
# ICAT Notification System

Antony Wilson

ICAT Workshop

Scientific Computing Department, RAL

27th September 2012

# Overview

- JMS
- Notification Request
- Notification Request Parameters
- Examples
- Summary

# Notification

- ICAT can send out notifications based on selection criteria
  - When a data set is updated
  - When a data file is read
- The notification system is intended for use by facility admin staff

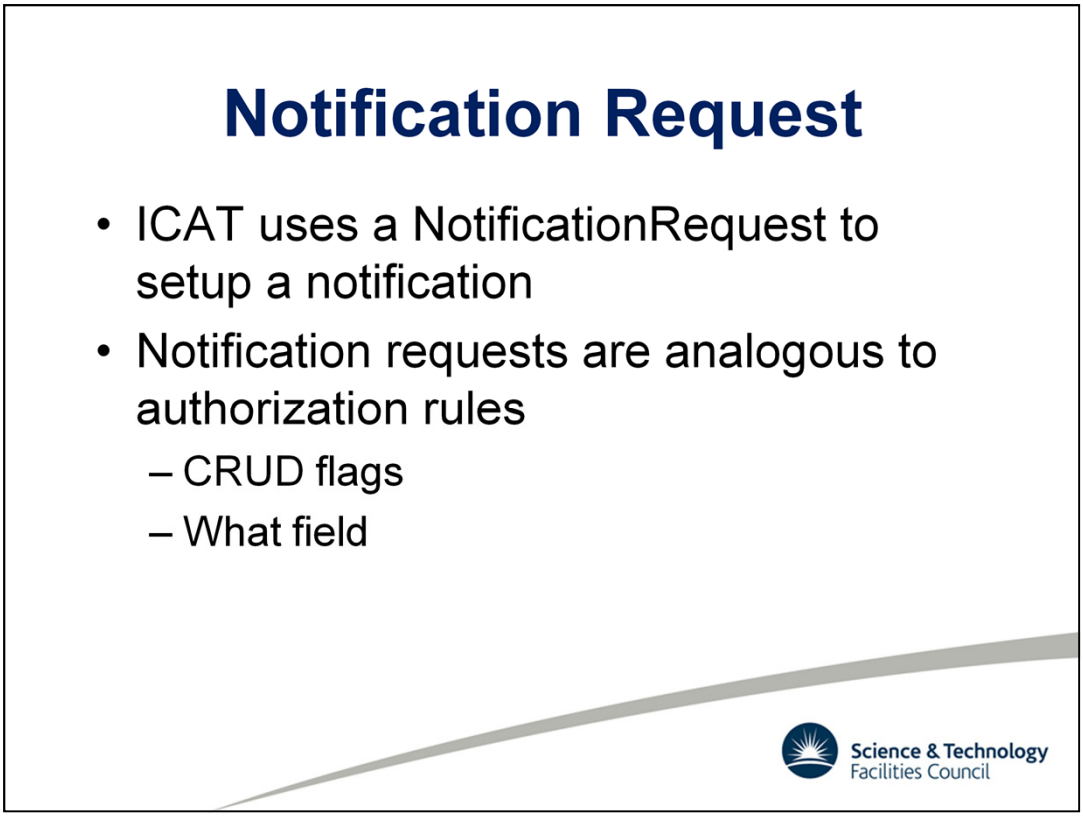
Science & Technology Facilities Council

# Java Message Service

- Java Message Service (JMS) is used to send messages/notifications
  - Publish subscribe
    - May be read by multiple consumers
  - Point to point
    - Only available to one consumer
- Makes use of the broker in the glassfish server running ICAT

Science & Technology Facilities Council

The message is sent as an ObjectMessage - but without an Object being attached. All the information is sent as properties with the same name as requested in the dataTypes.




ICAT is able to send JMS messages. To do this, the table "NotificationRequest" should be populated as needed. This table includes two columns: crudFlags and what which are treated in a similar way to columns of the same name in the authorization rules to choose the conditions for sending a message. However if the what field is more than just the name of the entity the request will not be honoured for search calls.

# Notification Request Parameters

- A NotificationRequest is configured using
  - CRUD flags
  - What
  - DataTypes
  - DestType
  - Name

Science & Technology Facilities Council

# CRUD Flags Parameter

- Define which operation(s) will trigger a notification
  - Create
  - Read
    - Get
    - Search
  - Update
  - Delete

Science & Technology Facilities Council

# What Parameter

- What is used to trigger a notification
- May contain an entity name or entity name and condition

```
Dataset <-> Investigation [name = "Fred"]
```

- If there is a condition the request will not be honoured for search calls

Science & Technology Facilities Council

# Data Types Parameter

- Defines what data to include in the notification
- notificationName
  - the name as provided in the name field of the request
- userId
  - the name of the authenticated user performing the operation resulting in this notification

Science & Technology Facilities Council

# Data Types Parameter

- entityName
  - the name of the main entity being referenced
    - This excludes any INCLUDE fields from a search or get call and also excludes entities besides the top one for create calls
- entityId
  - the id (primary key) of the main entity

Science & Technology Facilities Council

# Data Types Parameter

- query
  - the query string for a search call

Science & Technology Facilities Council

# Dest Type Parameter

- DestType.PUBSUB
  - Publish subscribe
  - Message published as a JMS topic
  - May be read by multiple consumers
- DestType.P_2_P
  - Point to point
  - Message put on a queue
  - Removed after it is consumed by first consumer

Science & Technology Facilities Council

The final field is the destType which must be either DestType.PUBSUB or DestType.P_2_P. The first case publishes the message as a JMS topic where it may be read by multiple consumers and the second puts it onto a queue where it will be consumed by the first consumer to take it.

# Name Parameter

- The name identifies the notification request
- MUST be unique

# Example 1

```
NotificationRequest notificationRequest = new
   NotificationRequest();
notificationRequest.setCrudFlags("C");
notificationRequest.setWhat("Datafile");
notificationRequest.setDatatypes("notificationName
   userId entityName entityId");
notificationRequest.setDestType(DestType.PUBSUB);
notificationRequest.setName("Test");
icat.create(sessionId, notificationRequest);
```


This will send a notification message containing the name of the notification ("Test" in this case), the userId, the entityName (which will always be "Datafile" in this case) and its id for every call where a Datafile is created. "Publish/Subscribe" mode will be used rather than "Point-to-Point". Note that the id field of the notificationRequest on the client side is not set on the assumption that the client side copy of the notificationRequest will not be needed further

# Example 2

```
NotificationRequest notificationRequest = new
   NotificationRequest();
notificationRequest.setCrudFlags("R");
notificationRequest.setWhat("Dataset <-> Investigation
   [name = 'Fred']");
notificationRequest.setDatatypes("userId entityId");
notificationRequest.setDestType(DestType.PUBSUB);
notificationRequest.setName("Fred readers");
icat.create(sessionId, notificationRequest);
```

Science & Technology Facilities Council

To see who is reading which datasets (with get calls) belonging to some investigation called "Fred" one could have the above example.

# Security

- Notifications allow publication of data from ICAT
- Only publish information that is required
- It is recommended that access to the NotificationRequest table is limited to members of the `FacilityAdmins` group

Science & Technology Facilities Council

Though the notification mechanism is powerful, it does allow information to be published from ICAT which is not protected by the normal ICAT authorization mechanism. Please publish only the information required to meet your needs and take care that the authorization for the NotificationRequest table is well controlled.

# Example Rule

```
Group facilityAdmins = (Group)port.search(sessionId,
   "Group[name='FacilityAdmins']").get(0);
Rule rule = new Rule();
rule.group = facilityAdmins;
rule.crudFlags = "CRUD";
rule.what = "NotificationRequest";
port.create(sessionId, rule);
```

Science & Technology Facilities Council

# Create Many, Delete Many

- Each create, get, update and delete call will result in one message for each matching request if the operation is successful
- The createMany and deleteMany calls produce one notification message for each matching NotificationRequest, for each entity, if the operation is successful
- The search operation produces one message per notification request if the operation is successful
  - The message will contain the query string if this was requested.
- The what field in the NotificationRequest must be just the entity name other the notification will not match.

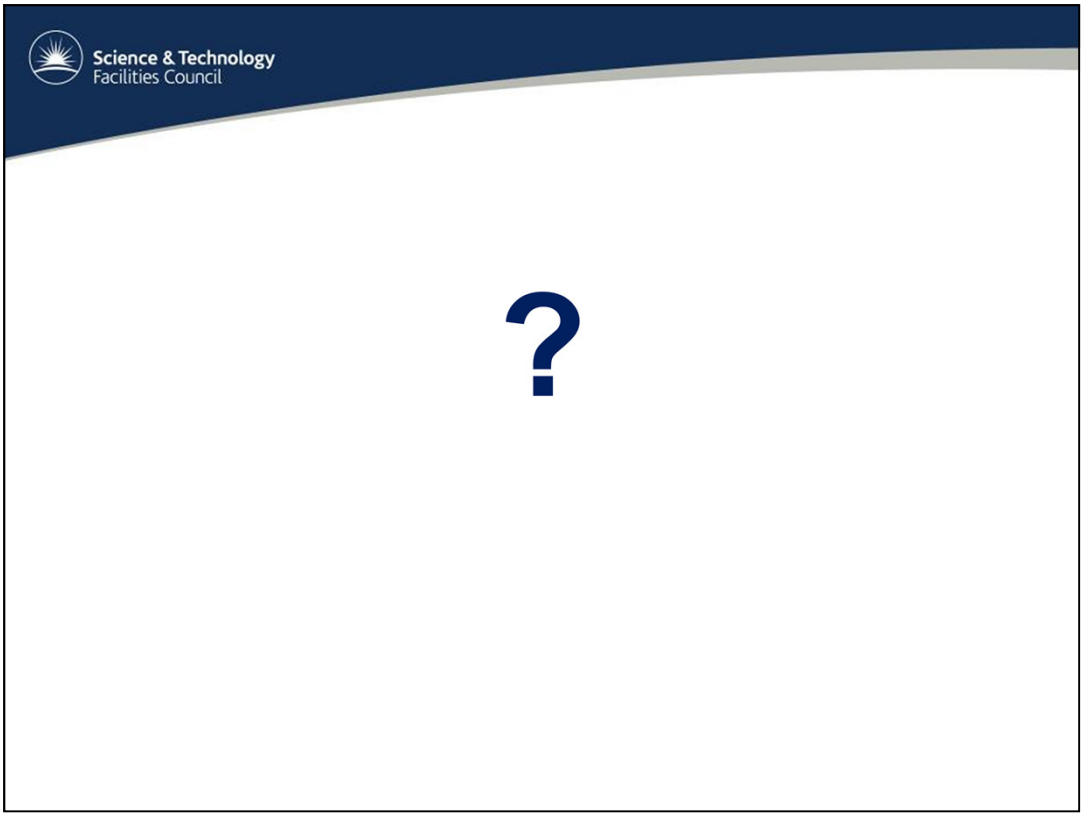
Science & Technology Facilities Council

Each create, get, update and delete call will result in one message for each matching request if the operation is successful. The createMany and deleteMany calls produce one notification message for each matching NotificationRequest, for each entity, if the operation is successful. The search operation produces one message per notification request if the operation is successful. The message will contain the query string if this was requested. The what field in the NotificationRequest must be just the entity name other the notification will not match.

# Summary

- Notifications are sent via JMS
- Use CRUD and What to define triggering of notifications
- Publish subscribe or point to point
- ICAT authorization rules should be added for the NotificationRequest table
- http://www.icatproject.org/mvn/site/icat/4.2.1/icat.client/manual.html

Science & Technology Facilities Council


Science & Technology
Facilities Council
?