The ICAT Data Server (IDS): interface, implementation and experience

Frazer Barnsley,
Steve Fisher <dr.s.m.fisher@gmail.com>,
Wojciech Grajewski and
Antony Wilson

Rutherford Appleton Laboratory - STFC

NOBUGS 24-26 September 2014

Overview

- How the IDS fits into the ICAT Project (http://icatproject.org) or doi:10.5286/SOFTWARE/ICAT
- Interface
- Design
- Calls
- Experience from three deployments
- Summary of experience

ICAT and IDS

- The metadata catalogue of ICAT provides a SOAP web service interface to metadata
- Designed to support scientific facilities
- Schema includes all you might need from proposal to publication
- IDS provides a “RESTful” interface to the data files cataloged by ICAT
- Complements ICAT

ICAT - Metadata

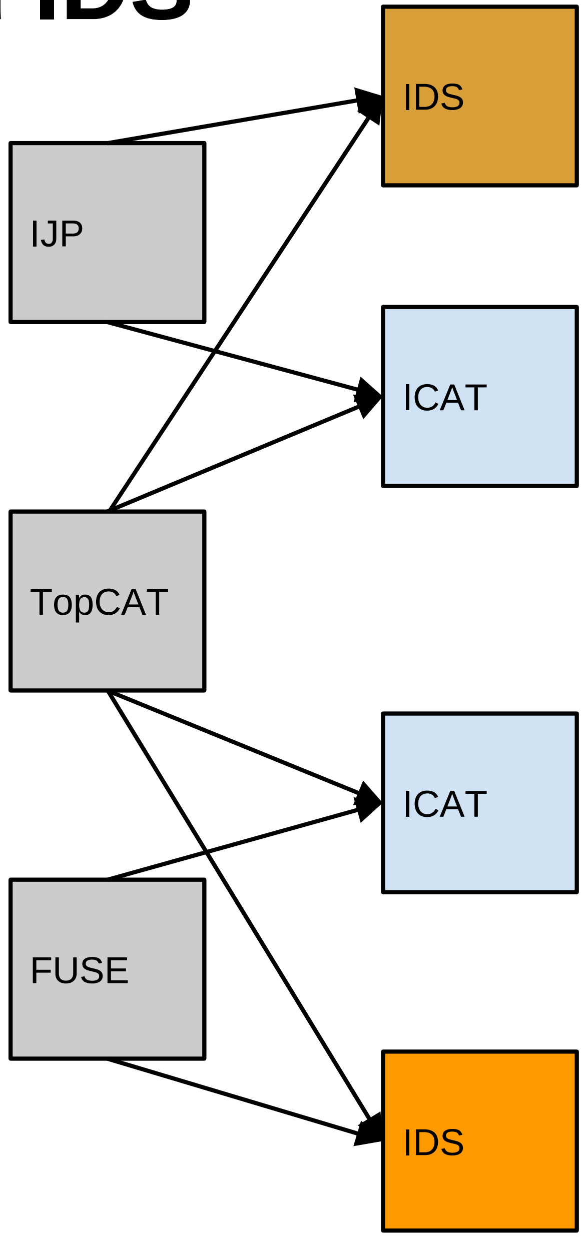
IDS - Data

Building on ICAT and IDS

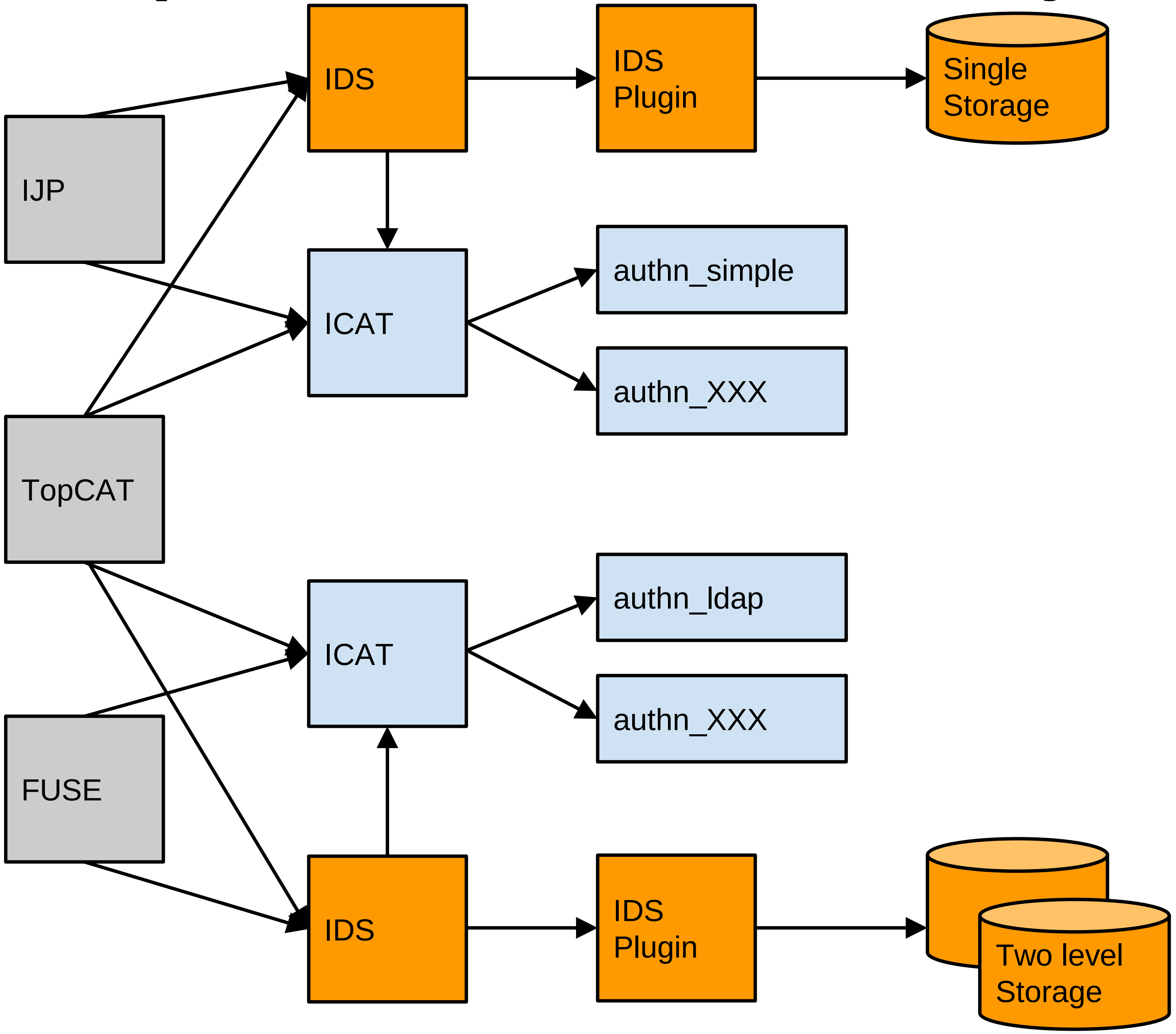
IJP - a job portal able to submit jobs operating on ICAT catalogued data
TopCAT - provides view of ICAT catalogued data from multiple facilities
FUSE - prototype showing ICAT catalogued data.

All need uniform access to:

- metadata - ICAT
- data - IDS

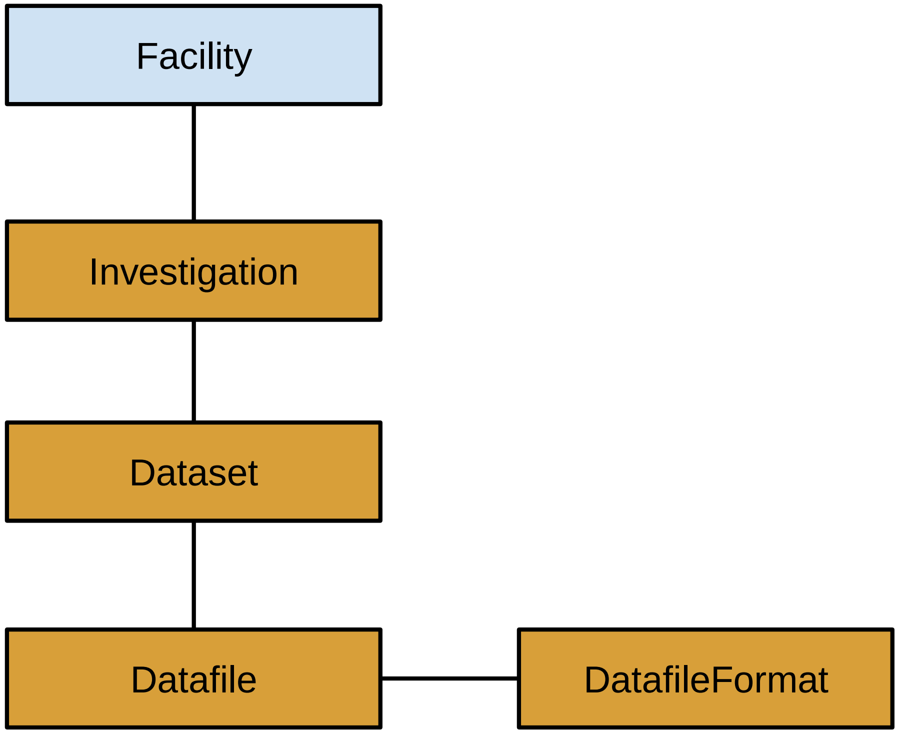


Components of the ICAT Project



ICAT Catalogue

- The metadata catalogue of ICAT provides a SOAP web service interface to an underlying database with an easy to use API.
- Authentication makes use of plugins
- Authorization is rule based.
- Around forty different entity types in the ICAT schema
- Four are of interest to the IDS.



The IDS Interface

- When a file is uploaded metadata are stored in ICAT.
- ICAT authorization rules for the datafile metadata applied to control read/write access to IDS files.
- Multiple downloaded files are zipped.
- The interface has archive and restore calls that suggest two level storage. Either:
    - all data available on “archive storage” with recently used data cached on “main storage”

  or

    - all data in “main storage”

Interface Design

- Avoided SOAP to transfer large files efficiently
- REST-like but with @Path as verbs rather than nouns such as archive or getSize.
- @Produces text/plain, application/octet-stream or application/json
- @Consumes multipart/form-data or application/octet-stream
- Many calls can operate in a uniform manner on sets of datafiles, datasets and investigations

The Calls

- put
- getStatus
- getSize
- getData
- getLink

- prepareData
- isPrepared
- getData

- delete

- ping
- getServiceStatus
- isReadOnly
- isTwoLevel

- archive
- restore

An implementation with plugins

A plugin can implement three interfaces

- main storage interface
    - decides how to store individual data files
- archive storage interface
    - decides how to store zipped dataset of datafiles
        - this assumes that files of a dataset are often wanted together.
        - zipping potentially saves space
- zip file structure interface
    - defines the structure of the zip file contents

Zip files are generated on the fly

- no delay in starting to deliver requested data provided it is in main storage.

LSF - plugin and experience

- The Lasers for Science Facility has been using a two level file storage plugin.
  - Main storage is on normal disk
  - Archive storage is on HSM file system (DMF)
- This is a new prototype deployment so we were free to define our own storage structure.
- No problems encountered but it has not had much use yet.

ISIS - plugin and experience

- ISIS already have a lot of data stored
  - avoid directory structure change
- They have sufficient disk to hold all data on line
  - They only need main storage
- Experiment data files are
  - written by the ISIS software and catalogued with ICAT
  - IDS not involved in writing of experimental data files
  - Users download via the IDS (normally from TopCAT)
- Derived data
  - users can also use the IDS for both upload and download
- They appreciate the speed of the implementation.

DLS - plugin and experience

- They already have a very large amount of data which is stored on tape.
- Like ISIS they do not store data via the IDS
- Wrote a plugin to
  - use the existing tape system as archive storage
  - use a large disk as main storage

- The main difficulty is that archive storage plugin is expected to deliver zipped datasets
  - The datafiles are not stored that way.
  - This requires accessing all the datafiles of a dataset and zipping them up.
- Enhancement planned to allow archive storage to be organised by datafile or by dataset.

Summary of experience

- The basic idea works
  - facility independent interface to read and write datafiles using ICAT catalog and ICAT authz rules
- The IDS needs to be able to work with very different pre-existing data storage structures
  - facilities don’t want to shuffle huge volumes of data to adopt an ICAT style data hierarchy
- Facilities want their own ZIP file layout