Efficient metadata management with the AMI ecosystem

CHEP, Kraków (19 - 25 Oct 2024)

F. Lambert, P.-A. Delsart, J. Fulachier, J. Odier
ami@lpsc.in2p3.fr

Outline

1. Metadata

2. The AMI ecosystem

1. Metadata

What are metadata?

“Metadata is data that provides information about other data. It describes the characteristics, content, and context of the data, making it easier to understand, organize, find, and manage.”

Metadata are essential for science

- A thought-provoking article from Nature (vol. 533, 2016):
  - >70% of researchers have failed to reproduce another scientist's experiments.
  - >50% have failed to reproduce their own experiments.
- Metadata must help make data **FAIR** for reproducible science:
  - **F**indable: The first step in (re)using data is to find them.
  - **A**ccessible: Long-term preservation and easy access to data.
  - **I**nteroperable: Open, widely shared languages and formats to combine metadata.
  - **R**eusable: Metadata must provide information about the origins of the data and the conditions for its reuse.

AMI and metadata challenges

- AMI (**A**TLAS **M**etadata **I**nterface) is a generic ecosystem dedicated to scientific metadata.
  - Over 24 years of experience within the ATLAS collaboration at CERN
  - Several years with smaller collaborations like NIKA2, n2EDM, and others
- This experience has shaped our vision of how metadata challenges should be addressed:
  - How can physicists efficiently select the data they need?
  - How to deal with heterogeneous sources of metadata?
  - How can metadata help ensure that data can be reused long after the experiment ends (in 5, 20, or even more years)?
- The primary goal of AMI is to help physicists identify the data that will be most useful to them.

2. The AMI ecosystem

AMI ecosystem in a nutshell

- **Front-end**: AWF (**A**MI **W**eb **F**ramework) - Modern JavaScript
  - Controls for building Web applications to select and display data
  - Fully configurable default "search" application
- **Back-end**: AMI Core - Java
  - Microservices providing interoperable outputs like XML, JSON, CSV, etc.
  - Interaction with any kind of datasource (auto-detection of DB structure)
- **Task Server**: A distributed super-CRON
  - Extracting metadata from primary sources (pull mode)
  - (Re)processing and storing metadata in AMI
- **Clients**: Python, C++, Java, JavaScript, etc.
- **Query Language**: MQL (**M**etadata **Q**uery **L**anguage)
  - Designed for non-database experts
  - No need for knowledge of the underlying DB schema

AMI typical usage

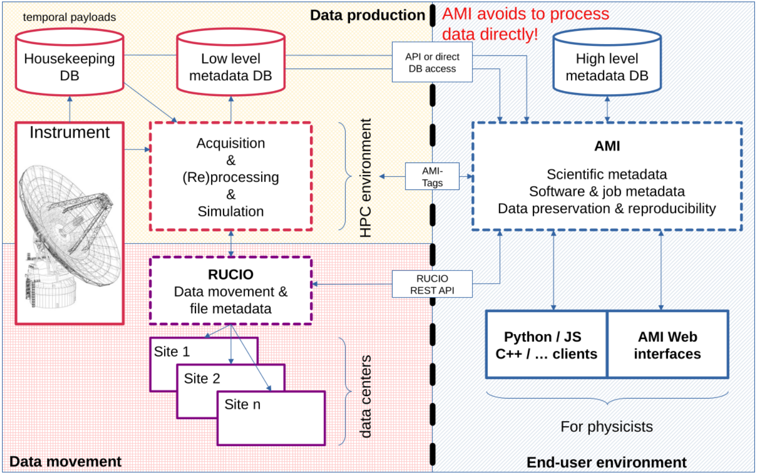


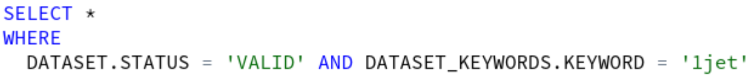
Figure 1: A typical data and metadata workflow.

End-users and AMI

- Varied profiles of end-users with diverse needs:
  - Some users need to write scripts to access data.
  - Others only require simple Web applications or command-line tools.
- Varied levels of expertise among end-users:
  - Some are scientists with development skills, while others are not.
  - Most users are not experts in SQL.
  - They often do not know the structure of the various metadata databases.
- AMI offers solutions tailored to these needs and expertise levels:
  - User-friendly Web applications (point-and-click).
  - Scriptable clients for more advanced users.
  - MQL, a high-level metadata-oriented language, designed for all users.

MQL: A Metadata-Oriented Language

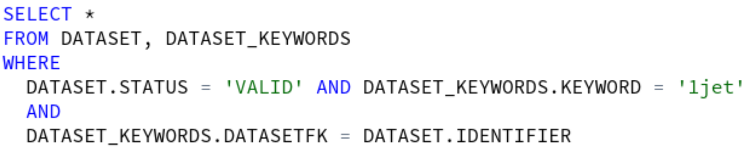
- MQL handles metadata entities; a dataset is defined by its characteristics.



```
SELECT *
WHERE
  DATASET.STATUS = 'VALID' AND DATASET_KEYWORDS.KEYWORD = '1jet'
```

Figure 2: MQL query on dataset entity.

- SQL manages database objects; a dataset is a "table" with fields.

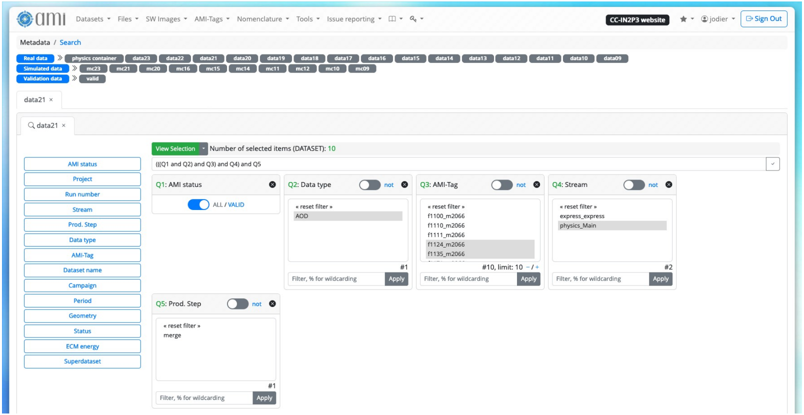


```
SELECT *
FROM DATASET, DATASET_KEYWORDS
WHERE
  DATASET.STATUS = 'VALID' AND DATASET_KEYWORDS.KEYWORD = '1jet'
  AND
  DATASET_KEYWORDS.DATASETFK = DATASET.IDENTIFIER
```

Figure 3: SQL query.

AWF: Search by Criteria Interface

Designed for point-and-click users.



Number of selected items (DATASET): 10

(((Q1 and Q2) and Q3) and Q4) and Q5

Q1: AMI status — ALL / VALID

Q2: Data type — AOD — #1

Q3: AMI-Tag — f1100_m2066, f1110_m2066, f1111_m2066, f1124_m2066, f1135_m2066 — #10, limit: 10 − / +

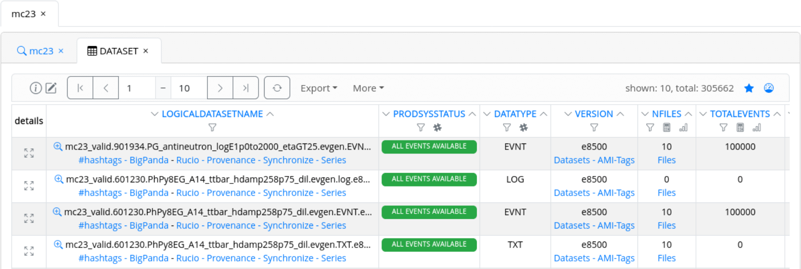
Q4: Stream — express_express, physics_Main — #2

Q5: Prod. Step — merge — #1

Figure 4: Interface for searching by criteria.

AWF: Search Result Interface

Also designed for point-and-click users.



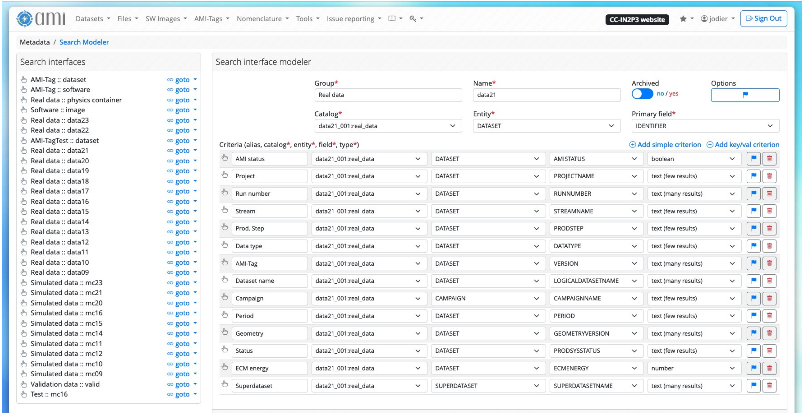
shown: 10, total: 305662

details	LOGICALDATASETNAME	PRODSYSSTATUS	DATATYPE	VERSION	NFILES	TOTALEVENTS
	mc23_valid.901934.PG_antineutron_logE1p0to2000_etaGT25.evgen.EVN... #hashtags - BigPanda - Rucio - Provenance - Synchronize - Series	ALL EVENTS AVAILABLE	EVNT	e8500 Datasets - AMI-Tags	10 Files	100000
	mc23_valid.601230.PhPy8EG_A14_ttbar_hdamp258p75_dil.evgen.log.e8... #hashtags - BigPanda - Rucio - Provenance - Synchronize - Series	ALL EVENTS AVAILABLE	LOG	e8500 Datasets - AMI-Tags	0 Files	0
	mc23_valid.601230.PhPy8EG_A14_ttbar_hdamp258p75_dil.evgen.EVNT.e... #hashtags - BigPanda - Rucio - Provenance - Synchronize - Series	ALL EVENTS AVAILABLE	EVNT	e8500 Datasets - AMI-Tags	10 Files	100000
	mc23_valid.601230.PhPy8EG_A14_ttbar_hdamp258p75_dil.evgen.TXT.e8... #hashtags - BigPanda - Rucio - Provenance - Synchronize - Series	ALL EVENTS AVAILABLE	TXT	e8500 Datasets - AMI-Tags	10 Files	0

Figure 5: Interface for displaying search results.

AWF: Search Modeler Interface

A tool for admin users to create search-by-criteria interfaces.



Metadata / Search Modeler

Search interfaces

- AMI-Tag :: dataset goto
- AMI-Tag :: software goto
- Real data :: physics container goto
- Software :: image goto
- Real data :: data23 goto
- Real data :: data22 goto
- AMI-TagTest :: dataset goto
- Real data :: data21 goto
- Real data :: data20 goto
- Real data :: data19 goto
- Real data :: data18 goto
- Real data :: data17 goto
- Real data :: data16 goto
- Real data :: data15 goto
- Real data :: data14 goto
- Real data :: data13 goto
- Real data :: data12 goto
- Real data :: data11 goto
- Real data :: data10 goto
- Real data :: data09 goto
- Simulated data :: mc23 goto
- Simulated data :: mc21 goto
- Simulated data :: mc20 goto
- Simulated data :: mc16 goto
- Simulated data :: mc15 goto
- Simulated data :: mc14 goto
- Simulated data :: mc11 goto
- Simulated data :: mc12 goto
- Simulated data :: mc10 goto
- Simulated data :: mc09 goto
- Validation data :: valid goto
- ~~Test :: mc16~~ goto

Search interface modeler

Group*: Real data

Name*: data21

Archived: no / yes

Options

Catalog*: data21_001:real_data

Entity*: DATASET

Primary field*: IDENTIFIER

Criteria (alias, catalog*, entity*, field*, type*)

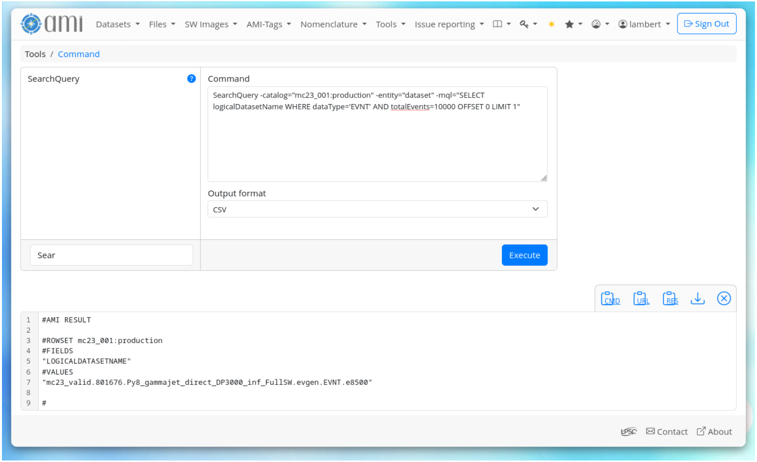
⊕ Add simple criterion ⊕ Add key/val criterion

AMI status	data21_001:real_data	DATASET	AMISTATUS	boolean
Project	data21_001:real_data	DATASET	PROJECTNAME	text (few results)
Run number	data21_001:real_data	DATASET	RUNNUMBER	text (many results)
Stream	data21_001:real_data	DATASET	STREAMNAME	text (few results)
Prod. Step	data21_001:real_data	DATASET	PRODSTEP	text (few results)
Data type	data21_001:real_data	DATASET	DATATYPE	text (few results)
AMI-Tag	data21_001:real_data	DATASET	VERSION	text (many results)
Dataset name	data21_001:real_data	DATASET	LOGICALDATASETNAME	text (many results)
Campaign	data21_001:real_data	CAMPAIGN	CAMPAIGNNAME	text (few results)
Period	data21_001:real_data	DATASET	PERIOD	text (few results)
Geometry	data21_001:real_data	DATASET	GEOMETRYVERSION	text (many results)
Status	data21_001:real_data	DATASET	PRODSYSSTATUS	text (few results)
ECM energy	data21_001:real_data	DATASET	ECMENERGY	number
Superdataset	data21_001:real_data	SUPERDATASET	SUPERDATASETNAME	text (many results)

Figure 6: Search Modeler interface (admin user).

Microservices: Interact from the Web

A tool for advanced users to interact using defined "commands".



Command:

```
SearchQuery -catalog="mc23_001:production" -entity="dataset" -mql="SELECT
logicalDatasetName WHERE dataType='EVNT' AND totalEvents=10000 OFFSET 0 LIMIT 1"
```

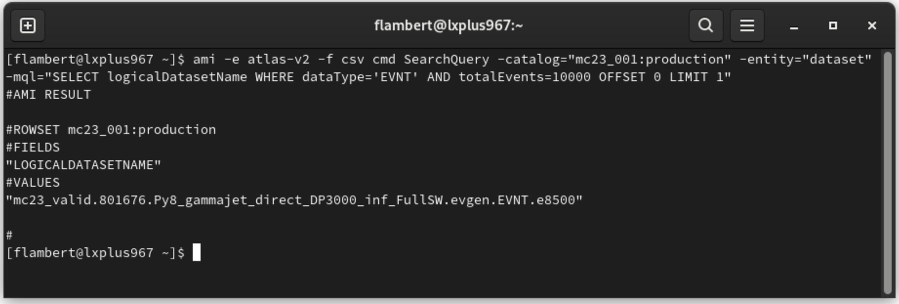
Output format: CSV

```
#AMI RESULT

#ROWSET mc23_001:production
#FIELDS
"LOGICALDATASETNAME"
#VALUES
"mc23_valid.801676.Py8_gammajet_direct_DP3000_inf_FullSW.evgen.EVNT.e8500"

#
```

Figure 7: MQL query "command" executed from a Web application.

Microservices: Interact from a Shell

Advanced users can interact using defined "commands" in a shell.

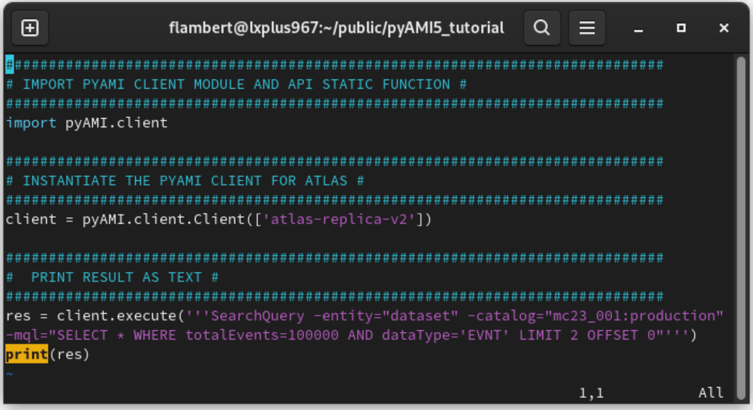


```
[flambert@lxplus967 ~]$ ami -e atlas-v2 -f csv cmd SearchQuery -catalog="mc23_001:production" -entity="dataset"
-mql="SELECT logicalDatasetName WHERE dataType='EVNT' AND totalEvents=10000 OFFSET 0 LIMIT 1"
#AMI RESULT

#ROWSET mc23_001:production
#FIELDS
"LOGICALDATASETNAME"
#VALUES
"mc23_valid.801676.Py8_gammajet_direct_DP3000_inf_FullSW.evgen.EVNT.e8500"

#
[flambert@lxplus967 ~]$ 
```

Figure 8: MQL query "command" executed from a shell.

Microservices: Interact with Python

Programmers can interact using an existing client in a program.

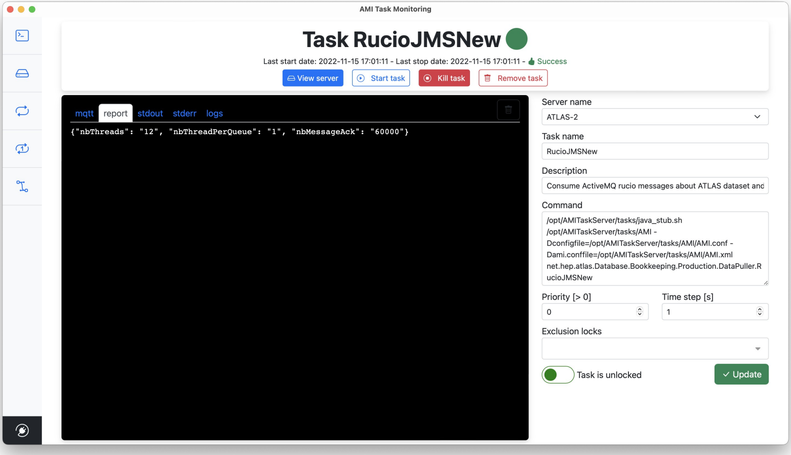


```
################################################################################
# IMPORT PYAMI CLIENT MODULE AND API STATIC FUNCTION #
################################################################################
import pyAMI.client

################################################################################
# INSTANTIATE THE PYAMI CLIENT FOR ATLAS #
################################################################################
client = pyAMI.client.Client(['atlas-replica-v2'])

################################################################################
#  PRINT RESULT AS TEXT #
################################################################################
res = client.execute('''SearchQuery -entity="dataset" -catalog="mc23_001:production"
-mql="SELECT * WHERE totalEvents=100000 AND dataType='EVNT' LIMIT 2 OFFSET 0"''')
print(res)
~
```

Figure 9: MQL query "command" executed with a Python script.

Task Server: Run any kind of task

The metadata manager can define tasks to aggregate metadata...



Task RucioJMSNew

Last start date: 2022-11-15 17:01:11 - Last stop date: 2022-11-15 17:01:11 - Success

```
{"nbThreads": "12", "nbThreadPerQueue": "1", "nbMessageAck": "60000"}
```

Server name: ATLAS-2

Task name: RucioJMSNew

Description: Consume ActiveMQ rucio messages about ATLAS dataset and

Command:

```
/opt/AMITaskServer/tasks/java_stub.sh
/opt/AMITaskServer/tasks/AMI -
Dconfigfile=/opt/AMITaskServer/tasks/AMI/AMI.conf -
Dami.conffile=/opt/AMITaskServer/tasks/AMI/AMI.xml
net.hep.atlas.Database.Bookkeeping.Production.DataPuller.R
ucioJMSNew
```

Priority [> 0]: 0

Time step [s]: 1

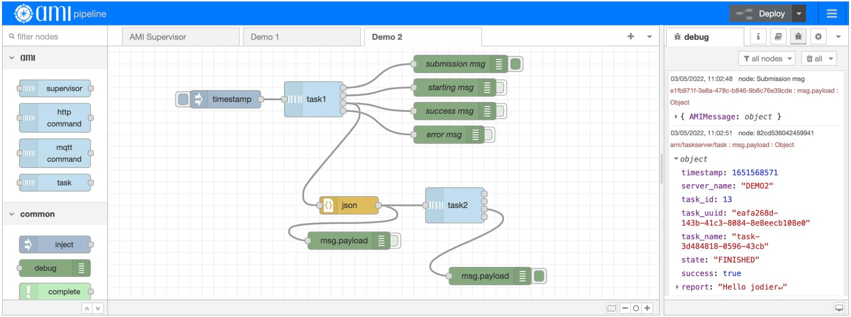
Exclusion locks:

Task is unlocked

Figure 10: Configuration of recurrent tasks.

Task server: pipelined tasks

... and configure pipelined tasks.



```
03/05/2022, 11:02:48   node: Submission msg
e1fb971f-3e8a-478c-b846-9b6c76e39cde : msg.payload :
Object
 ▸{ AMIMessage: object }
03/05/2022, 11:02:51   node: 82cd536042459941
ami/taskserver/task : msg.payload : Object
 ▾object
   timestamp: 1651568571
   server_name: "DEMO2"
   task_id: 13
   task_uuid: "eafa268d-
   143b-41c3-8084-8e8eecb108e0"
   task_name: "task-
   3d484818-0596-43cb"
   state: "FINISHED"
   success: true
 ▸ report: "Hello jodier↵"
```

Figure 11: Chained task executions.

Try AMI

- Official site
  - https://ami-ecosystem.in2p3.fr/
- Docker-compose-based demo
  - Test on your laptop: https://github.com/ami-team/AMIDemo/
  - Test online: http://demo.ami-ecosystem.in2p3.fr:667/
- Documentation (Admin/end-user/developer guides, MQL langage)
  - https://ami-ecosystem.in2p3.fr/doc/
- Contact
  - ami@lpsc.in2p3.fr

Thank You for Your Attention!