ICAT Workshop: Introduction

Brian Matthews
and Steve Fisher

Scientific Computing Department
Overview of workshop

• A look at the ICAT facilities Information CATalogue
  - Relatively straightforward to install
  - Much harder to set up so that the full value can be gained

• An in-depth look at the features of ICAT 4.2
• How to configure it to work in a particular environment
• Add on tools around ICAT
• What is planned for future developments
## Contents

<table>
<thead>
<tr>
<th>Time</th>
<th>Topic</th>
<th>Presenter</th>
</tr>
</thead>
<tbody>
<tr>
<td>13:00</td>
<td>Introduction to ICAT 4.2</td>
<td>Brian Matthews</td>
</tr>
<tr>
<td>13:30</td>
<td>Getting started - how to set up the common data tables</td>
<td>Kevin Phipps</td>
</tr>
<tr>
<td>14:00</td>
<td>Using the API to ingest metadata</td>
<td>Tom Griffin</td>
</tr>
<tr>
<td>14:30</td>
<td>The pluggable authentication system</td>
<td>Tom Griffin</td>
</tr>
<tr>
<td>14:40</td>
<td>The permissions/authorisation rules</td>
<td>Tom Griffin</td>
</tr>
<tr>
<td>15:00</td>
<td>Coffee break</td>
<td></td>
</tr>
<tr>
<td>15:15</td>
<td>Look at the notifications system</td>
<td>Antony Wilson</td>
</tr>
<tr>
<td>15:30</td>
<td>TopCAT configuration</td>
<td>Antony Wilson</td>
</tr>
<tr>
<td>16:00</td>
<td>Data download</td>
<td>Kevin Phipps</td>
</tr>
<tr>
<td>16:15</td>
<td>Discussion, requests and promises for 4.3 and beyond.</td>
<td>Brian Matthews</td>
</tr>
</tbody>
</table>
Assumptions

• You have some familiarity with the aims of ICAT
• You want to know how it is practically used in a facility
• You know a little about its architecture
• You have some knowledge of programming and setting up systems
A Word from our Sponsors

• ICAT team from STFC
  – ISIS Neutron Source
  – Scientific Computing Department
• Also support from the PaN-Data Open Data Infrastructure project
  – WP4: Data Catalogues
  – Promoting the use of data catalogues within European facilities
  – Using ICAT as the reference catalogue
What is ICAT?

• A metadata catalogue to manage data
  – Database (any JPA supported e.g. Oracle, MySql)
  – Web Service API (developed in Java, running Glassfish)

• To allow high-quality
  – Registration of data as it is collected in experiments
  – Discovery of data based on what it was collected for
  – Access to the data, according to some policy
  – Association of data with other resources

• Moving towards the ICAT Toolkit
  – Pluggable component architecture
  – Supporting different tasks
ICAT Vision

What is ICAT?

ICAT is a database (with a well-defined API) that provides a uniform interface to experimental data and a mechanism to link all aspects of research from proposal through to publication.

- Access data anywhere via the web
- Annotate your data
- Search for data in a meaningful way e.g. taxonomy, Sample, temperature, pressure etc
- Share data with colleagues
- Access data via your own programs (C++, Fortran, Java etc.) via the ICAT API
- Identify potential collaborations
- Utilise integrated e-Science High-Performance Computing and Visualisation resources
- Link to data from your publications
- Etc.

Proposals

Once awarded beamtime at ISIS, an entry will be created in ICAT that describes your proposed experiment.

Experiment

Data collected from your experiment will be indexed by ICAT (with additional experimental conditions) and made available to your experimental team.

Analysed Data

You will have the capability to upload any desired analysed data and associate it with your experiments.

Publication

Using ICAT you will also be able to associate publications to your experiment and even reference data from your publications.
History

- 2001: First rapid prototype (“Data Portal”)
- 2005: ICAT 2.0 – ISIS back catalogue
- 2008: ICAT 3.3 - production version released open source (Oracle, EJB, Glassfish)
- 2009: deployed in ISIS, DLS
- 2009-12: ESRF, ORNL, ILL, PSI, CLF, LSF, ELETTRA ...
- 2011: ICAT 4.0 – revisit the API
- 2012: ICAT 4.2 – pluggable authentication

http://www.icatproject.org/
http://code.google.com/p/icatproject/
# Release summary

<table>
<thead>
<tr>
<th>Release</th>
<th>Comments</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.x.x</td>
<td>Big API - many variants</td>
<td>A number of deployments</td>
</tr>
<tr>
<td>4.0</td>
<td>Small API - Never intended to be deployed for production use.</td>
<td>Should no longer be used</td>
</tr>
<tr>
<td>4.1</td>
<td>Intended for production use for new users.</td>
<td>Released June 2012</td>
</tr>
<tr>
<td>4.2</td>
<td>Pluggable authentication</td>
<td>Released August 2012</td>
</tr>
<tr>
<td>4.3</td>
<td>Under discussion</td>
<td>Planned for next 6 months</td>
</tr>
</tbody>
</table>
ICAT Workshop: ICAT 4.2: Architecture and Schema Overview

Brian Matthews
and Steve Fisher

Scientific Computing Department
• All objects have representation in RDBMS on the server and in the client.
  – Any EclipseLink supported RDMS
  – 4.1 Tested with Oracle, MySQL and Derby

• With a information model schema tailored to represent facilities data collection

• Schema can be adjusted very easily
  – only "special" objects are: User, Group, UserGroup, Rule, NotificationMessage
  – objects with special behaviour: Rule, NotificationMessage and Parameter

• A facility could easily add concepts such as Proposal, Run etc
  – should be coordinated by the ICAT project
  – otherwise working across facilities becomes difficult.
Schema - main components

Many objects take arbitrary parameters as an extensibility mechanism
- type identified by a:
  (name, value, unit, facility)

Also user management components

http://www.icatproject.org/mvn/site/icat/4.2.0/icat.core/schema.html
Investigation

An investigation or experiment

Uniqueness constraint name, visitId, facilityCycle, instrument.

Relationships

<table>
<thead>
<tr>
<th>Card</th>
<th>Class</th>
<th>Field</th>
<th>Cascaded</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0,1</td>
<td>Instrument</td>
<td>instrument</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1,1</td>
<td>Facility</td>
<td>facility</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0,1</td>
<td>FacilityCycle</td>
<td>facilityCycle</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0,*</td>
<td>Publication</td>
<td>publications</td>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td>0,*</td>
<td>Shift</td>
<td>shifts</td>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td>0,*</td>
<td>Sample</td>
<td>samples</td>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td>0,*</td>
<td>InvestigationUser</td>
<td>investigationUsers</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>1,1</td>
<td>InvestigationType</td>
<td>type</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0,*</td>
<td>StudyInvestigation</td>
<td>studyInvestigations</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>0,*</td>
<td>InvestigationParameter</td>
<td>parameters</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>0,*</td>
<td>Keyword</td>
<td>keywords</td>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td>0,*</td>
<td>Dataset</td>
<td>datasets</td>
<td></td>
<td>Yes</td>
</tr>
</tbody>
</table>

Other fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>visitId</td>
<td>String [255]</td>
<td>Identifier for the visit to which this investigation is related</td>
</tr>
<tr>
<td>endDate</td>
<td>Date</td>
<td></td>
</tr>
<tr>
<td>summary</td>
<td>String [4000]</td>
<td>Summary or abstract</td>
</tr>
<tr>
<td>startDate</td>
<td>Date</td>
<td></td>
</tr>
<tr>
<td>doi</td>
<td>String [255]</td>
<td>The Digital Object Identifier associated with this investigation</td>
</tr>
<tr>
<td>name</td>
<td>String [255]</td>
<td>A short name for the investigation</td>
</tr>
<tr>
<td>title</td>
<td>String [255]</td>
<td>Full title of the investigation</td>
</tr>
<tr>
<td>releaseDate</td>
<td>Date</td>
<td>When the data will be made freely available</td>
</tr>
</tbody>
</table>
Schema changes (4.2)

- All entities have an id value as primary key
- Remove attributes that are not of general utility
  - use xxxParameters instead
- Those entities which represent many to many relationships now formed by concatenating the names of the related entities
  - “Investigator” has now become "InvestigationUser"
- All relationships may now be navigated in both directions
- Relationship fields follow normal convention
- Harmonised names of the various string fields
- Reduce number of non-optional fields
- Added notification request
- Added job etc. for provenance
More Schema (4.2) changes

- Permit mixing data from more than one facility
  - Particularly interesting for an institutional or personal ICAT
  - Requires that a number of unique constraints would need to include the facility. This would affect the various “Type” and “Status” entities.

- Add restrictions to ParameterTypes to define:
  - possible values for strings
  - upper and lower limits for numbers
  - boolean to indicate whether or not ICAT should respect the constraint

- Get rid of Topic
  - Still have keyword

- Add DOI support
  - Publication, Investigation, Dataset, Datafile
Simple representation of provenance

Application

Job

InputDataset

OutputDataset

InputDatafile

OutputDatafile

Dataset

Datafile
ICAT 4.x: The new API

- A much simplified and consistent API
- No Schema Specific operations
  - Easier to change the schema without side effects
- A small set of core operations
  - Which can be applied to all entities.

login
  returns a sessionId to identify you for a finite period

Create
  for the various Schema Entities

Read
  actually search and get

Update
Delete
Getting started

• Initialise and configure ICAT
• Set the credentials to log in to ICAT
• Start the Session

        String sessionId = icat.login("db", credentials);

• Start creating entities in the Data Model

        Investigation inv = new Investigation();
        inv.setName("Fred");
        inv.setFacility(f);

More on these topics later in the day.
get ...

Call to retrieve the object for a particular entity:

- Parameterised by the ENTITY Name
- This is now a general pattern in ICAT 4.x

This is useful if you already know the ID of the entity from when it was created or as a result of a search:

```java
String pk = "Some facility";
Facility f = (Facility) icat.get(sessionId, "Facility", pk);

Long dsid = 76347;
Dataset ds = (Dataset) icat.get(sessionId, "Dataset", dsid);
```
The INCLUDE keyword

- Get (and Search - later) only retrieve the entity you request with no related entities
- To get a tree of related entities list them with INCLUDE:

```java
Dataset ds = (Dataset) icat.get(sessionId, "Dataset INCLUDE Datafile,DatasetParameter,DatafileParameter", dsid);
```

- Can then do:

```java
for (Datafile datafile : dataset.getDatafiles()) {
    System.out.println(datafile.getId() + " " +
                      datafile.getParameters().size());
}
```

- but dataset.getType() will always be null as DatasetType not included

Included entities must be linked via a graph without loops - i.e. only one route.
update ...

- First obtain the object that you wish to update using search or get.
- It is important to obtain it using "INCLUDE 1" where the "1" means to get all many to one relationships
- Listing individual related entities is error prone and may not work if the schema is modified.

```java
Facility f = (Facility) icat.get
    (sessionId, "Facility INCLUDE 1", facilityId);

f.setDaysUntilRelease(30);
icat.update(sessionId, f);
```
and delete ...

This call takes an entity object but only the value of its id actually matters so the following are equivalent:

```java
Facility f = (Facility)
    icat.get(sessionId, "Facility", facilityId);
icat.delete(sessionId, f);

Facility f = new Facility();
f.setId(facilityId);
icat.delete(sessionId, f);
```

- Again, "cascades" are followed so this delete would have a major effect.

In all cases cascaded relationships are followed - currently all 0-many and 1-many relationships are cascaded but no others. Check the current schema to be sure. For example: http://www.icatproject.org/mvn/site/icat/4.2.0/icat.core/schema.html
A Domain Language for Searching

We have added a language based on the ICAT model to search

String query = "Dataset"
List<?> results = icat.search(sessionId, query);

query = "Dataset.name";

query = "DISTINCT Dataset.name"

query = "Dataset.id ORDER BY id";

query = "MAX (Dataset.id) ";

query = "3,5 Dataset.id ORDER BY id";

query = "3, Dataset.id ORDER BY id";

query = " ,5 Dataset.id ORDER BY id";
Search restrictions

"Dataset [type.name = 'GS' OR type.name = 'GQ']"

"Dataset [type.name IN ('GS', 'GQ')]"

"Dataset <-> DatasetParameter[type.name = 'TEMP' AND numericValue > 300]"

"Dataset <-> DatasetParameter[(type.name = 'TEMP' AND numericValue > 300) AND (type.name = 'PRESSURE' AND numericValue > 1020)]"

"Dataset [type.name IN ('GS', 'GQ')] INCLUDE Datafile, DatasetParameter, DatafileParameter"

Q. Why the specialized query language?

A. To support authz

The restriction in the square brackets can be as complex as required - but must only refer to attributes of the object being restricted.
ICAT 4.2 includes

• Schema changes
  – Made more consistent
  – Adding support for Provenance

• Major change to API
  – CRUD calls consistent across all entities
  – Search Language

• Also some other changes
  – Security model
  – Notifications
  – Error messages

• Covered in later talks