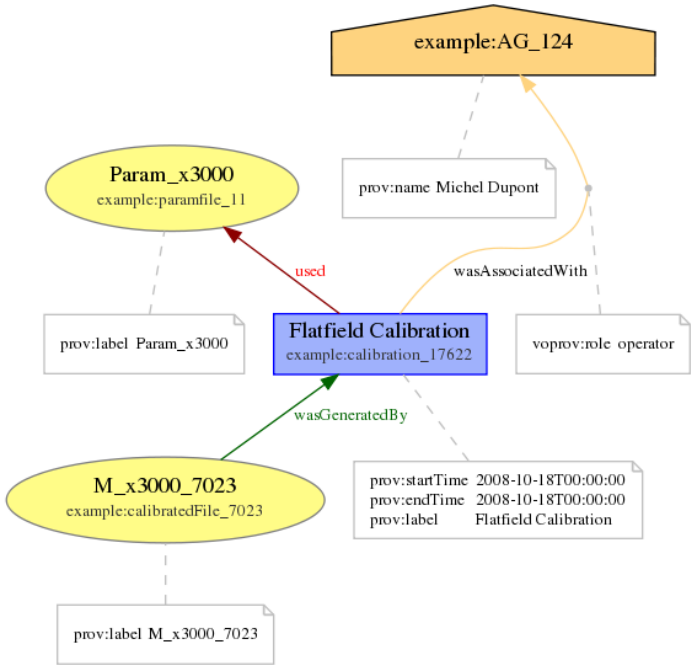
5 – and their relations

```
provdoc.used('example:calibration_17622', entity='example:paramfile_11')
provdoc.wasGeneratedBy('example:calibratedFile_7023', 'example:calibration_17622')
provdoc.wasAssociatedWith('example:calibration_17622', agent='example:AG_124',\
        other_attributes={'voprov:role': 'operator'})
```

Use of prov

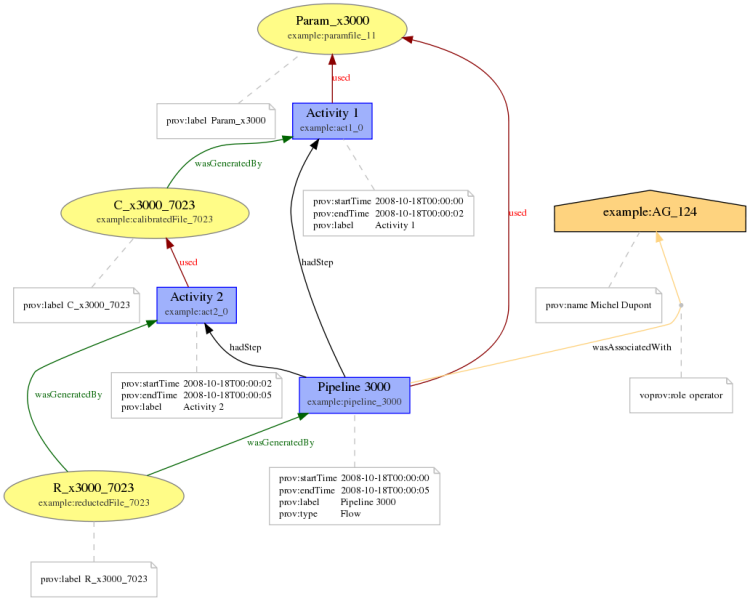
6 – Construct the serialized files

```
f_out = open('res.json','w')
f_out.write(provdoc.serialize(indent=2))
f_out.close()
f_out = open('res.provn','w')
f_out.write(provdoc.get_provn())
f_out.close()
provdoc.serialize(format='xml', destination='res.xml')
f_out = open('res.votable','w')
f_out.write(provdoc.serialize(format='votable',indent=2))
f_out.close()
dot = prov_to_dot(provdoc, use_labels=True)
dot.write_png('res.png')
dot.write_svg('res.svg')
dot.write_pdf('res.pdf')
```



Using voprov for workflows

```
…
provdoc.entity('example:paramfile_11', {'prov:label':'Param_x3000'})
provdoc.entity('example:calibratedFile_7023', {'prov:label':'C_x3000_7023'})
provdoc.entity('example:reductedFile_7023', {'prov:label':'R_x3000_7023'})
provdoc.activity('example:pipeline_3000', '2008-10-18T00:00:00', '2008-10-18T00:00:05', {'prov:type': 'Flow','prov:label':
'Pipeline 3000' })
provdoc.activity('example:act1_0', '2008-10-18T00:00:00', '2008-10-18T00:00:02',{'prov:label': 'Activity 1' })
provdoc.activity('example:act2_0', '2008-10-18T00:00:02', '2008-10-18T00:00:05',{'prov:label': 'Activity 2' })
provdoc.agent('example:AG_124', {'prov:name': 'Michel Dupont'})
provdoc.used('example:pipeline_3000', entity='example:paramfile_11')
provdoc.used('example:act1_0', entity='example:paramfile_11')
provdoc.used('example:act2_0', entity='example:calibratedFile_7023')
provdoc.wasGeneratedBy('example:calibratedFile_7023', 'example:act1_0')
provdoc.wasGeneratedBy('example:reductedFile_7023', 'example:act2_0')
provdoc.wasGeneratedBy('example:reductedFile_7023', 'example:pipeline_3
provdoc.wasAssociatedWith('example:pipeline_3000', \
        agent='example:AG_124', other_attributes={'voprov:role': 'operator'})
provdoc.hadStep('example:pipeline_3000', 'example:act1_0')
provdoc.hadStep('example:pipeline_3000', 'example:act2_0')
…
```



Using voprov for VOTable serialization

…

f_votable.write(provdoc.serialize(format='votable',indent=2))

…

```
<?xml version="1.0" encoding="UTF-8"?>
<VOTABLE version="1.2" xmlns:voprov="http://www.ivoa.net/documents/dm/provdm/voprov/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.ivoa.net/xml/VOTable/v1.1 http://www.ivoa.net/xml/VOTable/VOTable-1.1.xsd">
 <RESOURCE type="provenance">
  <DESCRIPTION>Provenance VOTable</DESCRIPTION>
…
   <TABLE name="Entity" utype="prov:entity">
    <FIELD arraysize="*" datatype="char" name="id" utype="prov:entity.id"/>
    <FIELD arraysize="*" datatype="char" name="label" utype="prov:label"/>
    <DATA>
     <TABLEDATA>
      <TR>
       <TD>example:paramfile_11</TD>
       <TD>Param_x3000</TD>
      </TR>
      <TR>
       <TD>example:calibratedFile_7023</TD>
       <TD>M_x3000_7023</TD>
      </TR>
     </TABLEDATA>
    </DATA>
   </TABLE>
…
 <INFO name="QUERY_STATUS" value="OK"/>
 </RESOURCE>
</VOTABLE>
```

To be discussed: voprov & description side

- In voprov, the Entity/Activity Description is a part of the Entity/Activity defclaration => no possibility to describe only to Description Part
- We have different ways to express the « EntityDescription »:

```
doc.entity("pollux:spectrum_1", { \
    "prov:label":  "M_s3000g0.5z0....",\
    "voprov:desc_type": "spectrum",\
    "voprov:desc_label": "Pollux Spect." })
```

```
doc.entity("pollux:spectrum_1", { \
    "prov:label":  "M_s3000g0.5z0....",\
    "voprov:entityDescription": { \
        "voprov:type": "spectrum",\
        "voprov:label": "Pollux Spect." })
```

```
doc.entity("pollux:spectrum_1", { \
    "prov:label":  "M_s3000g0.5z0....",\
    "voprov:entityDescription_id": "pollux:desc_124"})
```

```
doc.entity("pollux:spectrum_1", { \
    "prov:label":  "M_s3000g0.5z0....",\
    "voprov:entityDescription": { "id": "pollux:desc_124"}})
```

To be discussed: voprov & description side

- In a **Json serialization** there is no problem to implement the different possibilities:

```
entity : {
     "pollux:spectrum_1": {
   "prov:label":  "M_s3000g0.5z0....",
   "voprov:desc_type": "spectrum",
   "voprov:desc_label": "Pollux Spect."
  },
```

```
entity : {
     "pollux:spectrum_1": {
   "prov:label":  "M_s3000g0.5z0....",
   "voprov:entityDescription": {
     "voprov:desc_type": "spectrum",
     "voprov:desc_label": "Pollux Spectr."
    }
  },
```

```
entity : {
     "pollux:spectrum_1": {
   "prov:label":  "M_s3000g0.5z0...",
   "voprov:entityDescription": "pollux:desc_124"
},
```

- How to serialize the EntityDescription in a **VOTABLE**?

Idem
for activity parameters

provdoc.activity('example:calib_17622', '2008-10-18T00:00:00', '2008-10-18T00:00:00',
{'prov:label': 'Flatfield Calibration',\
'voprov:parameters':{'example:temp':'7000.', 'example:z':'0.5'} })

provdoc.activity('example:calib_17622', '2008-10-18T00:00:00', '2008-10-18T00:00:00',
{'prov:label': 'Flatfield Calibration', 'example:temp':'7000.', 'example:z':'0.5'})

```
"activity": {
        "example:calib_17622": {
                "prov:label": "Flatfield Calibration »,
                 "prov:endTime": "2008-10-18T00:00:00 »,
                "prov:startTime": "2008-10-18T00:00:00",
                "example:temp": "7000.",
                "example:z": "0.5"
            }
        },
```

voprov: to be done

- Reading VOTable
- Implementing all the attributes of VO Provenance Data Model classes
- Do we have to implement all the W3C relations ?
- Splitted from prov ?
- …

FITS files: To be discussed

- Different possibilities to define the provenance:
  - One keyword equal to a Web service URL to retrieve the provenance
  - One HDU identified as a HDU provenance in a given serialized format (PROV-N, JSON, XML, VOTable) or graphic one ?