Renku and Astronomy OAS

V. Savchenko
for CDCI, ISDC

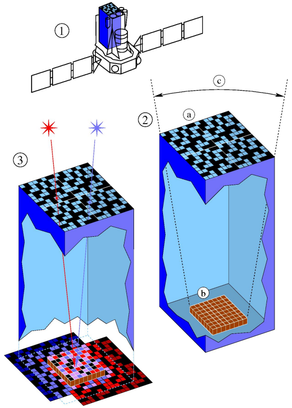
CERN, Renku-Reana meeting
26/06/2018

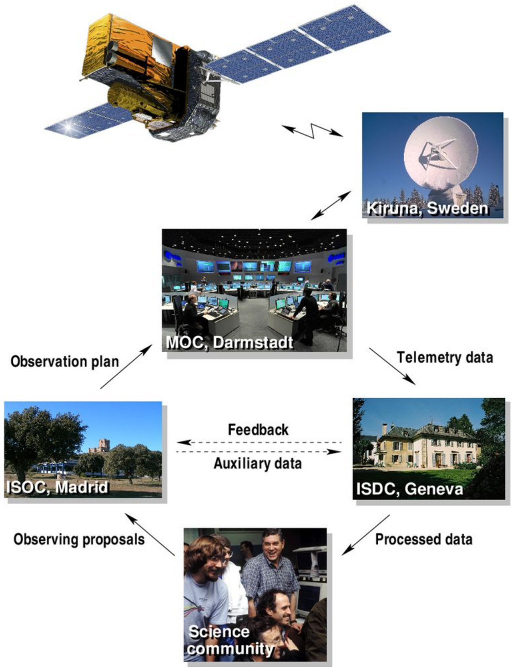
UNIVERSITÉ
DE GENÈVE

OBSERVATOIRE DE GENÈVE
FONDÉ EN 1772

Common Data Center Infrastructure (CDCI) and INTErnational Gamma-Ray Laboratory (INTEGRAL)

Gamma-Ray Astronomy is hard: mirrors can not be used, and the signal is encoded with mask projections. The data analysis is a complex process of reconstructing source properties.





INTEGRAL **S**cience **Data Center** (Versoix) is in charge of

- primary data processing
- data and software distribution
- quick-look analysis
- prompt investigation of transient astronomical events (including GW, UH Neutrino, etc)

CDCI supports data centers for INTEGRAL, POLAR, Gaia, etc

Challenges

Data Center (ISDC, or other CDCI-supported DC) is an interface between an experiment (e.g. INTEGRAL) and astronomical community, which features **diverse expertise but shared interests**

Astronomers are highly collaborative and want to easily **explore diverse data and data analysis options** at different levels of reduction complexity, **verify (repeat), and reuse the analysis** in diverse and evolving infrastructures, and integrate the analysis in **interoperable federated workflows**.

Astronomy, especially transient, demands rapid (automated, when possible) **dissemination of the scientific results**, clearly tracking origin of the data and code (e.g. **DOI**).

Astronomical events do not generally repeat: it is vital to **guarantee long-term availability of the data and the pipeline**

Routine operations at the data center include **Quick Look Analysis** with interactive interface to **long-running HPC-like** analysis pipeline. Reproducibility based on only on sharing Jupyter notebooks is not sufficient.

Online Analysis System: platform & infrastructure

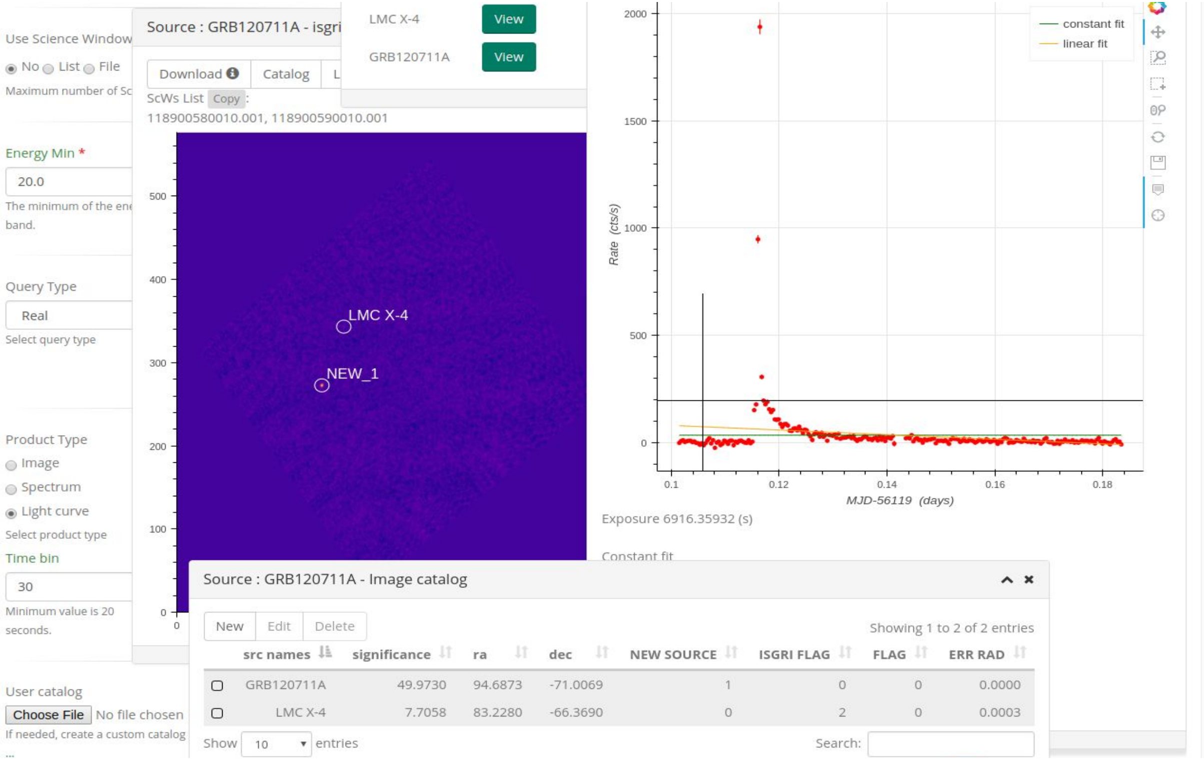
Frontend for easy data presentation and exploration

Several levels of **API** to integrate in interactive data analysis workflow, e.g. in Jupyter, and a CLI

Backend pipeline engine executes the workflow and stores results

The pipeline engine and analysis definition is open-source and **can be also executed offline** (no black box services)

Frontend



Use Science Window

● No ○ List ○ File

Maximum number of Sc

Energy Min *

20.0

The minimum of the ene band.

Query Type

Real

Select query type

Product Type

○ Image

○ Spectrum

● Light curve

Select product type

Time bin

30

Minimum value is 20 seconds.

User catalog

Choose File No file chosen

If needed, create a custom catalog

...

Source : GRB120711A - isgri

ScWs List Copy :

118900580010.001, 118900590010.001

Exposure 6916.35932 (s)

Constant fit

Source : GRB120711A - Image catalog

New Edit Delete

Showing 1 to 2 of 2 entries

	src names	significance	ra	dec	NEW SOURCE	ISGRI FLAG	FLAG	ERR RAD
□	GRB120711A	49.9730	94.6873	-71.0069	1	0	0	0.0000
□	LMC X-4	7.7058	83.2280	-66.3690	0	2	0	0.0003

Show 10 entries

Search:

INTEGRAL pipeline backend in CDCI OAS

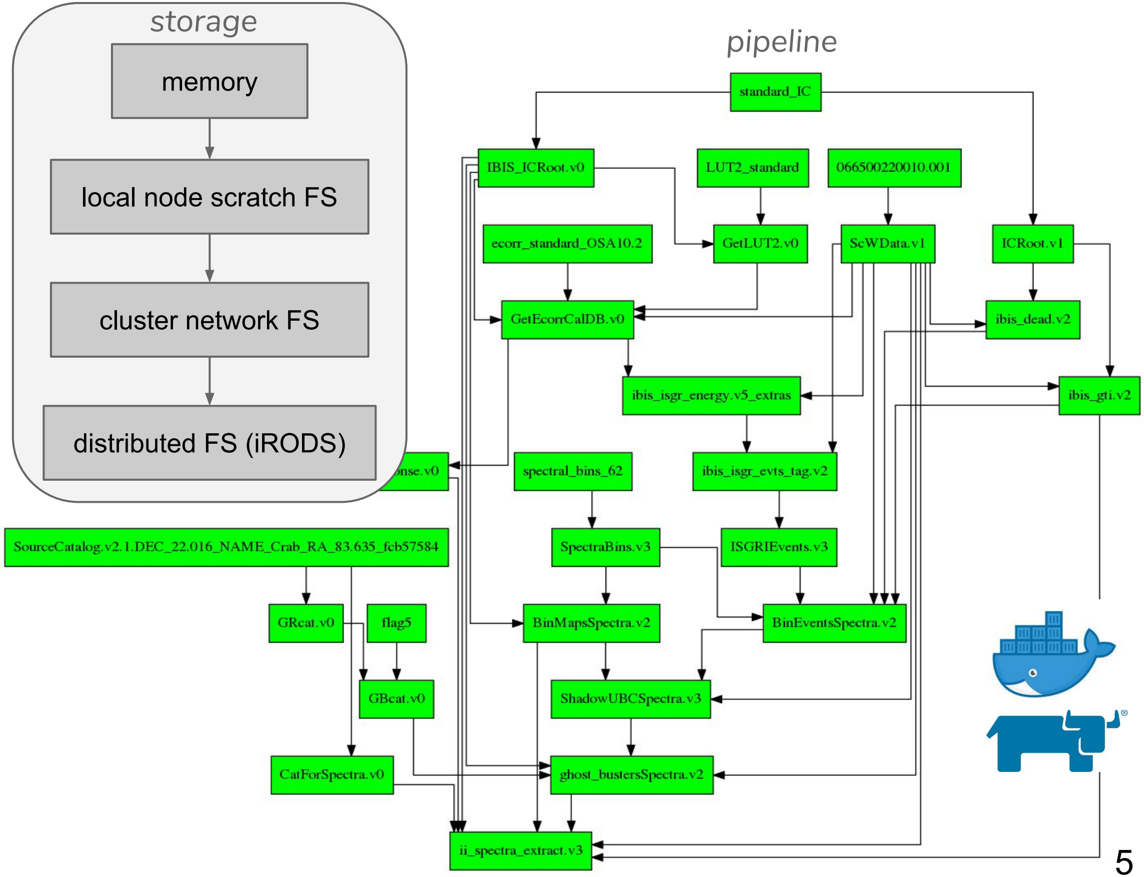
Declarative analysis definition is separated from scheduling and storage.

The pipeline is composed of analysis nodes with no side effects. Pipeline execution consists in cascading resolution of node dependencies.

Storage is a hierarchical immutable cache of the pipeline results, indexed with data provenance metadata expressed as directed acyclic graphs.

Lacking:

Easier collaboration, privacy restrictions, flexible exploring the result database



Astronomy OAS - Renku integration status

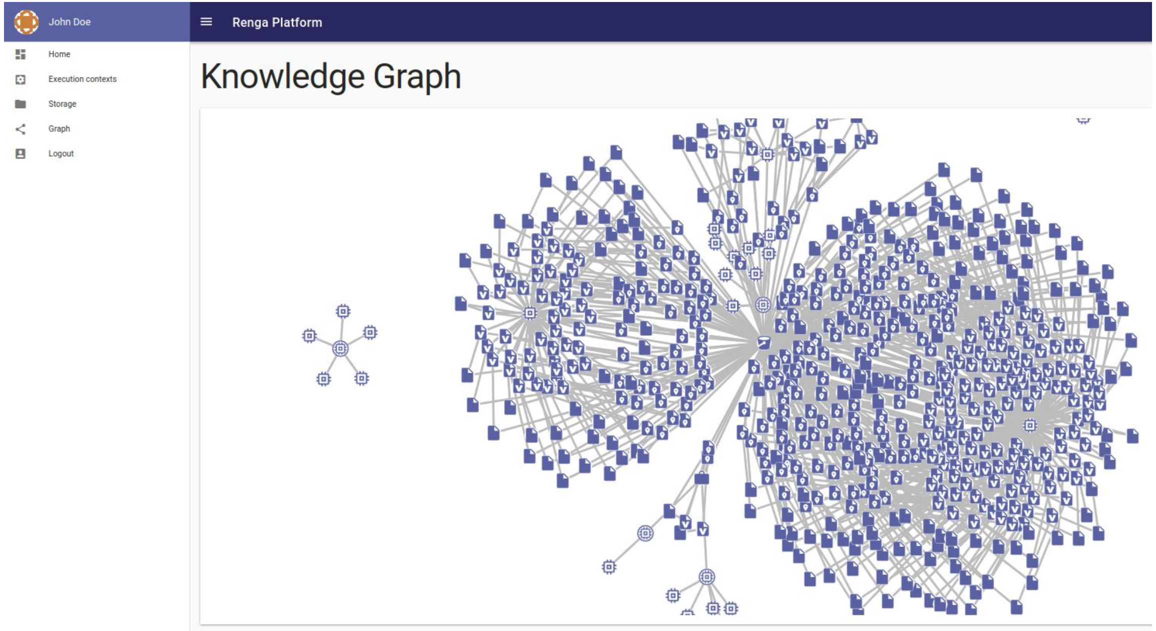
Since analysis nodes are deterministic and IO is separated from analysis, integration with Renku was trivial.

Still desired Renku features:

Constructing and executing (**Reana**?) complex workflow

Exploring, efficiently searching the graph, finding similar, related data and workflow descriptions

Simplify the pipeline and provenance information by expressing graph transformations

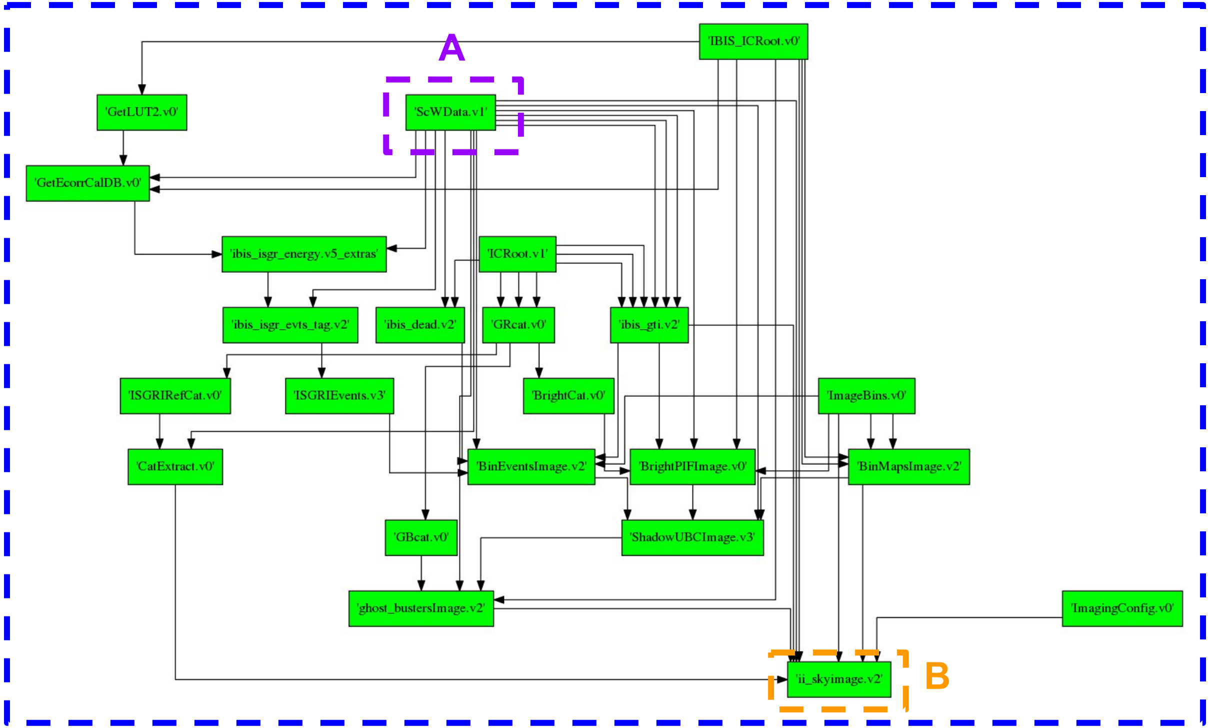


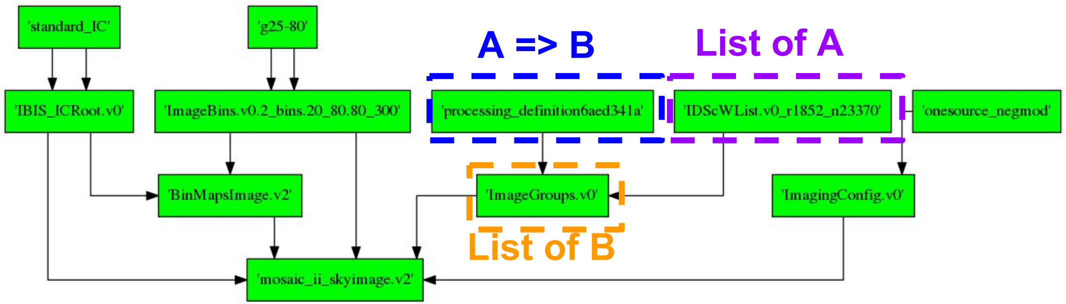
(from late 2017)

Graph simplification

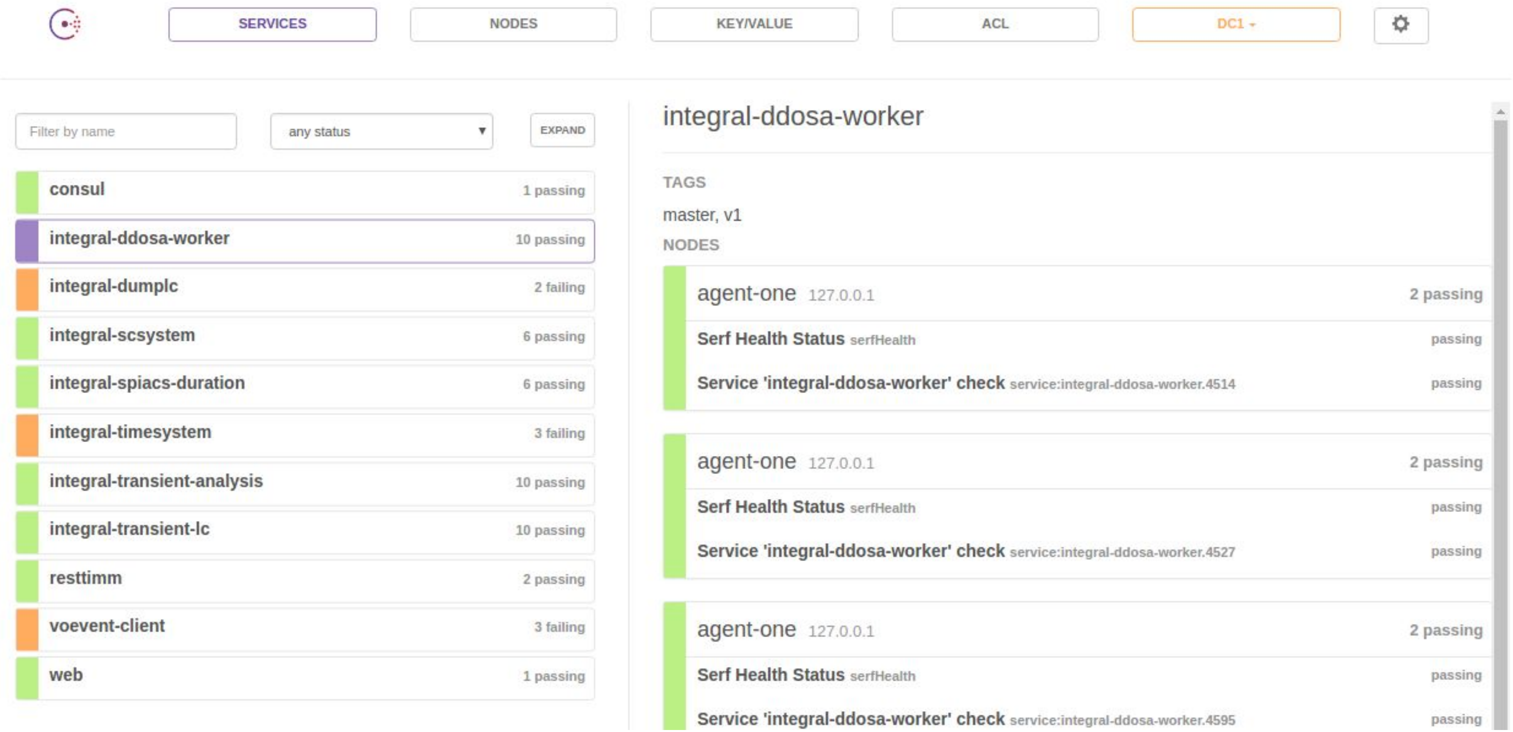
by edge contraction

Note that without this, a typical graph for minimal analysis would have ~1000 nodes, many blocks similar





Microservices: consul (hashicorp)



consul	1 passing
integral-ddosa-worker	10 passing
integral-dumplc	2 failing
integral-scsystem	6 passing
integral-spiacs-duration	6 passing
integral-timesystem	3 failing
integral-transient-analysis	10 passing
integral-transient-lc	10 passing
resttimm	2 passing
voevent-client	3 failing
web	1 passing

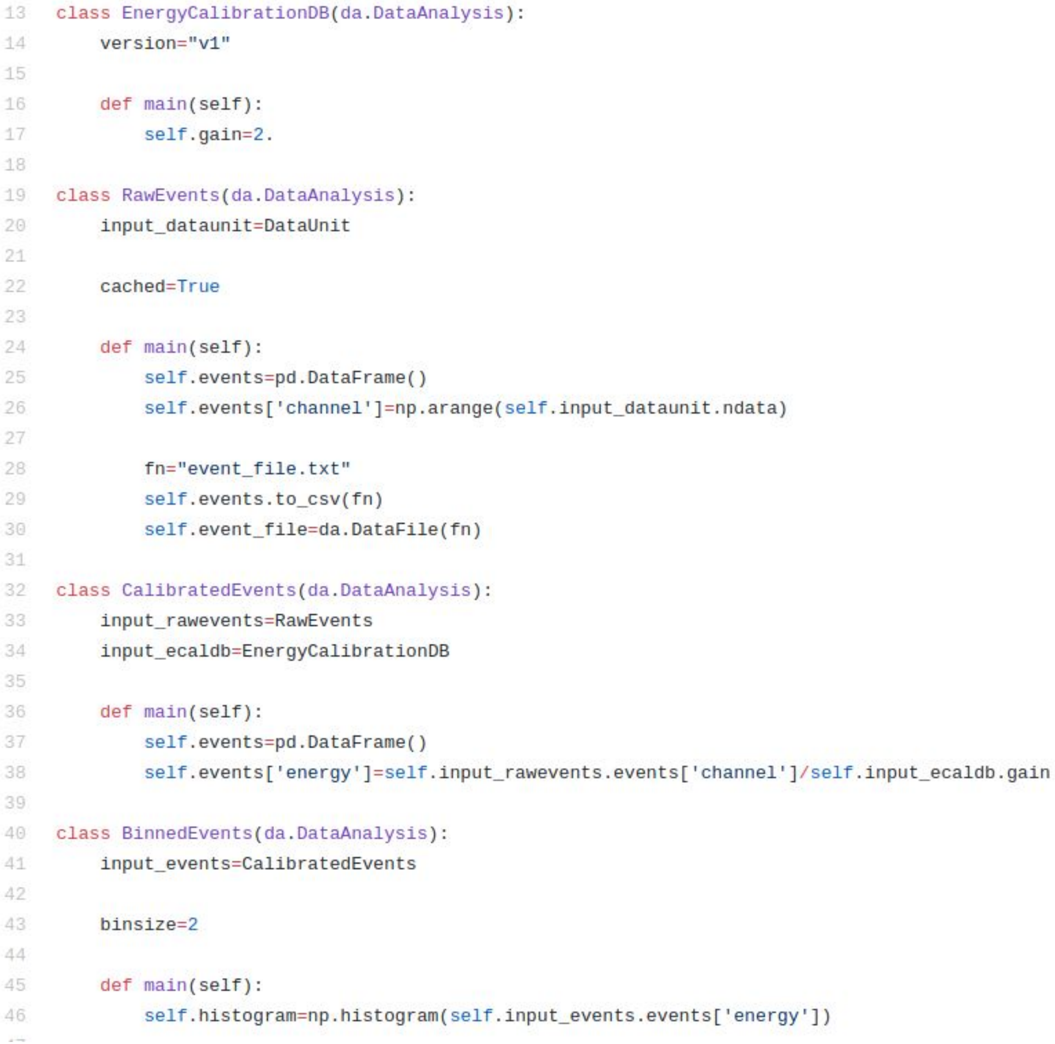
integral-ddosa-worker

TAGS

master, v1

NODES

agent-one 127.0.0.1	2 passing
Serf Health Status serfHealth	passing
Service 'integral-ddosa-worker' check service:integral-ddosa-worker.4514	passing
agent-one 127.0.0.1	2 passing
Serf Health Status serfHealth	passing
Service 'integral-ddosa-worker' check service:integral-ddosa-worker.4527	passing
agent-one 127.0.0.1	2 passing
Serf Health Status serfHealth	passing
Service 'integral-ddosa-worker' check service:integral-ddosa-worker.4595	passing



```
class EnergyCalibrationDB(da.DataAnalysis):
    version="v1"

    def main(self):
        self.gain=2.

class RawEvents(da.DataAnalysis):
    input_dataunit=DataUnit

    cached=True

    def main(self):
        self.events=pd.DataFrame()
        self.events['channel']=np.arange(self.input_dataunit.ndata)

        fn="event_file.txt"
        self.events.to_csv(fn)
        self.event_file=da.DataFile(fn)

class CalibratedEvents(da.DataAnalysis):
    input_rawevents=RawEvents
    input_ecaldb=EnergyCalibrationDB

    def main(self):
        self.events=pd.DataFrame()
        self.events['energy']=self.input_rawevents.events['channel']/self.input_ecaldb.gain

class BinnedEvents(da.DataAnalysis):
    input_events=CalibratedEvents

    binsize=2

    def main(self):
        self.histogram=np.histogram(self.input_events.events['energy'])
```