A GA4GH Data Repository Service for iRODS

Mike Conway
Data Systems Architect/Engineer
National Institute of Environmental Health Sciences
NIEHS Office of Data Science
Developing an NIEHS Data Commons

Developing a Commons to manage research data, using iRODS as a platform for unifying and managing local and cloud resources.
Data Commons integrated with processing pipelines and workflow systems.

Use Case:
• Data Commons as the hub for managing research projects in an ISA model
• Sample submission integrated with Clarity LIMS triggers NextFlow pipelines
• Data Commons as delivery mechanism gathering metadata and pipeline results

Setting future strategy anticipating move to cloud over time, with a hybrid of local research data, published artifacts and tiered storage in the cloud.

How can we develop strategies that work for cloud and local use cases?
GA4GH Cloud Work Stream APIs

- Tool Registry Service (TRS)
- Workflow Execution Service (WES)
- Task Execution Service (TES)
- Data Object Service (DOS) (now the Data Repository Service, DRS)

Sharing Tools and Workflows
Executing Workflows
Executing Individual Tasks
Accessing Data

GA4GH Data Repository Service

Described by GA4GH:

“The Data Repository Service (DRS) API provides a generic interface to data repositories so data consumers, including workflow systems, can access data in a single, standard way regardless of where it’s stored and how it’s managed. The primary functionality of DRS is to map a logical ID to a means for physically retrieving the data represented by the ID.”
GA4GH DRS implementation for ‘native’ iRODS collections.

Service to designate an iRODS Collection ‘in place’ as a Data Bundle.

URL creation, including ticket based access via https are supported.

Low barrier to entry, no special setup, stateless Docker image.

https://github.com/michael-conway/irods-ga4gh-dos
Demo – Designate an iRODS Collection as a Data Bundle

Code snippet designates a collection root as a bundle

```java
String bundleRoot = irodsCollectionRootAbsolutePath + "/" + bundleDir;
DosConfiguration dosConfiguration = new DosConfiguration();
DosServiceFactory factory = new ExplodedDosServiceFactoryImpl(irodsFileSystem.getIRODSAccessObjectFactory());

ExplodedDosBundleManagementServiceImpl
    .getIRODSAccessObjectFactory(), irodsAccount, factory, dosConfiguration);
String guid = explodedDosService.createDataBundle(bundleRoot);
Assert.notNull("no guid returned", guid);
```

Marks bundle with AVUs for GUID and checksum of checksums

```bash
@server1:~$ imeta ls -C /zone1/home/test1/jargon-scratch/ExplodedDosBundleManagementServiceImplTest/testCreateDataBundle
AVUs defined for collection /zone1/home/test1/jargon-scratch/ExplodedDosBundleManagementServiceImplTest/testCreateDataBundle:
attribute: ga4gh:bundleChecksum
value: d68d4685be3b13168b63bb31df1c0a4ee9c267a0cbe300869b08768c43d9b6dd
units: irods:ga4gh:doi
attribute: ga4gh:bundleId
value: aa713031-e421-4783-b522-068b63020def
units: irods:ga4gh:doi
```

```bash
@server1:~$ 
```
Demo – Designate an iRODS Collection as a Data Bundle

Child objects (nested) flattened and marked as a Data Object. GUID is added as AVU and checksum is computed.
## Data Repository Service

https://github.com/ga4gh/data-repository-service-schemas

**Contact the developer**  
Apache 2.0

<table>
<thead>
<tr>
<th>Method</th>
<th>URL Path</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>/bundles/{bundle_id}</td>
<td>Get info about a Data Bundle.</td>
</tr>
<tr>
<td>GET</td>
<td>/objects/{object_id}</td>
<td>Get info about a Data Ga4ghObject.</td>
</tr>
<tr>
<td>GET</td>
<td>/objects/{object_id}/access/{access_id}</td>
<td>Get a URL for fetching bytes.</td>
</tr>
<tr>
<td>GET</td>
<td>/service-info</td>
<td>Get information about this implementation.</td>
</tr>
</tbody>
</table>

[BASE URL: /, API VERSION: 0.6.0]
Service Info - Configurable

```
curl -X GET --header 'Accept: application/json' 'http://localhost:8080/service-info'
```

**Request URL**
http://localhost:8080/service-info

**Request Headers**
```
{
  "Accept": "application/json"
}
```

**Response Body**
```
{
  "version": "0.0.1-SNAPSHOT - 2019-06-25T14:40:59Z",
  "title": "iRODS GA4GH Data Repository Service Demo",
  "description": "",
  "contact": "NIEHS Office of Data Science",
  "license": "BSD 3-Clause"
}
```

**Response Code**
200

**Response Headers**
Retrieve a Data Bundle via GUID


Request URL
http://localhost:8080/bundles/aa713031-e421-4783-b522-068b63020def

Request Headers
{
"Accept": "application/json"
}

Response Body
{
  "id": "aa713031-e421-4783-b522-068b63020def",
  "name": "/zone1/home/test1/jargon-scratch/ExplodedDosBundleManagementServiceImplTest/testCreateDataBundle",
  "size": "0",
  "created": "2019-06-25T11:15:11Z",
  "updated": "2019-06-25T11:15:11Z",
  "version": "0",
  "checksums": [
    {
      "checksum": "d68d4685be3b13168b63bb31df1c0a4ee9c267a0cbe300869b08768c43d9b6dd",
      "name": "sha256"
    }
  ],
  "description": "iRODS exploded bundle collection",
  "aliases": [
    "/zone1/home/test1/jargon-scratch/ExplodedDosBundleManagementServiceImplTest/testCreateDataBundle"
  ],
  "contents": [
    
  ],
  "name": "testFile0.txt",

Checksum of Checksums

Response Code
200

Response Headers
Data Bundle links to child Data Objects

Child Data Objects listed in bundle with GUID and access URI

```
"/zone1/home/test1/jargon-scratch/ExplodedDosBundleManagementServiceImplTest/testCreateDataBundle"
],
"contents": [

  {
    "name": "testFile0.txt",
    "id": "4de8f3ee-73c0-4e47-9f95-617e05998d91",
    "drs_uri": [
      "http://localhost:8080//objects/4de8f3ee-73c0-4e47-9f95-617e05998d91"
    ],
    "type": "object"
  },
  {
    "name": "testFile1.txt",
    "id": "9ade673d-83df-4d2e-ad6b-a468ff9c0243",
    "drs_uri": [
      "http://localhost:8080//objects/9ade673d-83df-4d2e-ad6b-a468ff9c0243"
    ],
    "type": "object"
  },
  {
    "name": "testFile2.txt",
```
Accessing a Data Object by GUID

Individual iRODS file as a Data Object, has GUID, checksum, computed MIME Type
Generating an Access URL on demand

An access method without a URL requires a call to obtain the URL. In this case generating an iRODS ticket on demand for read access.

Try it out!  Hide Response

Curl

curl -X GET --header 'Accept: application/json' 'http://localhost:8080/objects/27119224-78ff-42f6-8e99-a64d39e092ec/access/irods-rest'

Request URL

http://localhost:8080/objects/27119224-78ff-42f6-8e99-a64d39e092ec/access/irods-rest

Request Headers

{
  "Accept": "application/json"
}

Request is for GUID and Access id

Response Body

{
  "url": "http://example.com/irods-rest/fileStream?path=/zone1/home/test1/jargon-scratch/ExplodedDosBundleManagement"
  "headers": [
    "Authorization : Bearer IxXb42lajFdNsAr"
  ]
}

iRODS Ticket is generated on demand, can be shared.
Next Steps

- Complete packaging and unit tests
- Validation with GA4GH
- Incorporate the ability to attach descriptions to bundles and data objects
- Beta release
- Implement https download access as first service in new irods-rest REST API revision
- Possible command line tool or rule set:
  - CRUD on bundles
  - Rules enforcing optional immutability?
- Possible ‘quick download’ util that can download irods:// URIs via high speed transfer
What iRODS needs!

- Focus on i/o performance of streaming.
- Standard way of computing MIME type (via extension inspection or optional file content scanning) and storing computed MIME type for subsequent query.
- Possible iCommand support for irods:// URI download
- Work with GA4GH to put iRODS semantics into the mix in DRS, add to CI.
- Standard notion of a file ‘Description’, is it the ‘comment’? Is it a standard AVU?
- Mark as ‘immutable’ at collection level?
Thank You!

Mike Conway
NIH/NIEHS
Office of Data Science

https://www.niehs.nih.gov/research/atniehs/dntp/osim/index.cfm

mike.conway@nih.gov

GitHub:
https://github.com/michael-conway/irods-ga4gh-dos