Data Repository Service

Table of Contents

1. Overview . . . 1
   1.1. Version information . . . 1
   1.2. Contact information . . . 1
   1.3. License information . . . 1
   1.4. URI scheme . . . 1
   1.5. Consumes . . . 1
   1.6. Produces . . . 1
2. Introduction . . . 2
3. DRS API Principles . . . 3
   3.1. DRS IDs . . . 3
   3.2. DRS Datatypes . . . 3
   3.3. Read-only . . . 3
   3.4. URI convention (WORK IN PROGRESS) . . . 3
   3.5. Standards . . . 3
4. Authorization & Authentication (WORK IN PROGRESS) . . . 4
5. Paths . . . 5
   5.1. Create a new Data Bundle . . . 5
      5.1.1. Parameters . . . 5
      5.1.2. Responses . . . 5
      5.1.3. Tags . . . 5
   5.2. List the Data Bundles . . . 5
      5.2.1. Parameters . . . 5
      5.2.2. Responses . . . 6
      5.2.3. Tags . . . 6
   5.3. Retrieve a Data Bundle . . . 6
      5.3.1. Parameters . . . 6
      5.3.2. Responses . . . 7
      5.3.3. Tags . . . 7
   5.4. Update a Data Bundle . . . 7
      5.4.1. Parameters . . . 7
      5.4.2. Responses . . . 7
      5.4.3. Tags . . . 8
   5.5. Delete a Data Bundle . . . 8
      5.5.1. Parameters . . . 8
      5.5.2. Responses . . . 8
      5.5.3. Tags . . . 8
   5.6. Retrieve all versions of a Data Bundle . . . 8
      5.6.1. Parameters . . . 8

5.6.2. Responses . . . 9
5.6.3. Tags . . . 9
5.7. Make a new Data Object . . . 9
5.7.1. Parameters . . . 9
5.7.2. Responses . . . 9
5.7.3. Tags . . . 9
5.8. List the Data Objects . . . 10
5.8.1. Parameters . . . 10
5.8.2. Responses . . . 10
5.8.3. Tags . . . 11
5.9. Retrieve a Data Object . . . 11
5.9.1. Parameters . . . 11
5.9.2. Responses . . . 11
5.9.3. Tags . . . 11
5.10. Update a Data Object . . . 11
5.10.1. Parameters . . . 11
5.10.2. Responses . . . 12
5.10.3. Tags . . . 12
5.11. Delete a Data Object index entry . . . 12
5.11.1. Parameters . . . 12
5.11.2. Responses . . . 12
5.11.3. Tags . . . 13
5.12. Retrieve all versions of a Data Object . . . 13
5.12.1. Parameters . . . 13
5.12.2. Responses . . . 13
5.12.3. Tags . . . 13
5.13. Returns service version and other information . . . 13
5.13.1. Responses . . . 13
5.13.2. Tags . . . 14
6. Definitions . . . 15
6.1. AuthorizationMetadata . . . 15
6.2. Bundle . . . 15
6.3. Checksum . . . 16
6.4. CreateBundleRequest . . . 16
6.5. CreateBundleResponse . . . 16
6.6. CreateObjectRequest . . . 16
6.7. CreateObjectResponse . . . 17
6.8. DeleteBundleResponse . . . 17
6.9. DeleteObjectResponse . . . 17
6.10. ErrorResponse . . . 17
6.11. GetBundleResponse . . . 17

6.12. GetBundleVersionsResponse . . . 17
6.13. GetObjectResponse . . . 18
6.14. GetObjectVersionsResponse . . . 18
6.15. ListBundlesRequest . . . 18
6.16. ListBundlesResponse . . . 19
6.17. ListObjectsRequest . . . 19
6.18. ListObjectsResponse . . . 20
6.19. Object . . . 20
6.20. ServiceInfoResponse . . . 21
6.21. SystemMetadata . . . 21
6.22. URL . . . 21
6.23. UpdateBundleRequest . . . 22
6.24. UpdateBundleResponse . . . 22
6.25. UpdateObjectRequest . . . 22
6.26. UpdateObjectResponse . . . 22
6.27. UserMetadata . . . 22
7. Appendix: Motivation . . . 23
7.1. Federation . . . 24

Chapter 1. Overview

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

1.1. Version information

Version : 0.0.1

1.2. Contact information

Contact : GA4GH Cloud Work Stream
Contact Email : ga4gh-cloud@ga4gh.org

1.3. License information

License : Apache 2.0
License URL : https://raw.githubusercontent.com/ga4gh/data-repository-service-schemas/master/LICENSE
Terms of service : null

1.4. URI scheme

BasePath : /ga4gh/drs/v1
Schemes : HTTPS, HTTP

1.5. Consumes

- `application/json`

1.6. Produces

- `application/json`

Chapter 2. Introduction

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. This document describes the DRS API and provides details on the specific endpoints, request formats, and response. It is intended for developers of DRS-compatible services and of clients that will call these DRS services.

The primary functionality of DRS is to map a logical ID to a means for physically retrieving the data represented by the ID. The sections below describe the characteristics of those IDs, the types of data supported, and how the mapping works.

NOTE: this document represents a work in progress towards DRS 1.0.0. It may not be fully in sync with the OpenAPI schema since both are being worked on. The 0.0.1 release represents the schema as it existed at the time of the transition from the DOS to DRS name and is subject to change as we evolve it to a DRS 1.0.0.

Chapter 3. DRS API Principles

3.1. DRS IDs

Each implementation of DRS can choose its own id scheme, as long as it follows these guidelines:

- DRS IDs are URL-safe text strings made up of alphanumeric characters and any of [.-_/]
- One DRS ID MUST always return the same object data (or, in the case of a collection, the same set of objects). This constraint aids with reproducibility.
- DRS does NOT support semantics around multiple versions of an object. (For example, there's no notion of "get latest version" or "list all versions" in DRS v1.) Individual implementation MAY choose an ID scheme that includes version hints.
- DRS implementations MAY have more than one ID that maps to the same object.

3.2. DRS Datatypes

DRS v1 supports two datatypes:

- Blobs — these are file-like objects
- Collections — these are sets of other DRS objects (either Blobs or Collections)

3.3. Read-only

DRS v1 is a read-only API. We expect that each implementation will define its own mechanisms and interfaces (graphical and/or programmatic) for adding and updating data.

3.4. URI convention (WORK IN PROGRESS)

For convenience, we define a recommended syntax for fully referencing DRS-accessible objects. Strings of the form drs://<server>/<id> mean "make a DRS call to the HTTP address at <server>, passing in the DRS id <id>, to retrieve the object". For example, these strings are useful when passing objects to a WES server for processing.

3.5. Standards

The DRS API specification is written in OpenAPI and embodies a RESTful service philosophy. It uses JSON in requests and responses and standard HTTP/HTTPS for information transport.

Chapter 4. Authorization & Authentication (WORK IN PROGRESS)

Users must supply credentials that establish their identity and authorization in order to use a DRS endpoint. We recommend that DRS implementations use an OAuth2 bearer token, although they can choose other mechanisms if appropriate. DRS callers can use the `auth_instructions_url` from the service-info endpoint to learn how to obtain and use a bearer token for a particular implementation.

The DRS implementation is responsible for checking that a user is authorized to submit requests. The particular authorization policy is up to the DRS implementer.

Chapter 5. Paths

5.1. Create a new Data Bundle

```
POST /bundles
```

5.1.1. Parameters

Type	Name	Schema
Body	**body** *required*	CreateBundleRequest

5.1.2. Responses

HTTP Code	Description	Schema
200	The Data Bundle was successfully created.	CreateBundleResponse
400	The request is malformed.	ErrorResponse
401	The request is unauthorized.	ErrorResponse
403	The requester is not authorized to perform this action.	ErrorResponse
500	An unexpected error occurred.	ErrorResponse

5.1.3. Tags

- DataRepositoryService

5.2. List the Data Bundles

```
GET /bundles
```

5.2.1. Parameters

Type	Name	Description	Schema
Query	**alias** *optional*	If provided returns Data Bundles that have any alias that matches the request.	string
Query	**checksum** *optional*	The hexlified checksum that one would like to match on.	string

Type	Name	Description	Schema
Query	**checksum_type** *optional*	If provided will restrict responses to those that match the provided type. possible values: md5 # most blob stores provide a checksum using this multipart-md5 # multipart uploads provide a specialized tag in S3 sha256 sha512	string
Query	**page_size** *optional*	Specifies the maximum number of results to return in a single page. If unspecified, a system default will be used.	integer (int32)
Query	**page_token** *optional*	The continuation token, which is used to page through large result sets. To get the next page of results, set this parameter to the value of `next_page_token` from the previous response.	string

5.2.2. Responses

HTTP Code	Description	Schema
200	Successfully listed Data Bundles.	ListBundlesResponse
400	The request is malformed.	ErrorResponse
401	The request is unauthorized.	ErrorResponse
403	The requester is not authorized to perform this action.	ErrorResponse
500	An unexpected error occurred.	ErrorResponse

5.2.3. Tags

- DataRepositoryService

5.3. Retrieve a Data Bundle

```
GET /bundles/{bundle_id}
```

5.3.1. Parameters

Type	Name	Description	Schema
Path	**bundle_id** *required*		string

Type	Name	Description	Schema
Query	**version** *optional*	If provided will return the requested version of the selected Data Bundle. Otherwise, only the latest version is returned.	string

5.3.2. Responses

HTTP Code	Description	Schema
200	Successfully found the Data Bundle.	GetBundleResponse
400	The request is malformed.	ErrorResponse
401	The request is unauthorized.	ErrorResponse
403	The requester is not authorized to perform this action.	ErrorResponse
404	The requested Data Bundle wasn't found.	ErrorResponse
500	An unexpected error occurred.	ErrorResponse

5.3.3. Tags

- DataRepositoryService

5.4. Update a Data Bundle

```
PUT /bundles/{bundle_id}
```

5.4.1. Parameters

Type	Name	Description	Schema
Path	**bundle_id** *required*	The ID of the Data Bundle to update	string
Body	**body** *required*	The new content for the Data Bundle identified by the given bundle_id. If the ID specified in the request body is different than that specified in the path, the Data Bundle's ID will be replaced with the one in the request body.	UpdateBundleRequest

5.4.2. Responses

HTTP Code	Description	Schema
200	The Data Bundle was updated successfully.	UpdateBundleResponse
400	The request is malformed.	ErrorResponse

HTTP Code	Description	Schema
401	The request is unauthorized.	ErrorResponse
403	The requester is not authorized to perform this action.	ErrorResponse
404	The requested Data Bundle wasn't found.	ErrorResponse
500	An unexpected error occurred.	ErrorResponse

5.4.3. Tags

- DataRepositoryService

5.5. Delete a Data Bundle

```
DELETE /bundles/{bundle_id}
```

5.5.1. Parameters

Type	Name	Schema
Path	**bundle_id** *required*	string

5.5.2. Responses

HTTP Code	Schema
200	DeleteBundleResponse

5.5.3. Tags

- DataRepositoryService

5.6. Retrieve all versions of a Data Bundle

```
GET /bundles/{bundle_id}/versions
```

5.6.1. Parameters

Type	Name	Schema
Path	**bundle_id** *required*	string

5.6.2. Responses

HTTP Code	Description	Schema
200	The versions for the Data Bundle were found successfully.	GetBundleVersionsResponse
400	The request is malformed.	ErrorResponse
401	The request is unauthorized.	ErrorResponse
403	The requester is not authorized to perform this action.	ErrorResponse
404	The requested Data Bundle wasn't found.	ErrorResponse
500	An unexpected error occurred.	ErrorResponse

5.6.3. Tags

- DataRepositoryService

5.7. Make a new Data Object

```
POST /objects
```

5.7.1. Parameters

Type	Name	Description	Schema
Body	**body** *required*	The Data Object to be created. The ID scheme is left up to the implementor but should be unique to the server instance.	CreateObjectRequest

5.7.2. Responses

HTTP Code	Description	Schema
200	Successfully created the Data Object.	CreateObjectResponse
400	The request is malformed.	ErrorResponse
401	The request is unauthorized.	ErrorResponse
403	The requester is not authorized to perform this action.	ErrorResponse
500	An unexpected error occurred.	ErrorResponse

5.7.3. Tags

- DataRepositoryService

5.8. List the Data Objects

```
GET /objects
```

5.8.1. Parameters

Type	Name	Description	Schema
Query	**alias** *optional*	If provided will only return Data Objects with the given alias.	string
Query	**checksum** *optional*	The hexlified checksum that one would like to match on.	string
Query	**checksum_type** *optional*	If provided will restrict responses to those that match the provided type. possible values: md5 # most blob stores provide a checksum using this multipart-md5 # multipart uploads provide a specialized tag in S3 sha256 sha512	string
Query	**page_size** *optional*	Specifies the maximum number of results to return in a single page. If unspecified, a system default will be used.	integer (int32)
Query	**page_token** *optional*	The continuation token, which is used to page through large result sets. To get the next page of results, set this parameter to the value of `next_page_token` from the previous response.	string
Query	**url** *optional*	If provided will return only Data Objects with a that URL matches this string.	string

5.8.2. Responses

HTTP Code	Description	Schema
200	The Data Objects were listed successfully.	ListObjectsResponse
400	The request is malformed.	ErrorResponse
401	The request is unauthorized.	ErrorResponse
403	The requester is not authorized to perform this action.	ErrorResponse
500	An unexpected error occurred.	ErrorResponse

5.8.3. Tags

- DataRepositoryService

5.9. Retrieve a Data Object

```
GET /objects/{object_id}
```

5.9.1. Parameters

Type	Name	Description	Schema
Path	**object_id** *required*		string
Query	**version** *optional*	If provided will return the requested version of the selected Data Object.	string

5.9.2. Responses

HTTP Code	Description	Schema
200	The Data Object was found successfully.	GetObjectResponse
400	The request is malformed.	ErrorResponse
401	The request is unauthorized.	ErrorResponse
403	The requester is not authorized to perform this action.	ErrorResponse
404	The requested Data Object wasn't found	ErrorResponse
500	An unexpected error occurred.	ErrorResponse

5.9.3. Tags

- DataRepositoryService

5.10. Update a Data Object

```
PUT /objects/{object_id}
```

5.10.1. Parameters

Type	Name	Description	Schema
Path	**object_id** *required*	The ID of the Data Object to update	string

Type	Name	Description	Schema
Body	**body** *required*	The new Data Object for the given object_id. If the ID specified in the request body is different than that specified in the path, the Data Object's ID will be replaced with the one in the request body.	UpdateObjectRequest

5.10.2. Responses

HTTP Code	Description	Schema
200	The Data Object was successfully updated.	UpdateObjectResponse
400	The request is malformed.	ErrorResponse
401	The request is unauthorized.	ErrorResponse
403	The requester is not authorized to perform this action.	ErrorResponse
404	The requested Data Object wasn't found.	ErrorResponse
500	An unexpected error occurred.	ErrorResponse

5.10.3. Tags

- DataRepositoryService

5.11. Delete a Data Object index entry

```
DELETE /objects/{object_id}
```

5.11.1. Parameters

Type	Name	Schema
Path	**object_id** *required*	string

5.11.2. Responses

HTTP Code	Description	Schema
200	The Data Object was deleted successfully.	DeleteObjectResponse
400	The request is malformed.	ErrorResponse
401	The request is unauthorized.	ErrorResponse
403	The requester is not authorized to perform this action.	ErrorResponse

HTTP Code	Description	Schema
404	The requested Data Object wasn't found.	ErrorResponse
500	An unexpected error occurred.	ErrorResponse

5.11.3. Tags

- DataRepositoryService

5.12. Retrieve all versions of a Data Object

```
GET /objects/{object_id}/versions
```

5.12.1. Parameters

Type	Name	Schema
Path	**object_id** *required*	string

5.12.2. Responses

HTTP Code	Description	Schema
200	The versions for the Data Object were returned successfully.	GetObjectVersions Response
400	The request is malformed.	ErrorResponse
401	The request is unauthorized.	ErrorResponse
403	The requester is not authorized to perform this action.	ErrorResponse
404	The requested Data Object wasn't found.	ErrorResponse
500	An unexpected error occurred.	ErrorResponse

5.12.3. Tags

- DataRepositoryService

5.13. Returns service version and other information

```
GET /service-info
```

5.13.1. Responses

HTTP Code	Description	Schema
200	Service information returned successfully	ServiceInfoResponse

5.13.2. Tags

- DataRepositoryService

Chapter 6. Definitions

6.1. AuthorizationMetadata

OPTIONAL
A set of key-value pairs that represent sufficient metadata to be granted access to a resource. It may be helpful to provide details about a specific provider, for example.

Name	Description	Schema
auth_type *optional*	The auth standard being used to make data available. For example, 'OAuth2.0'.	string
auth_url *optional*	The URL where the auth service is located, for example, a URL to get an OAuth token.	string

6.2. Bundle

Name	Description	Schema
aliases *optional*	A list of strings that can be used to identify this Data Bundle.	< string > array
checksums *required*	At least one checksum must be provided. The Data Bundle checksum is computed over all the checksums of the Data Objects that bundle contains.	< Checksum > array
created *required*	Timestamp of object creation in RFC3339.	string (date-time)
description *optional*	A human readable description.	string
id *required*	An identifier, unique to this Data Bundle	string
object_ids *required*	The list of Data Objects that this Data Bundle contains.	< string > array
system_metadata *optional*		SystemMetadata
updated *required*	Timestamp of update in RFC3339, identical to create timestamp in systems that do not support updates.	string (date-time)
user_metadata *optional*		UserMetadata

Name	Description	Schema
version *required*	A string representing a version, some systems may use checksum, a RFC3339 timestamp, or incrementing version number. For systems that do not support versioning please use your update timestamp as your version.	string

6.3. Checksum

Name	Description	Schema
checksum *required*	The hex-string encoded checksum for the Data.	string
type *optional*	The digest method used to create the checksum. If left unspecified md5 will be assumed. possible values: md5 # most blob stores provide a checksum using this multipart-md5 # multipart uploads provide a specialized tag in S3 sha256 sha512	string

6.4. CreateBundleRequest

Name	Schema
bundle *optional*	Bundle

6.5. CreateBundleResponse

Name	Description	Schema
bundle_id *required*	The identifier of the Data Bundle created.	string

6.6. CreateObjectRequest

The Data Object one would like to index. One must provide any aliases and URLs to this file when sending the CreateObjectRequest. It is up to implementations to validate that the Data Object is available from the provided URLs.

Name	Schema
object *required*	Object

6.7. CreateObjectResponse

Name	Description	Schema
object_id *optional*	The ID of the created Data Object.	string

6.8. DeleteBundleResponse

Name	Schema
bundle_id *optional*	string

6.9. DeleteObjectResponse

Name	Description	Schema
object_id *required*	The identifier of the Data Object deleted.	string

6.10. ErrorResponse

An object that can optionally include information about the error.

Name	Description	Schema
msg *optional*	A detailed error message.	string
status_code *optional*	The integer representing the HTTP status code (e.g. 200, 404).	integer

6.11. GetBundleResponse

Name	Schema
bundle *optional*	Bundle

6.12. GetBundleVersionsResponse

Name	Description	Schema
bundles *required*	All versions of the Data Bundles that match the GetBundleVersions request.	< Bundle > array

6.13. GetObjectResponse

Name	Schema
object *required*	Object

6.14. GetObjectVersionsResponse

Name	Description	Schema
objects *required*	All versions of the Data Objects that match the GetObjectVersions request.	< Object > array

6.15. ListBundlesRequest

Only return Data Bundles that match all of the request parameters. A page_size and page_token are provided for retrieving a large number of results.

Name	Description	Schema
alias *optional*	If provided returns Data Bundles that have any alias that matches the request.	string
checksum *optional*	The hexlified checksum that one would like to match on.	string
checksum_type *optional*	If provided will restrict responses to those that match the provided type. possible values: md5 # most blob stores provide a checksum using this multipart-md5 # multipart uploads provide a specialized tag in S3 sha256 sha512	string
page_size *optional*	Specifies the maximum number of results to return in a single page. If unspecified, a system default will be used.	integer (int32)

Name	Description	Schema
page_token *optional*	The continuation token, which is used to page through large result sets. To get the next page of results, set this parameter to the value of `next_page_token` from the previous response.	string

6.16. ListBundlesResponse

A list of Data Bundles matching the request parameters and a continuation token that can be used to retrieve more results.

Name	Description	Schema
bundles *optional*	The list of Data Bundles.	< Bundle > array
next_page_token *optional*	The continuation token, which is used to page through large result sets. Provide this value in a subsequent request to return the next page of results. This field will be empty if there aren't any additional results.	string

6.17. ListObjectsRequest

Allows a requester to list and filter Data Objects. Only Data Objects matching all of the requested parameters will be returned.

Name	Description	Schema
alias *optional*	If provided will only return Data Objects with the given alias.	string
checksum *optional*	The hexlified checksum that one would like to match on.	string
checksum_type *optional*	If provided will restrict responses to those that match the provided type. possible values: md5 # most blob stores provide a checksum using this multipart-md5 # multipart uploads provide a specialized tag in S3 sha256 sha512	string
page_size *optional*	Specifies the maximum number of results to return in a single page. If unspecified, a system default will be used.	integer (int32)

Name	Description	Schema
page_token *optional*	The continuation token, which is used to page through large result sets. To get the next page of results, set this parameter to the value of `next_page_token` from the previous response.	string
url *optional*	If provided will return only Data Objects with a that URL matches this string.	string

6.18. ListObjectsResponse

A list of Data Objects matching the requested parameters, and a paging token, that can be used to retrieve more results.

Name	Description	Schema
next_page_token *optional*	The continuation token, which is used to page through large result sets. Provide this value in a subsequent request to return the next page of results. This field will be empty if there aren't any additional results.	string
objects *optional*	The list of Data Objects.	< Object > array

6.19. Object

Name	Description	Schema
aliases *optional*	A list of strings that can be used to find this Data Object. These aliases can be used to represent the Data Object's location in a directory (e.g. "bucket/folder/file.name") to make Data Objects more discoverable. They might also be used to represent	< string > array
checksums *required*	The checksum of the Data Object. At least one checksum must be provided.	< Checksum > array
created *required*	Timestamp of object creation in RFC3339.	string (date-time)
description *optional*	A human readable description of the contents of the Data Object.	string
id *required*	An identifier unique to this Data Object.	string
mime_type *optional*	A string providing the mime-type of the Data Object. For example, "application/json".	string

Name	Description	Schema
name *optional*	A string that can be optionally used to name a Data Object.	string
size *required*	The computed size in bytes.	string (int64)
updated *optional*	Timestamp of update in RFC3339, identical to create timestamp in systems that do not support updates.	string (date-time)
urls *optional*	The list of URLs that can be used to access the Data Object.	< URL > array
version *optional*	A string representing a version.	string

6.20. ServiceInfoResponse

Placeholder for the Info Object

Name	Description	Schema
contact *optional*	Maintainer contact info	object
description *optional*	Service description	string
license *optional*	License information for the exposed API	object
title *optional*	Service name	string
version *required*	Service version	string

6.21. SystemMetadata

OPTIONAL
These values are reported by the underlying object store.
A set of key-value pairs that represent system metadata about the object.

Type : object

6.22. URL

Name	Description	Schema
authorization_metadata *optional*		AuthorizationMetadata

Name	Description	Schema
system_metadata *optional*		SystemMetadata
url *required*	A URL that can be used to access the file.	string
user_metadata *optional*		UserMetadata

6.23. UpdateBundleRequest

Name	Schema
bundle *required*	Bundle

6.24. UpdateBundleResponse

Name	Description	Schema
bundle_id *required*	The identifier of the Data Bundle updated.	string

6.25. UpdateObjectRequest

Name	Schema
object *required*	Object

6.26. UpdateObjectResponse

Name	Description	Schema
object_id *required*	The identifier of the Data Object updated.	string

6.27. UserMetadata

OPTIONAL
A set of key-value pairs that represent metadata provided by the uploader.

Type : object

Chapter 7. Appendix: Motivation

Data sharing requires portable data, consistent with the FAIR data principles (findable, accessible, interoperable, reusable). Today's researchers and clinicians are surrounded by potentially useful data, but often need bespoke tools and processes to work with each dataset. And today's data publishers don't have a reliable way to make their data useful to all (and only) the people they choose.

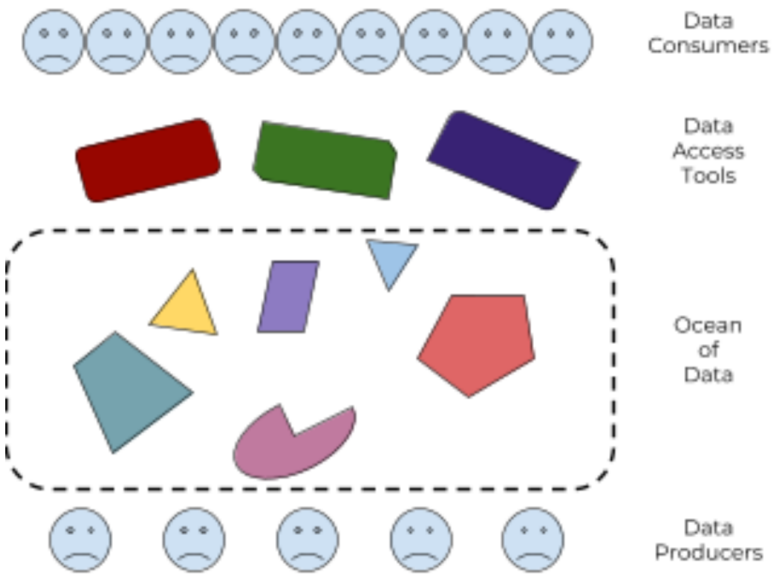

Figure 1: there's an ocean of data, with many different tools to drink from it, but no guarantee that any tool will work with any subset of the data

We need a standard way for data producers to make their data available to data consumers, that supports the control needs of the former and the access needs of the latter. And we need it to be interoperable, so anyone who builds access tools and systems can be confident they'll work with all the data out there, and anyone who publishes data can be confident it will work with all the tools out there.

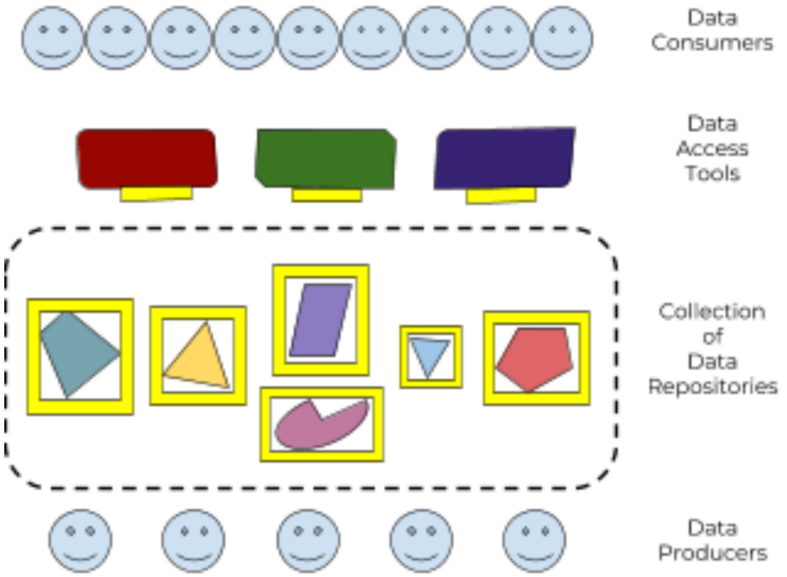

Figure 2: by defining a standard Data Repository API, and adapting tools to use it, every data publisher can now make their data useful to every data consumer

We envision a world where:

- there are many many **data consumers**, working in research and in care, who can use the tools of their choice to access any all data that they have permission to see
- there are many **data access tools** and platforms, supporting discovery, visualization, analysis, and collaboration
- there are many **data repositories**, each with their own policies and characteristics, which can be accessed by a variety of tools
- there are many **data publishing tools** and platforms, supporting a variety of data lifecycles and formats
- there are many many **data producers**, generating data of all types, who can use the tools of their choice to make their data as widely available as is appropriate

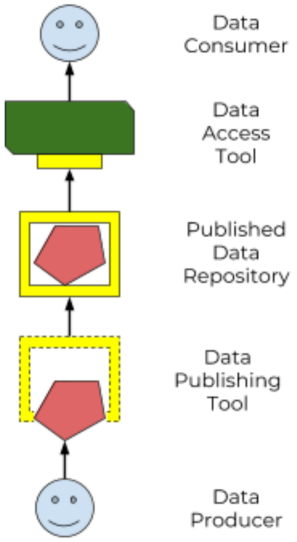

Figure 3: a standard Data Repository API enables an ecosystem of data producers and consumers

This spec defines a standard **Data Repository Service (DRS) API** (“the yellow box”), to enable that ecosystem of data producers and consumers. Our goal is that all data consumers need to know about a data repo is "here’s the DRS endpoint to access it", and all data publishers need to know about tapping into the world of consumption tools is "here’s how to tell it where my DRS endpoint lives".

7.1. Federation

The world’s biomedical data is controlled by groups with very different policies and restrictions on where their data lives and how it can be accessed. A primary purpose of DRS is to support unified access to disparate and distributed data. (As opposed to the alternative centralized model of "let’s just bring all the data into one single data repository”, which would be technically easier but is no more realistic than “let’s just bring all the websites into one single web host”.)

In a DRS-enabled world, tool builders don’t have to worry about where the data their tools operate on lives — they can count on DRS to give them access. And tool users only need to know which DRS server is managing the data they need, and whether they have permissions; they don’t have to worry about how to physically get access to, or (worse) make a copy of the data. For example, if I have appropriate permissions, I can run a pooled analysis where I run a single tool across data managed by different DRS servers, potentially in different locations.