Data Ingestion with python-icat

Rolf Krahl

ICAT Meeting @ 11th NOBUGS, Copenhagen, October 2016

python-icat — Python interface to ICAT and IDS

python-icat is a library for writing Python programs that access an ICAT service using the SOAP interface based on Suds.

Design Goals

- Keep the general structure and flexibility of Suds.
- Remove annoying details.
- Simplify things where possible.
- Make use object oriented design.
- Provide useful tools around all this.

Example: Upload a Datafile

Create a Dataset and Upload a Datafile

```
import icat
import icat.config
from icat.query import Query

conf = icat.config.Config(ids="mandatory").getconfig()
client = icat.Client(conf.url, **conf.client_kwargs)
client.login(conf.auth, conf.credentials)

query = Query(client, "Investigation",
              conditions={"name": "= '08100122-EF'"})
inv = client.assertedSearch(query)[0]
dst = client.assertedSearch("DatasetType[name='raw']")[0]
ds = client.new("dataset", name="e201217", complete=False,
                investigation=inv, type=dst)
ds.create()
fname = "e201217.nxs"
dff = client.assertedSearch("DatafileFormat[name='NeXus']")[0]
datafile = client.new("datafile", name=fname,
                      dataset=ds, datafileFormat=dff)
client.putData(fname, datafile)
```

Dump and Ingest

- Scripts `icatdump` and `icatingest` that dump the content of an ICAT to a file and ingests it from the file respectively.
- Backends for YAML and for XML dump file format.
- Other file formats may be added by a backend module.
- Original intent was a debugging tool for ICAT, but has grown out of this use case in the meanwhile.

icatdump and icatingest

```
$ icatdump.py -s root -f XML -o dump.xml
$ icatingest.py -s root -f XML -i dump.xml
```

Data Ingest

- `icatingest` turns out to be useful as a general ingestion tool.
- It reads ICAT entity objects from the ingest file and creates them in the ICAT server.
- The structure of the objects in the file corresponds to the ICAT schema.
- As an option, Datafile objects will not be created in the ICAT, but the corresponding file will be uploaded to IDS instead.
- Workflow at the beamline:
  1. Write datafiles.
  2. Write ingest file.
  3. Call `icatingest`.

Ingest File Format

Example Ingest File

```
<?xml version="1.0" encoding="utf-8"?>
<icatdata>
  <!-- <head>[...]</head> -->
  <data>
    <dataset id="Dataset_001">
      <complete>false</complete>
      <endDate>2016-06-05T12:36:31+02:00</endDate>
      <name>e201218</name>
      <startDate>2016-06-05T07:48:07+02:00</startDate>
      <investigation name="08100122-EF" visitId="1.1-P"/>
      <type name="raw"/>
      <parameters>
        <stringValue>NiMnGa T=293.15 K</stringValue>
        <type name="comment"/>
      </parameters>
      <parameters>
        <numericValue>5.3</numericValue>
        <type name="magnetic field" units="T"/>
      </parameters>
    </dataset>
    <datafile>
      <name>e201218.nxs</name>
      <datafileFormat name="NeXus"/>
      <dataset ref="Dataset_001"/>
    </datafile>
  </data>
</icatdata>
```

Referencing Objects

Entity objects in the ingest file generally need to refer to other objects. E.g. a Dataset needs to refer to its Investigation and DatasetType.

Reference by Attributes

```
<investigation name="08100122-EF" visitId="1.1-P"/>
<type name="raw"/>
<datafileFormat name="NeXus"/>
```

References are needed to set many to one relations. The referenced object must already exist and must be unique.

Referencing Objects

Reference by Id

```
<dataset id="Dataset_001">
</dataset>
<datafile>
  <dataset ref="Dataset_001"/>
</datafile>
```

You can reference objects defined in the same ingest file by id. The identifier is an arbitrary string. The referenced object must be defined in the same data element before any reference to it.

Referencing Objects

As a convenience, it is also possible to define an id to reference an already existing object.

Define an Id for an existing object

```
<datasetRef id="Dataset_001"
            name="e201218"
            investigation.name="08100122-EF"
            investigation.visitId="1.1-P"/>
```

The reference element must have an id attribute and the attributes to identify the referenced object. It must not have any subelements.

Conclusion

- `icatingest` can be used as a general ingestion tool.
- Loose coupling of ingestion with experiment control workflow.
- Aside from creating the same datafiles as usual, the experiment only need to produce one additional ingest file and call one external script.
- Standard file format for the ingest file: XML.
- Ingestion can be postponed and occour at any time after the experiment.