Science and Technology Facilities Council

ICAT Performance: Rules and Public Steps

Sections

1 Rules, Public Tables, Public Steps

2 Authorization in ICAT architecture

3 Impact of authorization on performance

4 General comments

Science and Technology Facilities Council

Rule

Allows creates, reads, updates, deletes (CRUD) to be performed on entities

Can be applied to users from specific groups, or everyone

Logic is defined as the field “what”:

- `Datafile`
- `SELECT df FROM Datafile df JOIN df.dataset d JOIN d.investigation i JOIN i.investigationInstruments ii JOIN ii.instrument inst JOIN inst.instrumentScientists instSci JOIN instSci.user u WHERE d.name='raw' AND u.name = :user`

Rule

“what” is then turned into JPQL for three purposes:

CRUDJPQL

```
SELECT COUNT(Datafile$.id) FROM  Datafile AS Datafile$ WHERE
Datafile$.id = :pkid
```

IncludeJPQL

```
SELECT Datafile$.id FROM  Datafile AS Datafile$ WHERE
Datafile$.id IN (:pkids)
```

SearchJPQL

```
SELECT Datafile$.id FROM  Datafile AS Datafile$
```

Public Tables

Public tables are defined by SQL:

```
SELECT DISTINCT r.bean
FROM Rule r LEFT JOIN r.grouping g
WHERE r.restricted = FALSE AND g IS NULL
```

- Restricted is false when the rule applies to all entities of a given type
- If the user group is null, then the rule is applied to everyone

- Effectively, this is just an ICAT rule that gets special treatment
- The entity names it applies to are cached
- Saves time when authorizing as do not need to evaluate anything

Public Steps

Much simpler than rules:

- Defines an origin entity (e.g. Investigation)
- Defines a field on that entity (e.g. samples)

If there is a rule which lets you see the former, then implicitly you are allowed to see the latter

Cached like public tables are

Used for include queries (potentially recursively)

Root Access

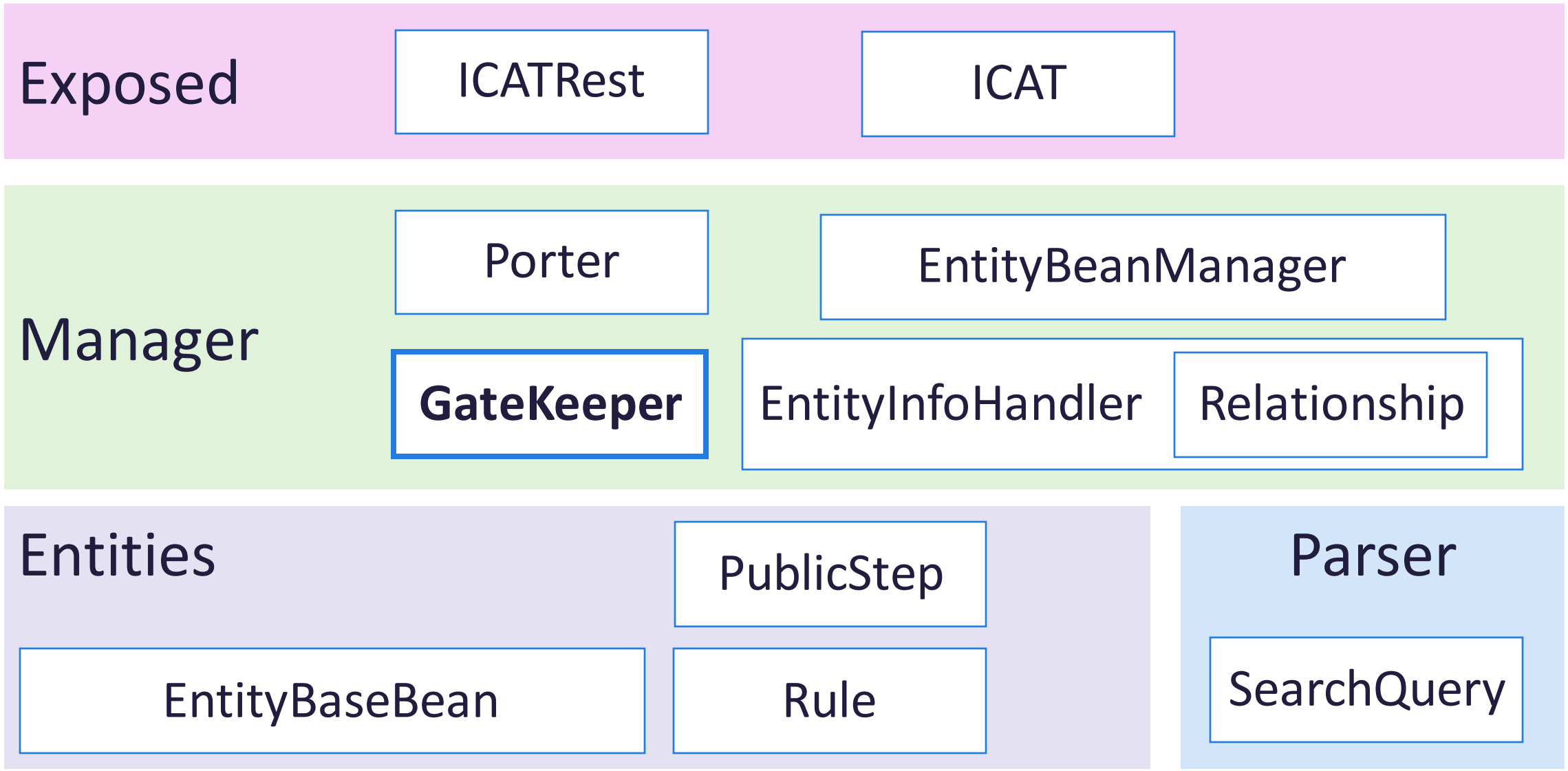
Separate to the **User/UserGroup** tables, ***run.properties*** for ICAT server defines:

```
rootUserNames = db/root simple/root
```

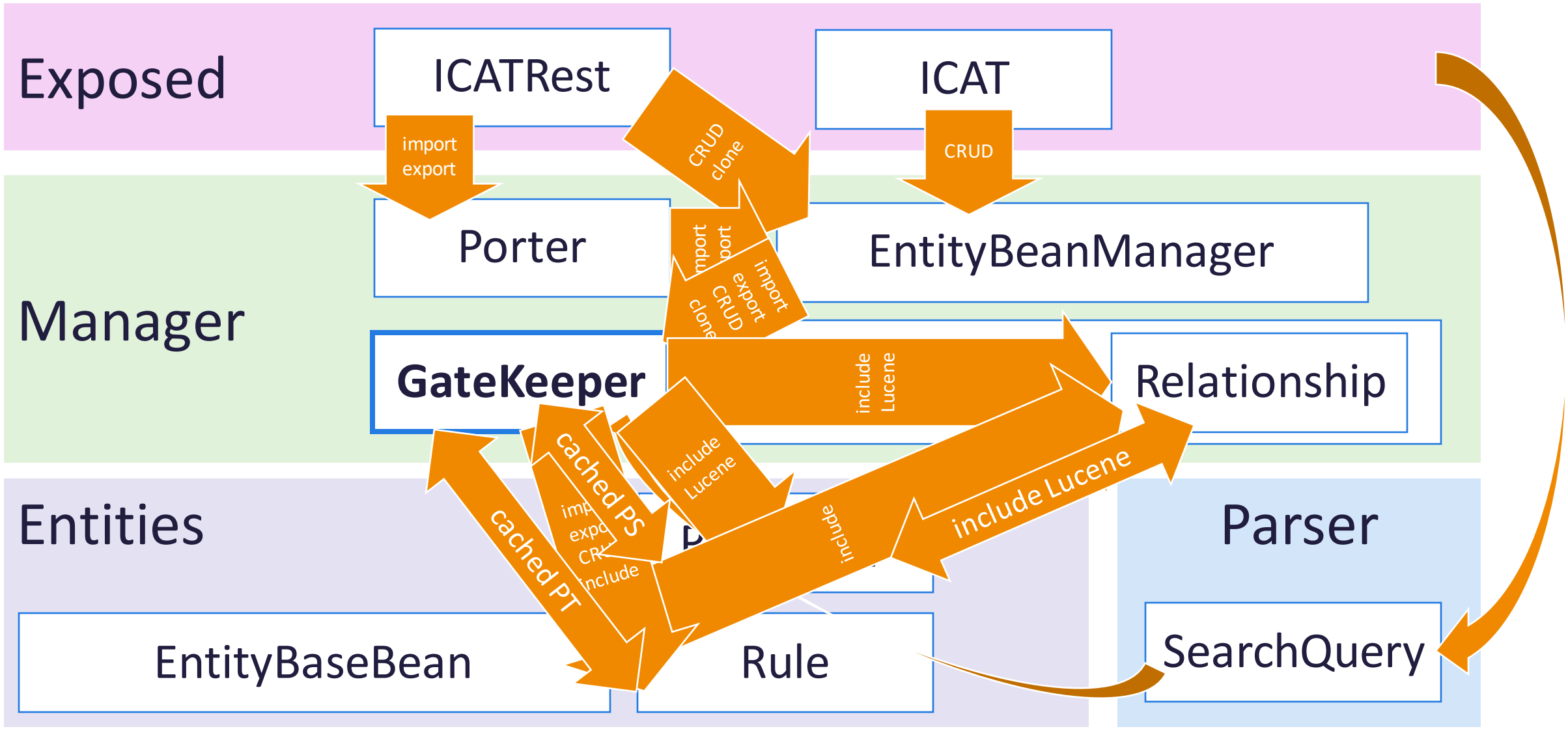
Before using Rules orPublicSteps, the user is checked against this list

If the user “is root”, then they automatically pass all authorization

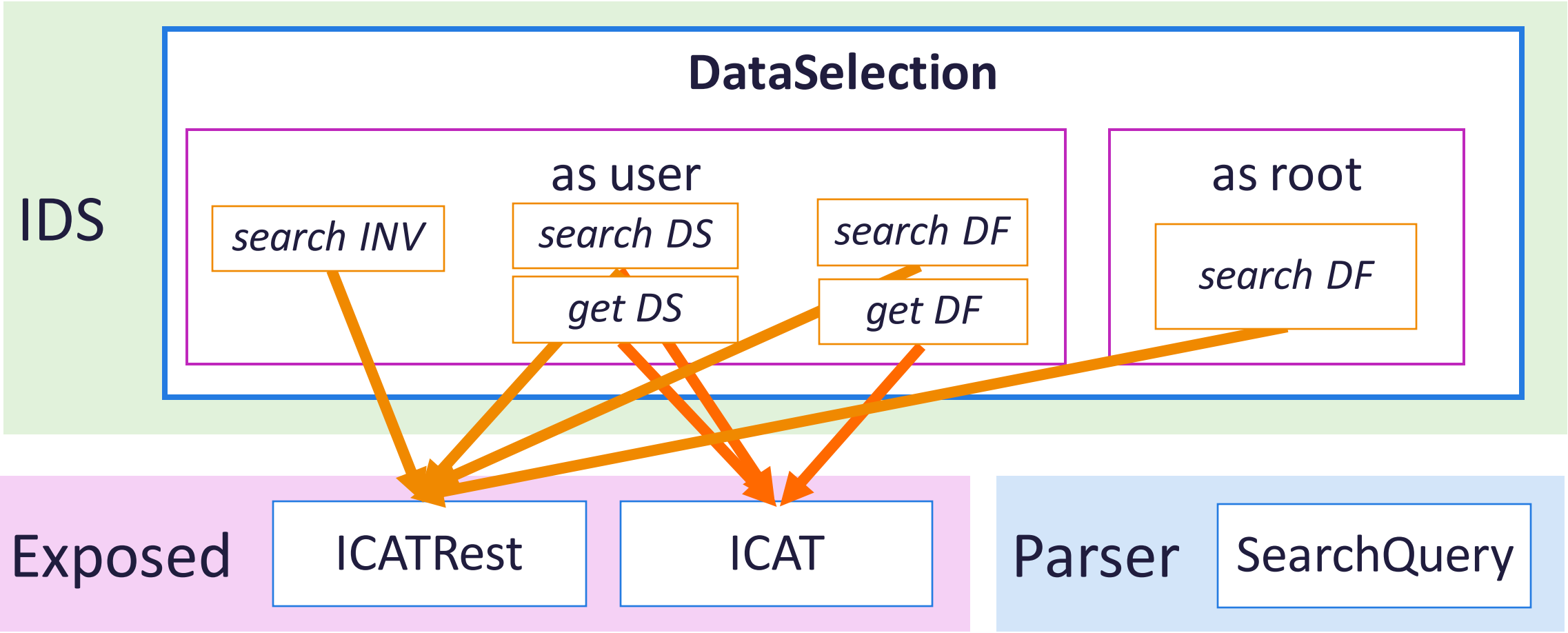
Architecture: icat.server



Architecture: icat.server



Architecture: ids.server



Science and Technology Facilities Council

IDS: SearchQuery

Identified in March 2021 for IDS: https://github.com/icatproject/ids.server/issues/115

Requests took longer than 30 minutes due to complexity of rules added by the SearchQuery:

```
SELECT * FROM (SELECT a.*, ROWNUM rnum FROM (
   SELECT t0.ID AS a1, t0.NAME AS a2, t0.LOCATION AS a3, t0.CREATE_ID AS a4, t0.MOD_ID AS a5 FROM DATAFILE t0 WHERE (
      (
         ((t0.DATASET_ID = :1 ) AND (t0.LOCATION IS NOT NULL)) AND (t0.ID BETWEEN :2 AND :3 )
      ) AND (
         (t0.ID IN (
            SELECT t1.ID FROM INVESTIGATION t3, DATASET t2, DATAFILE t1 WHERE ((t3.VISIT_ID IN (:4 , :5 )) AND ((t2.ID = t1.DATASET_ID) AND (t3.ID =
t2.INVESTIGATION_ID)))
         ) OR t0.ID IN (
            SELECT DISTINCT t4.ID FROM INVESTIGATIONINSTRUMENT t7, INVESTIGATION t6, DATASET t5, DATAFILE t4, USER_ t10, INSTRUMENTSCIENTIST t9,
INSTRUMENT t8 WHERE ((t10.NAME = :6 ) AND ((((((t5.ID = t4.DATASET_ID) AND (t6.ID = t5.INVESTIGATION_ID)) AND (t7.INVESTIGATION_ID = t6.ID)) AND (t8.ID =
t7.INSTRUMENT_ID)) AND (t9.INSTRUMENT_ID = t8.ID)) AND (t10.ID = t9.USER_ID)))
         )) OR t0.ID IN (
            SELECT DISTINCT t11.ID FROM INVESTIGATION t13, DATASET t12, DATAFILE t11, USER_ t15, INVESTIGATIONUSER t14 WHERE ((t15.NAME = :7 ) AND ((((t12.ID =
t11.DATASET_ID) AND (t13.ID = t12.INVESTIGATION_ID)) AND (t14.INVESTIGATION_ID = t13.ID)) AND (t15.ID = t14.USER_ID)))
         )
      )
   )
) a WHERE ROWNUM <= :8 ) WHERE rnum > :9
```

Science and Technology Facilities Council

IDS: SearchQuery Solution

Authorize Datasets as normal (less expensive than Datafiles)

If the Dataset was authorized, skip Datafile authorization by using a root account to perform the query

Controlled by config option

Comparable to a PublicStep between Dataset and Datafile

Science and Technology Facilities Council

IDS: Includes

Also identified in March 2021 for IDS:
https://github.com/icatproject/ids.server/issues/117

Performing the following took 2 seconds (get with id provided):

```
Dataset ds INCLUDE ds.investigation.facility
```

Performing the following took 80ms:

```
SELECT ds.id, ds.name, ds.location, inv.id, inv.name, inv.visitId, fac.id, fac.name FROM
Dataset ds JOIN ds.investigation inv JOIN inv.facility fac WHERE ds.id=?
```

Authz methods differ in each case.

IDS: Includes Solution?

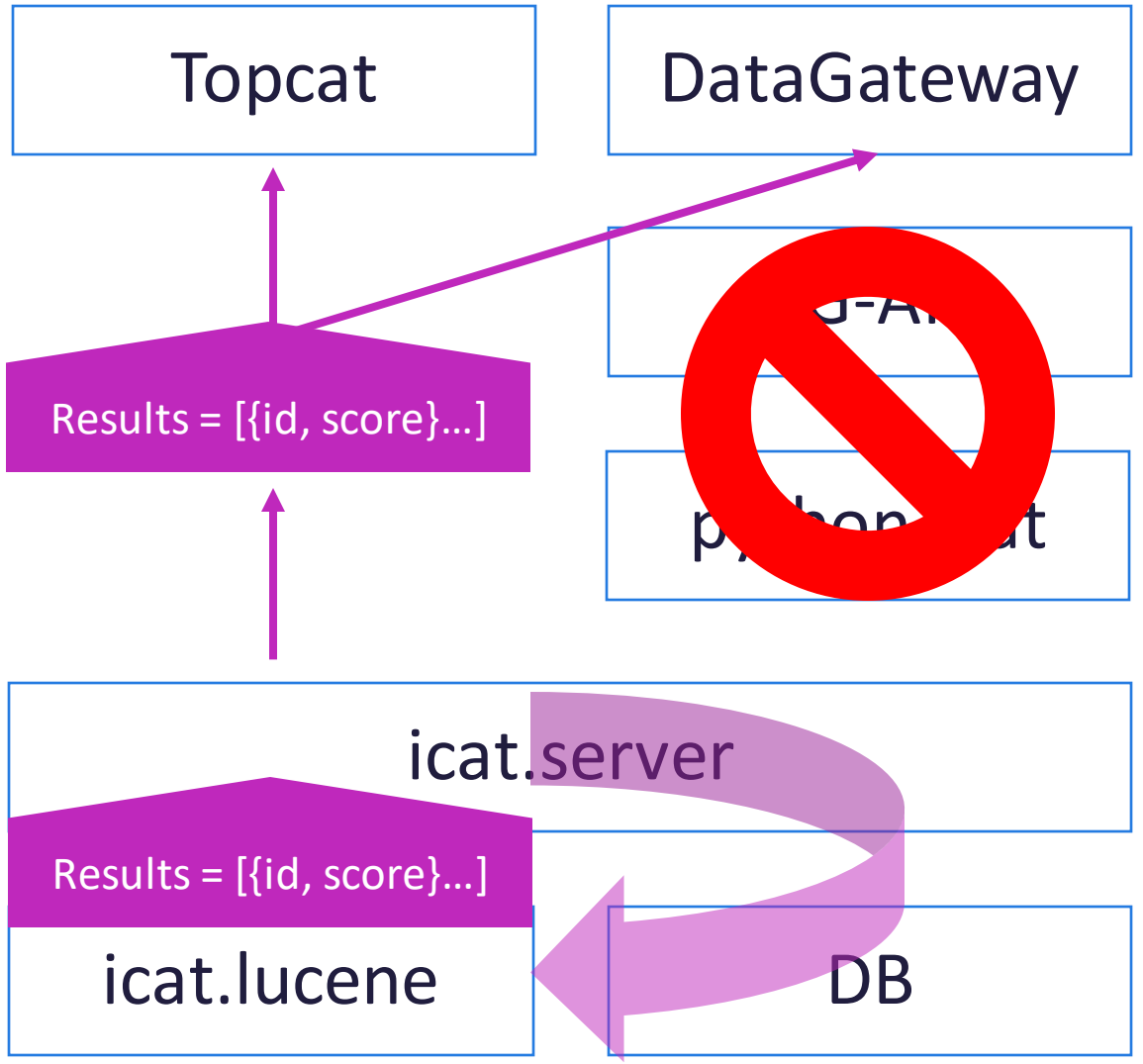
Never addressed, but possible approaches would be:

- Use of PublicSteps: Dataset -> investigation, Investigation -> facility
  - If not already in place, and if appropriate
  - INCLUDE queries will send subsequent DB calls for each included entity
  - If there's a PublicStep in place, can be authorized within ICAT server

- Replace INCLUDE with JOINs:
  - As documented the second query ran faster
  - Uses the SearchQuery to perform authorization during the search
  - On the other hand, this didn't work well for the previous Datafile example…

icat.lucene: Searching

- icat.lucene component returns ids of entities which match the search text
- icat.server performs authorization on each result with a **separate** query to the database
- If we don't have enough authorized results, go back for another batch and repeat
- Once the frontend has a list of authorized ids, it will submit another query which will perform authorization **again**



Science and Technology Facilities Council

icat.lucene: Searching Solution(s)

Alongside other changes to free text search:

- Get all metadata directly from the Lucene index (remove second DB call)
- Authorize ids in batches (configurable in size but ~1000 to 10000)
- Optional: return early if a minimum number of results found
- Optional: instead of searching entire index, only search results where the user is InstrumentScientist or InvestigationUser
  - Drastically limits number of returned results, and expect that all results returned will pass authorization
- Configurable: timeout long running searches

icat.lucene: Includes

In order to get all metadata directly from the Lucene index, need to return related metadata

- E.G. Dataset table has column for Investigation title in DataGateway:

Name	Datafile Count	Investigation	Create Time	Modified Time
raw	2		12/09/2007	05/08/2016

icat.lucene has no concept of Rules or PublicSteps, and replicating this would be:

- Difficult – would need to index and keep these up to date in Lucene
- Probably slow – in principle need to check multiple includes for every result

Science and Technology Facilities Council

icat.lucene: Includes Solution

Public Tables and Public Steps identify things we can quickly authorize, and are cached in ICAT server

- From these, build cached lists of which Lucene fields are safe to return
- Only request these fields from Lucene in the first place
- At this point, only need to authorize the “main” entity being searched

Name	Datafile Count	Investigation	Create Time	Modified Time
analyzed	Unknown	SGC BAG	11/04/2014	11/04/2014

Downsides are:

- Overly restrictive – doesn’t take all Rules into account
- Might not want to create a particular Public Step

Science and Technology Facilities Council

General Comments

Areas of difficulty:

- INCLUDE queries
  - Relies on “pruning” after original search, so can take longer
- Datafiles
  - When there are a lot, things that normally work can break down (e.g. SearchQuery)

Potential solutions:

- Hardcoded workarounds (using root, caching fields for Lucene)
- Creation of more PublicSteps (where possible)
- Syntax of query (hard to anticipate but has an impact)

Caveats

Authorization performance depends on a lot of things:

- Rules and Public Tables
- Public Steps
- Exact query syntax
- Who the current user is
- How much data does that user have access to
- How much data does that user NOT have access to
- SOAP versus Rest(like) API in icat.server

Ultimately difficult to make categorical statements about performance