



National Institute of Environmental Health Sciences  
*Your Environment. Your Health.*

# Updates on iRODS Data Repository Service Adapter

**Mike Conway, Deep Patel**

**National Institute of Environmental Health Sciences**

# Software and AI/ML Tools for Computational Toxicology

"Computational toxicology is a subdiscipline of toxicology that aims to use the mathematical, statistical, modeling, and computer science tools to better understand the mechanisms through which a given chemical induces harm and, ultimately, to be able to predict adverse effects of the toxicants on human health and/or the environment."



Rusyn I, Daston GP. Computational toxicology: realizing the promise of the toxicity testing in the 21st century. *Environ Health Perspect.* 2010 Aug;118(8):1047-50. doi: 10.1289/ehp.1001925. Epub 2010 May 18. PMID: 20483702; PMCID: PMC2920091.

## Current Activities

### IRODS at NIEHS

iRODS has been in use for several years to preserve NGS data

- **20,075,498 raw run files**
- **328,426 processed FASTQ files**

**All runs have full metadata at the project and sample level**



# Challenges we Face

Standards, Sustainable Software Practices, Cloud-ready, FAIR Research Software

## Data Sharing and Federation

- Increasingly data-driven
- Multi-institutional data collaboration
- Hybrid cloud/on-prem computing
- Heterogeneous data, institutional, specialized and generalist repositories
- Handling PHI/PII, maintaining privacy

## AI and Machine Learning

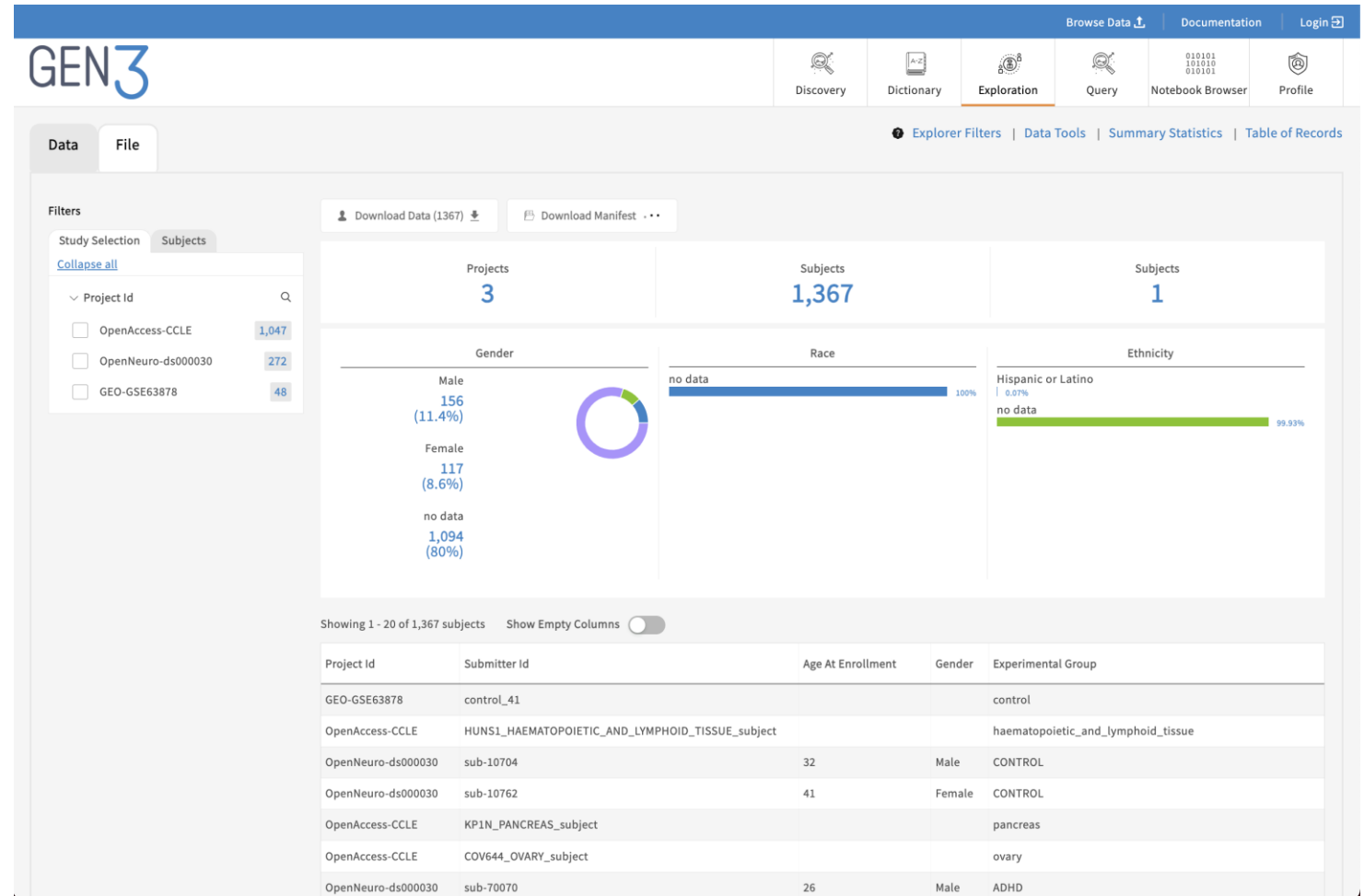
- Systematic Review, extracting data from reports and publications
- ToxPipe - Interacting with and discovering relationships in diverse data sources



Elevating the role of Research Software Engineers

# CHORDS Project

- Intersection of Climate and Health (Wildfires) - A Data Catalog
- Gen3 Platform (Biomedical Research Hub)
- An attempt to better align NIEHS with NIH platforms/standards
- **Data Grid, Data Commons, Data Mesh**



# Data Sharing Architectures (As defined by Gen3 developers)

## Data Commons

Software platforms that co-locate:

1. curated data
2. cloud-based computing infrastructure, and
3. commonly used software applications, tools and services to create a governed resource for managing, analyzing and sharing data with a research community.

## Data Mesh

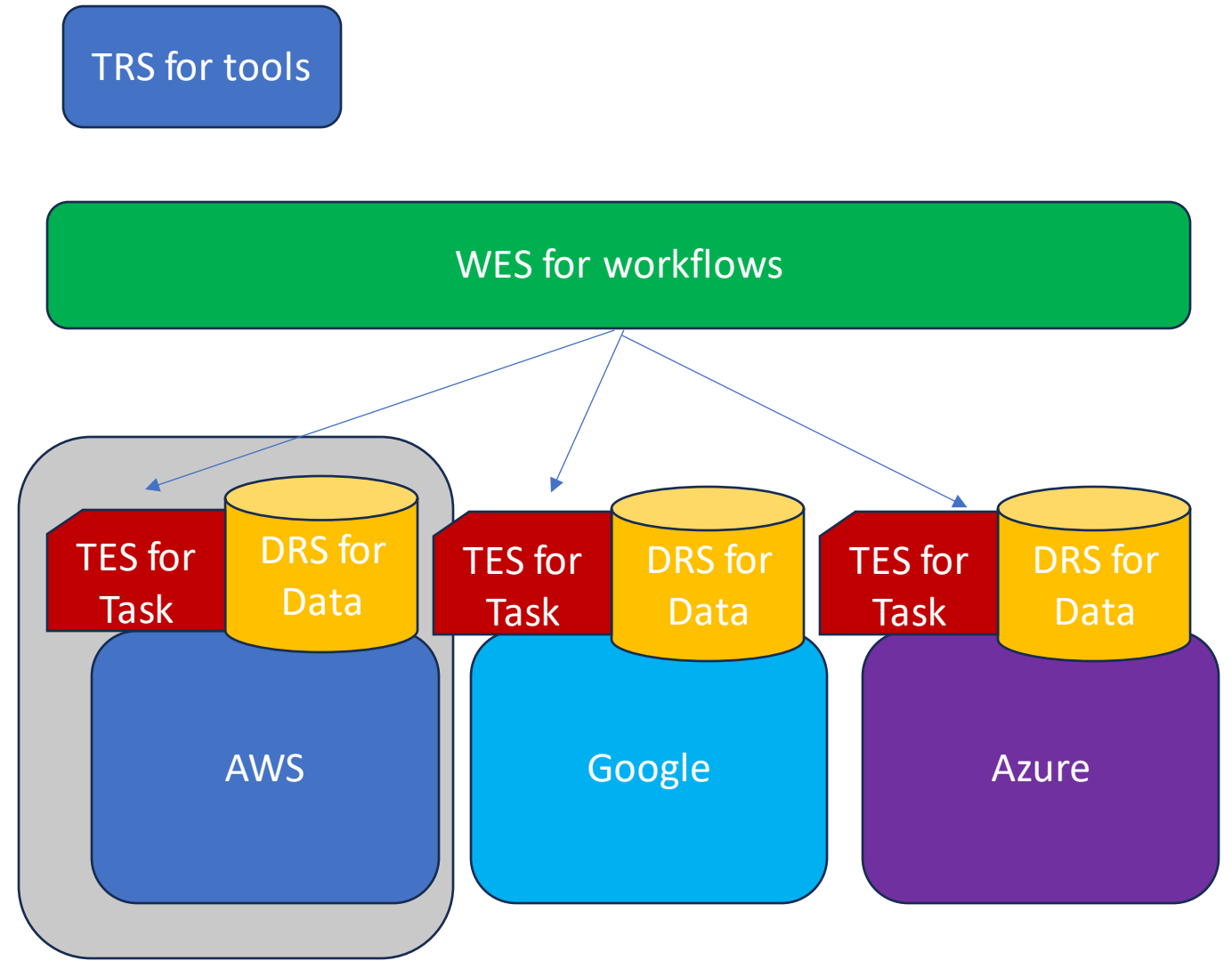
(aka data ecosystems) integrate multiple data commons, computational platforms, and other cloud-based resources operated by different organizations, along with a hybrid governance framework, and enable the management, discovery, analysis and sharing of data.

Bob Grossman. "How to Build a Data Mesh Using Gen3," May 1, 2023.

<https://gen3.org/community/events/Gen3%20Forum%20May%201%202023%20-%20Data%20Meshes.pdf>.

## Federated Analysis

- Registry of Tools
- Assembled into workflows
- Dispatched to task runners
- Referencing Data
- Mediated by data usage policies and researcher identity



WES for workflows

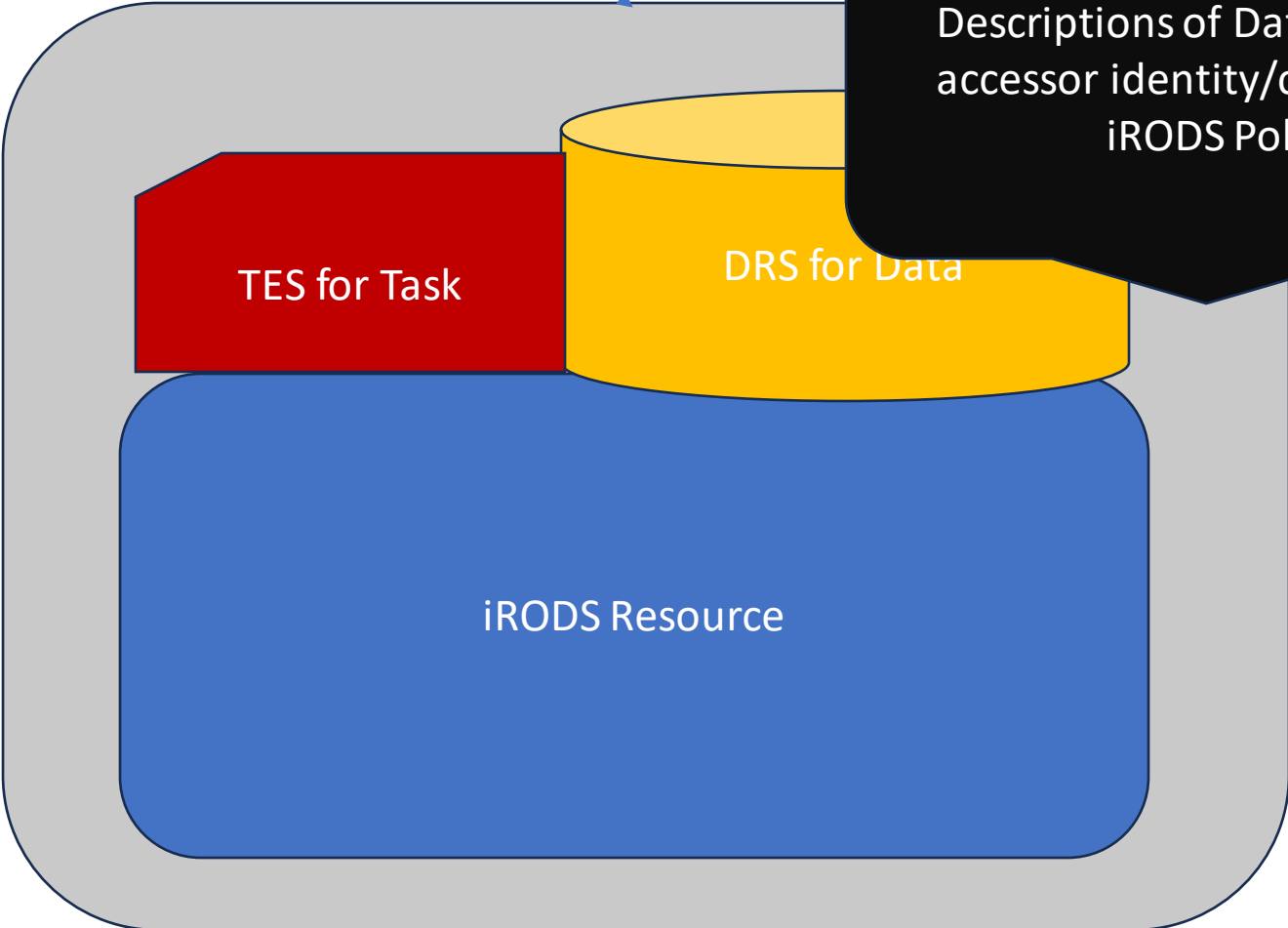
TES for Task

An iRODS Resource is an on-prem analogue of a data set in a particular location (cloud+region) and can take advantage of data locality

iRODS Resource



WES for workflows



Descriptions of Data Usage limitations are married with accessor identity/claims - this is a natural application of iRODS Policy-Based Data Management

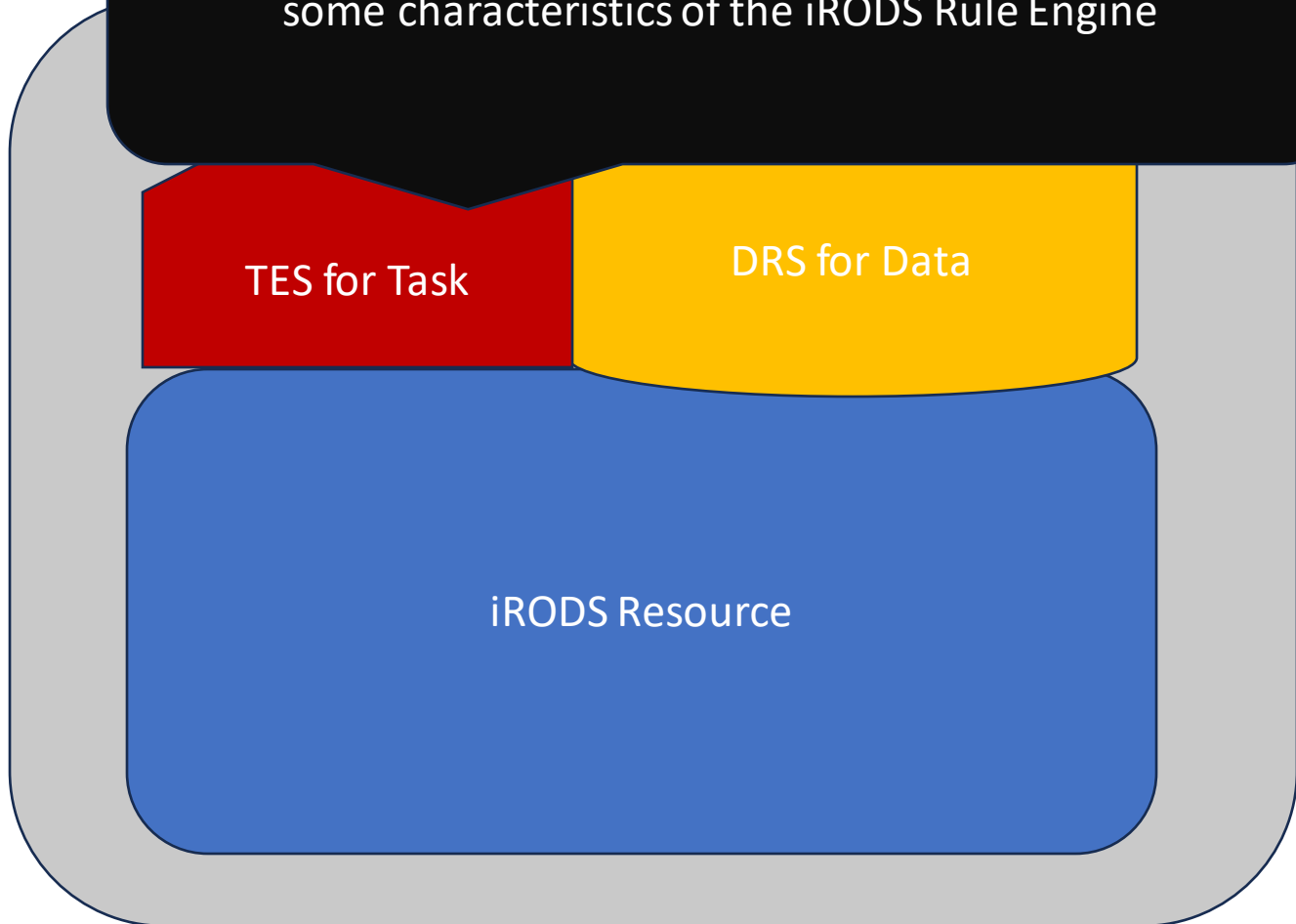
WES for workflows

TRS is a task runner with a REST-ful interface that shares some characteristics of the iRODS Rule Engine

TES for Task

DRS for Data

iRODS Resource



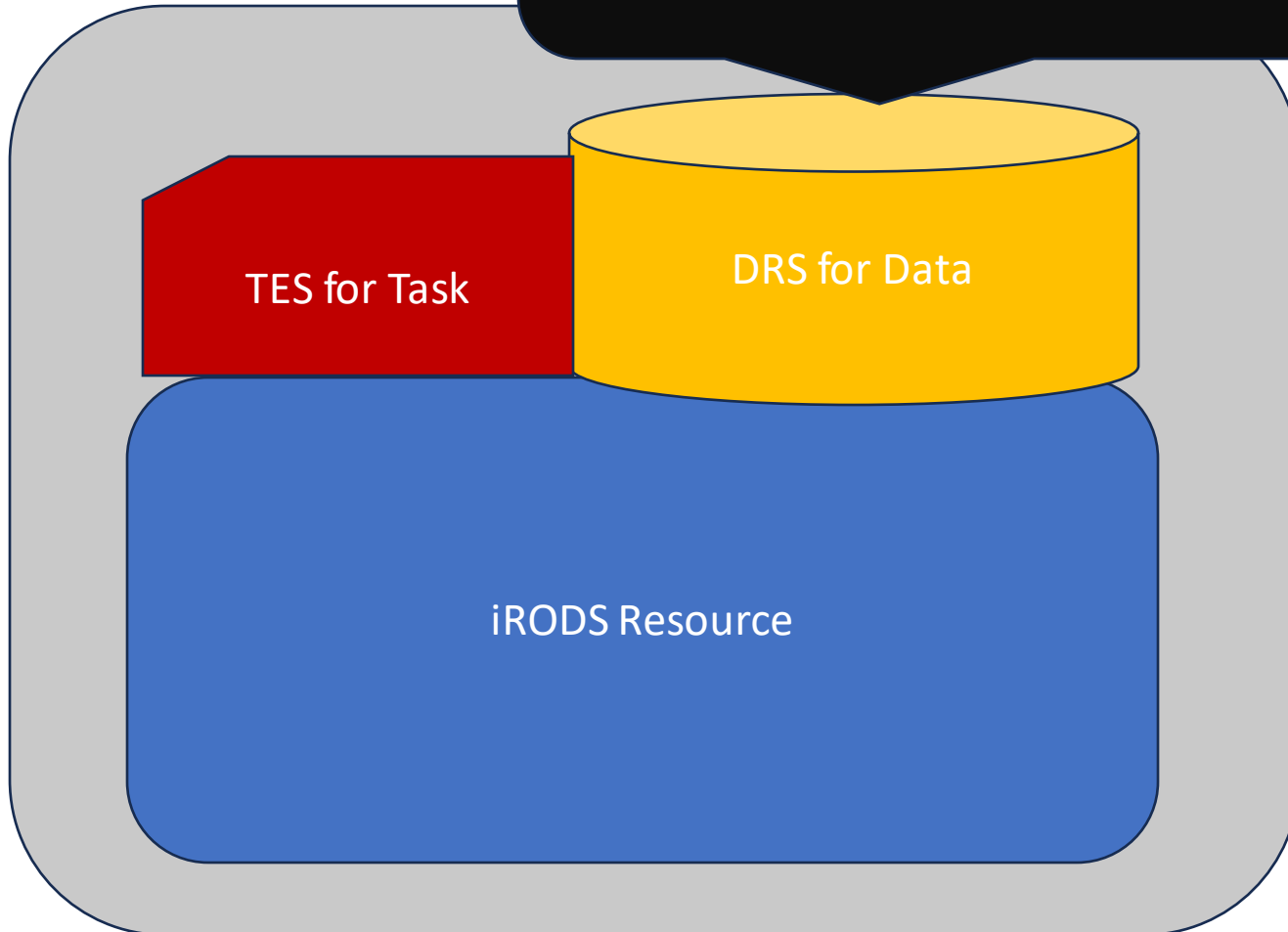
WES for workflow

DRS is a REST-ful interface that maps iRODS Data Objects to a GUID via AVU metadata decoration

TES for Task

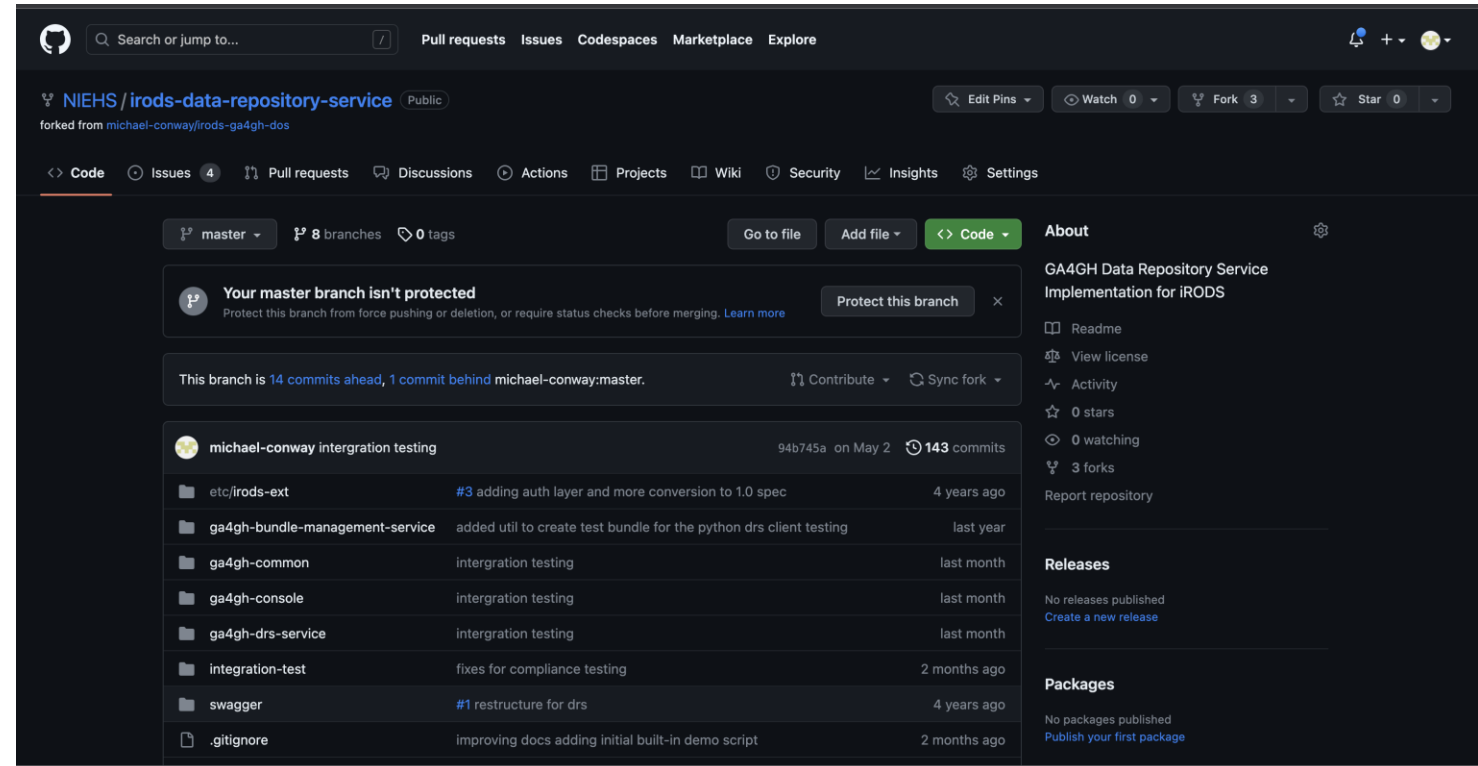
DRS for Data

iRODS Resource



# iRODS DRS Interface

- <https://github.com/NIEHS/irods-data-repository-service>
- Implements Version 1.2  
<https://ga4gh.github.io/data-repository-service-schemas/preview/release/drs-1.2.0/docs/>
- Currently testing with DRS Compatability Suite (<https://github.com/ga4gh/drs-compliance-suite>)
- Previously demonstrated at GA4GH Connect, GA4GH DRS Hackathon
- Current work:
  - Adding Authn via Keycloak
  - Upgrading to 1.3 Spec



Firing up the included Compose framework  
This starts an iRODS server, the DRS API, as well as a 'starter' console that emulates potential iCommand interfaces.

```
conwaymc@ALMBP02246093 ~ % docker ps
CONTAINER
ID IMAGE COMMAND CREATED STATUS
PORTS NAMES
0b717911971d michaelconway/ga4gh-drs:latest "/runit.sh" 2
minutes ago Up 2 minutes 0.0.0.0:8080->8080/tcp irods-drs
e3cc8967907d michaelconway/irods-rest2:1.0.1 "/runit.sh" 6
weeks ago Up 2 minutes 0.0.0.0:8888->8080/tcp irods-rest
324d035af2d7 michaelconway/ga4gh-console:latest "/runit.sh" 6
weeks ago Up 2 minutes ga4gh_console
15aefedb2a2d compose-irods-catalog-provider "./start_provider.sh" 6
weeks ago Up 2 minutes 0.0.0.0:1247->1247/tcp, 1248/tcp irods-
catalog-provider
conwaymc@
```

# GA4GH Console

- Follows iCommands style
- Make test files for testing
- Turn collections into bundles

In reality, making a bundle just adds AVU metadata at collection and data object level with GUIDS and Checksum information

```
shell:>help
```

## AVAILABLE COMMANDS

### Built-In Commands

clear: Clear the shell screen.

exit, quit: Exit the shell.

help: Display help about available commands.

script: Read and execute commands from a file.

stacktrace: Display the full stacktrace of the last error.

### Drs Bundles Command

\* icd: Change working directory in iRODS

iinit: Initialize connection

\* ilistdrsb: List all DRS bundles

\* ils: List directory contents

\* imakedrsb: Make a DRS bundle at current directory

\* ipwd: Print working directory in iRODS

\* irmdrsb: Remove a DRS bundle by directory path or GUID

maketestbundle: Create test bundle

Commands marked with (\*) are currently unavailable.

Type `help <command>` to learn more.

## Make a bundle

- Mark iRODS collection as a DRS Bundle
- AVUs mark collection and data objects with GUIDs
- Does some checksum computation for the whole bundle and adds as AVU

```
shell:>ils
```

```
/tempZone/home/test1
```

```
    /tempZone/home/test1/testbundle2  
    COLLECTION
```

```
shell:>icd testbundle2
```

```
/tempZone/home/test1/testbundle2
```

```
shell:>imakedrsb
```

```
created bundle with GUID:64487235-4bb0-4849-8eff-  
957581ff933f
```

```
shell:>
```

# Obtain a Bearer Token (JWT)

- Currently via REST Client
- KeyCloak Implementation is next step!

The screenshot shows a REST Client interface for a POST request to `http://localhost:8888/token`. The request headers are:

KEY	VALUE	DESCRIPTION
Authorization	Basic dGVzdDE6dGVzdA==	
Postman-Token	<calculated when request is sent>	
Content-Length	0	
Host	<calculated when request is sent>	

The response status is `200 OK` with a time of `4.39 s` and a size of `531 B`. The response body is a JWT token:

```
1 eyJhbGciOiJIUzUxMiJ9.eyJzdWIiOiJ0ZXN0MSIsImIzcyI6Imlyb2RzLXJlc3QyIiwiaWF0IjoxNjg2NzU1NDg2fQ.yp1UFxU0nAdR1Agb1bVb1VjsqBnLj4QzKv7AtbY8zg_fqSZzZsLuBRQcGty2N4tmAj5N0So3eUKZMcgkhu2m5A
```



# Obtain information on Bundle

- Pass a GUID
- Describes the bundle and the constituent objects

iRODS GA4GH DRS / Obtain DRS Bundle

GET http://localhost:8080/ga4gh/drs/v1/objects/64487235-4bb0-4849-8eff-957581ff933f

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies

Type Bearer Token Token eyJhbGciOiJIUzUxMiJ9.eyJzdWIiOiJ0ZXN0I...

The authorization header will be automatically generated when you send the request. [Learn more about authorization](#)

Body Cookies Headers (9) Test Results Status: 200 OK Time: 1325 ms Size: 2.47 KB Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": "64487235-4bb0-4849-8eff-957581ff933f",
3   "name": "/tempZone/home/test1/testbundle2",
4   "self_uri": "drs://localhost:8080/ga4gh/drs/v1/64487235-4bb0-4849-8eff-957581ff933f",
5   "size": 0,
6   "created_time": "2023-06-14T15:06:11Z",
7   "updated_time": "2023-06-14T15:06:11Z",
8   "version": "0",
9   "mime_type": "text/directory",
10  "checksums": [
11    {
12      "checksum": "44b474940e65c695a965edd13c85e853862628fe9f3b9436b0d119f183415a4b",
13      "type": "sha256"
14    }
15  ],
16  "access_methods": [],
17  "contents": [
18    {
```

## DRS File in Bundle

- Response highlights a DRS Object in the bundle and a URI that references

- "contents": [
  - {
  - "name": "file0",
  - "id": "11fb34e7-776b-479a-baf8-3e4c28296655",
  - "drs\_uri": [
    - "drs://localhost:8080/ga4gh/drs/v1/objects/11fb34e7-776b-479a-baf8-3e4c28296655"
  - ],
- "contents": []
- },

- Use DRS Object ID
- Returns similar structure with some important additions
  - Let's highlight the access methods

The screenshot shows a REST client interface for a GET request to the endpoint `http://localhost:8080/ga4gh/drs/v1/objects/11fb34e7-776b-479a-baf8-3e4c28296655`. The request is configured with a Bearer Token authorization header. The response is a JSON object with the following structure:

```
1 {
2   "id": "11fb34e7-776b-479a-baf8-3e4c28296655",
3   "name": "/tempZone/home/test1/testbundle2/file0",
4   "self_uri": "drs://localhost:8080/ga4gh/drs/v1/11fb34e7-776b-479a-baf8-3e4c28296655",
5   "size": 10,
6   "created_time": "2023-06-14T15:06:11Z",
7   "updated_time": "2023-06-14T15:06:11Z",
8   "version": "",
9   "mime_type": "application/octet-stream",
10  "checksums": [
11    {
12      "checksum": "meqqgL/qU9NmnxsTeoUbx/tBesbVff4sV9kilAkj5kk=",
13      "type": "SHA256"
14    }
15  ],
16  "access_methods": [
17    {
18      "type": "file",
```

## Access Methods

- Type is focused on HTTPS as well as S3, here we add an iRODS type but would require special provisioning in the task running environment
- iRODS S3 interface development enables new access methods in DRS
- HTTPS can target the new iRODS REST interface, here it's using the prior Jargon interface for prototyping.
- Region in DRS is focused on cloud but can map to iRODS zone:resource for data locality

```
"access_methods": [  
  {  
    "type": "file",  
    "access_url": {  
      "url": "irods://test1@irods-catalog-  
provider:1247/tempZone/home/test1/testbundle2/f  
ile0",  
      "headers": []  
    },  
    "access_id": "irods",  
    "region": ""  
  },  
  {  
    "type": "https",  
    "access_url": null,  
    "access_id": "irods-rest",  
    "region": null  
  }  
],
```

Pass DRS GUID and access method to obtain the Access URL

- This step converts the DRS reference into an accessible endpoint
- Invoking this method creates an iRODS REST API call with an attached iRODS Ticket

The screenshot shows a REST client interface for the endpoint `http://localhost:8080/ga4gh/drs/v1/objects/11fb34e7-776b-479a-baf8-3e4c28296655/access/irods-rest`. The request is a GET method. The response status is 200 OK, with a time of 617 ms and a size of 447 B. The response body is displayed in JSON format:

```
1 {
2   "url": "http://localhost:8888/fileBytes?path=%2FtempZone%2Fhome%2Ftest1%2Ftestbundle%2Ffile0",
3   "headers": [
4     "X-API-KEY GDbqYsNJbv1K4z9"
5   ]
6 }
```