

Registry Considerations for Manifest Files:
Configurable Data Curation System (CDCS) and FAIR Containers
Benjamin Long

CDCS Team: Guillaume Sousa Amaral Philippe Dessauw, Hamza Bouhanni

Introduction

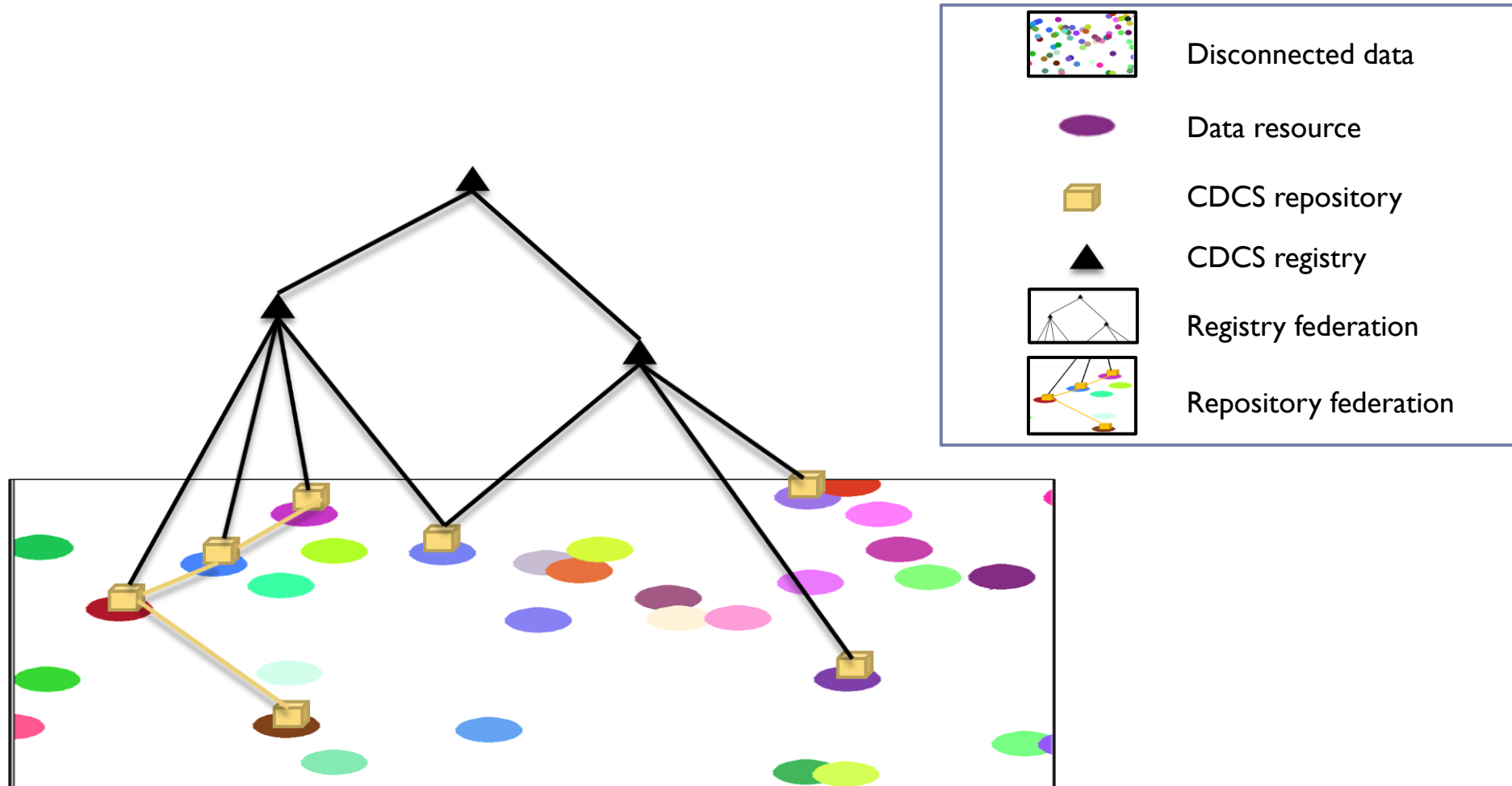
- ▶ **Will discuss**
 - ▶ an informatics platform: Configurable Data Curation System (CDCS)
 - ▶ that may be relevant to your context: interoperable, FAIR containers
 - brief contextual intro
- ▶ **Will show**
 - ▶ some examples
 - ▶ that demonstrate its ability to support this concept in common formats (XML, JSON), with relevant API automation
- ▶ **Will consider**
 - ▶ how this might fit into use-cases such as
 - ▶ UI support,
 - ▶ automation,
 - ▶ file-format usage and translation
 - ▶ even touching a little on consensus specification and evolution

Configurable Data Curation System (CDCS)

- ▶ **Informatics infrastructure / eco-system**
 - ▶ Developed at NIST
 - ▶ For Materials Genome Initiative (MGI) – original context: materials science
 - ▶ For input, output, transformation, search for, and storage of
 - ▶ information structures (data, their abstractions, associated files/resources)
 - ▶ Supports FAIR (findable, accessible, interoperable, reusable) data
 - ▶ Infrastructure for storage (repositories) and search/discovery (registries)
- ▶ **Will discuss**
 - ▶ Your context: FAIR containers/manifests
 - ▶ How this fits, potentially, into that context + those goals

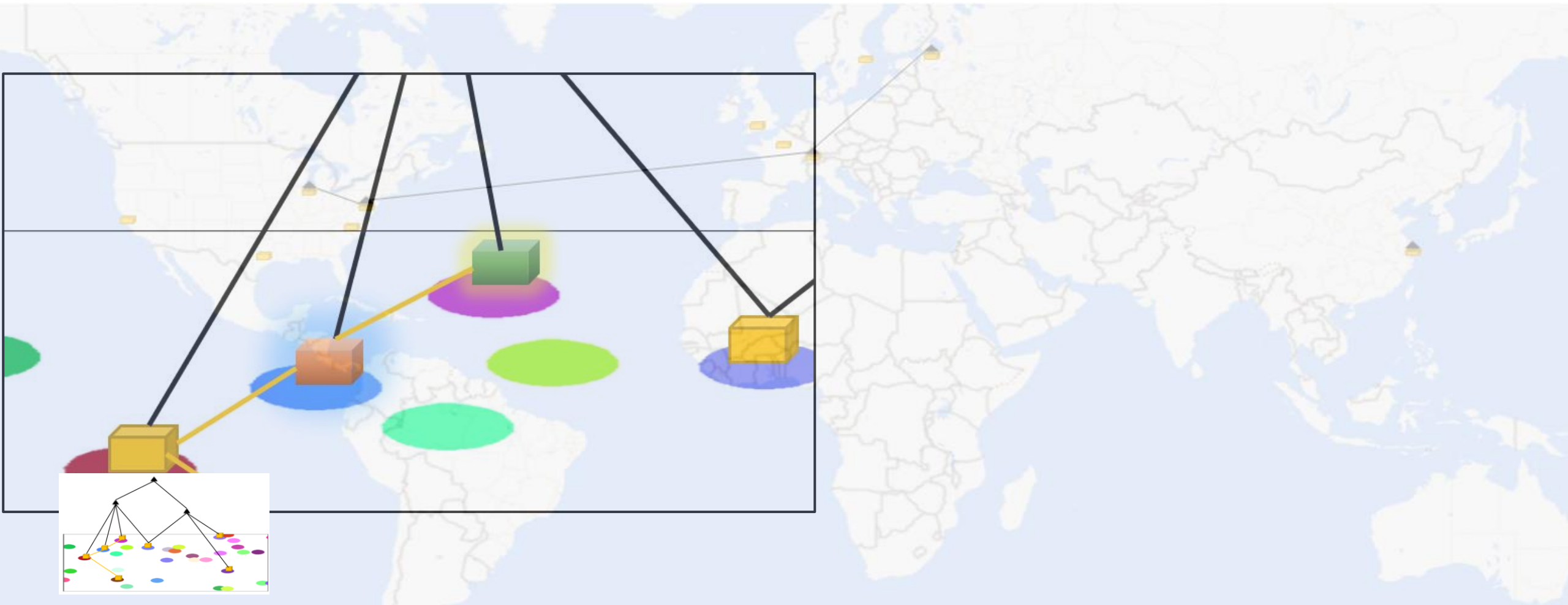
Organizing and connecting resources

(Question: How to organize, connect, and coordinate an eco-system of manifests and specs over time?)



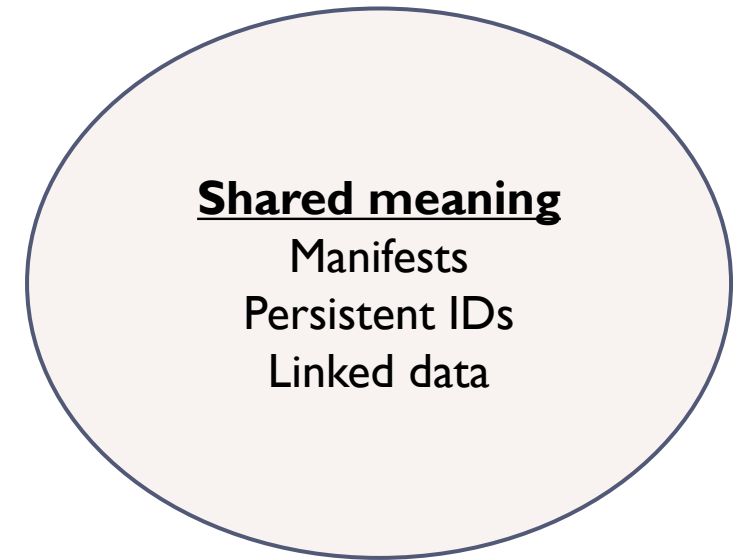
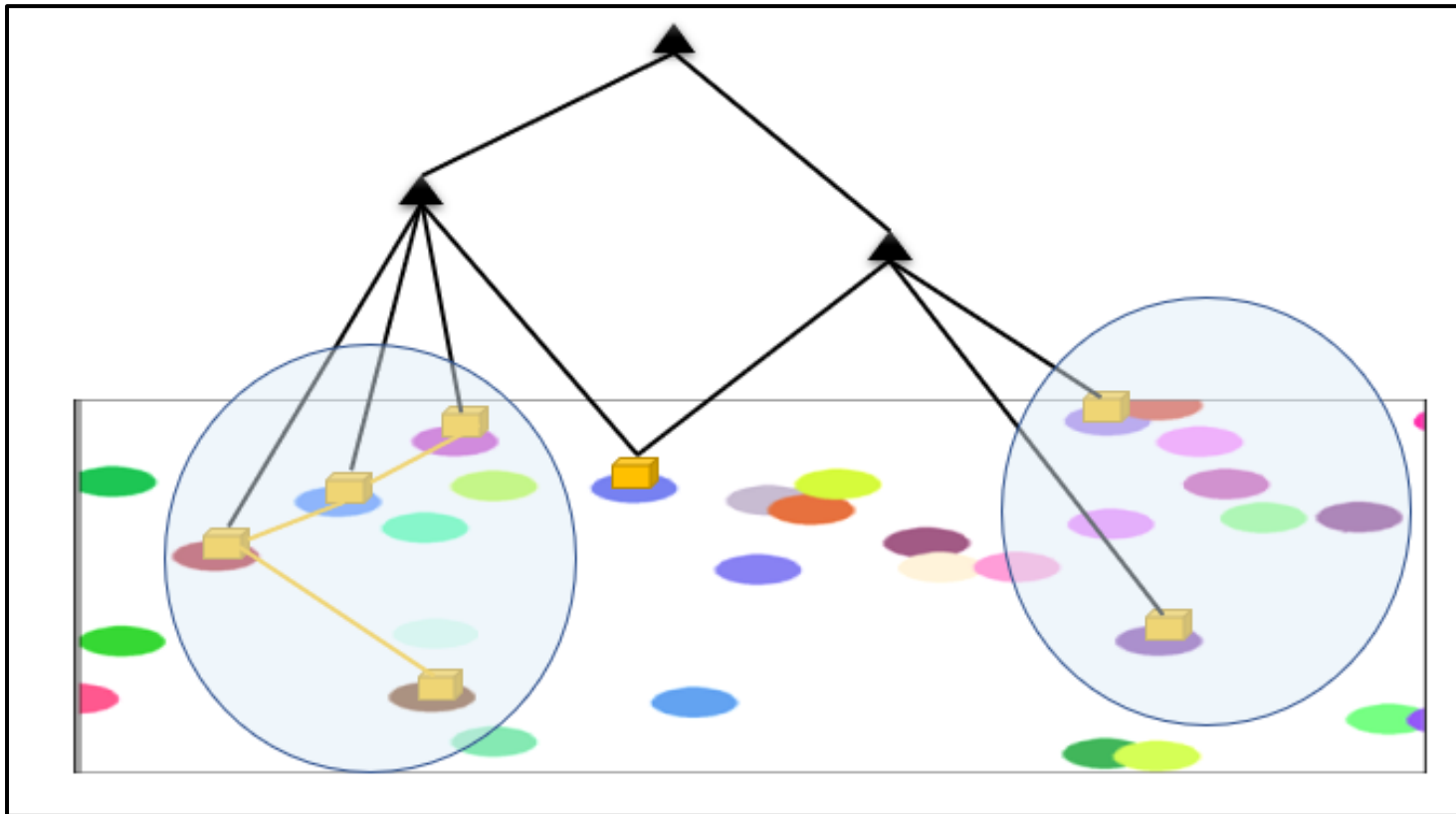
Strategy – customize individually, connect globally

(Question: How to support rapidly growing, diverse groups, communities and their outputs in a unified way where all can benefit?)

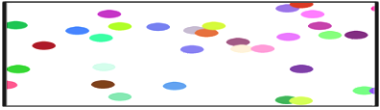


Strategy – organize communities of meaning

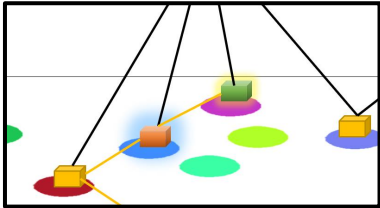
(Question: Will you need to do analysis and informatics on your manifests and containers themselves?)



The CDCS Process

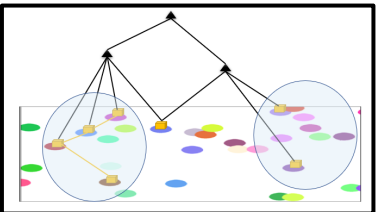


Meet with users and communities about their resources and data



Start structuring and sharing data one resource, one project, or one community at a time

- Customize online containers for structured data
- Create repositories of data (in XML and JSON format)
- Convert data into different formats as needed
- Customize how it is presented, displayed, and searched
- Share and use it in a FAIR manner

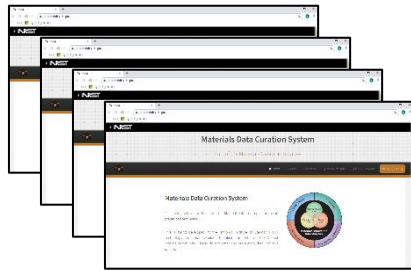


Start linking together data into a more global infrastructure

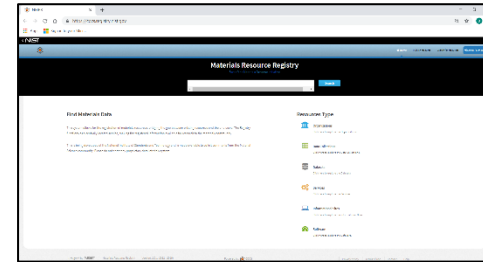
- Connecting repositories together
- Creating and connecting registries together
- Creating and connecting data via linking (persistent IDs, handle services, REST APIs)

Configurable Data Curation System (CDCS)

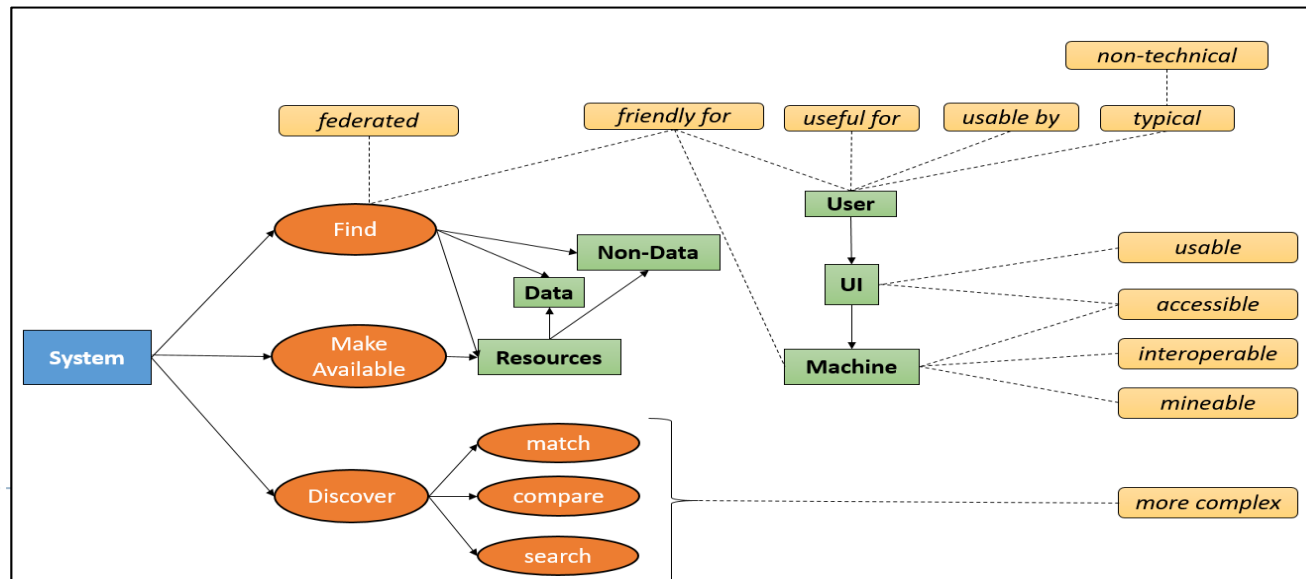
Curators (Repositories)



Registry (Search)

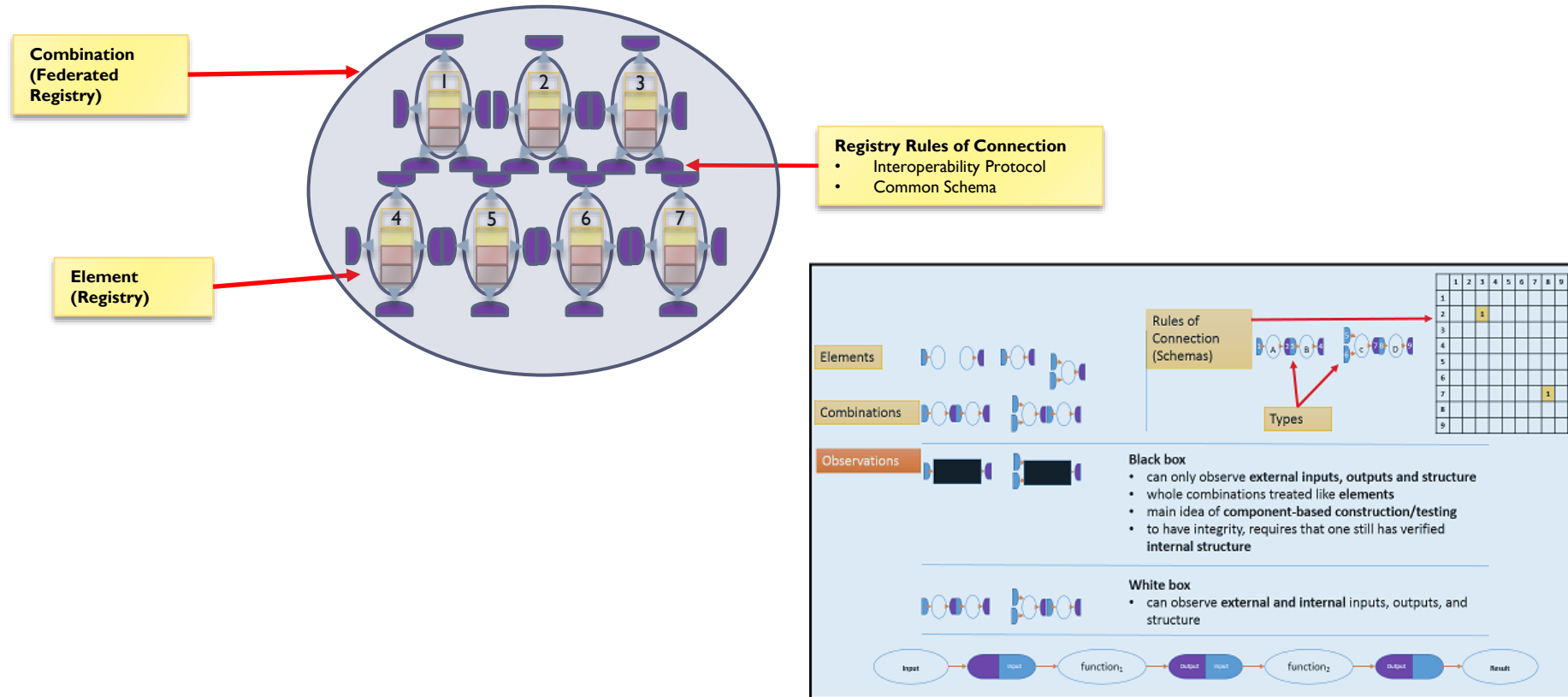


Function	Description
General tasks	General user tasks such as requesting an account, logging in, etc.
Structuring data	Creating data entry templates with the composer application.
Entering data	Inputting data using data entry templates with the curator application.
Accessing data	Controlling data access via creation and use of users, groups, and workspaces.
Finding data	Searching for data locally and remotely.
Transforming data	Transforming data from one format or representation to another.
Managing resources	Managing resources and configurations on the curator platform.
Operations	Operational tasks such as deployment, creating custom themes, etc.



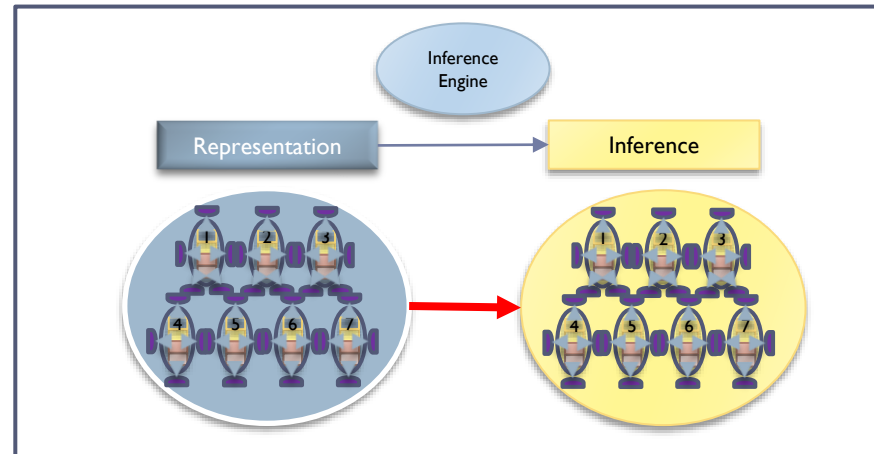
Federated Registries

(Question: How do you anticipate distributing the load of tracking, coordinating evolution in manifests and specs across a potentially global ecosystem like this one?)



Reason About Knowledge of Entire Domain

(Question: What knowledge regarding your manifests, containers, specifications, and eco-system would you like to reason about?)



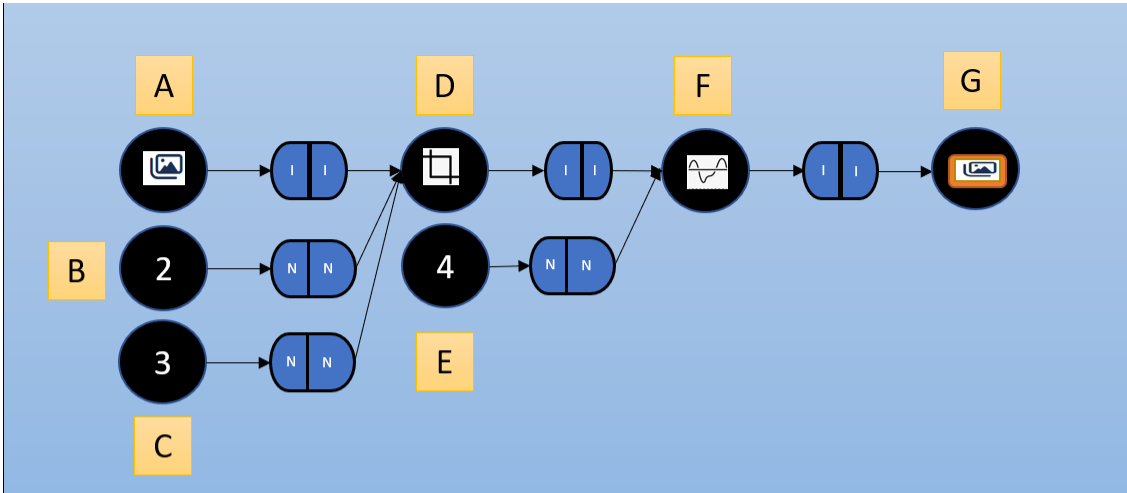
OVERALL GOAL:

**USE FEDERATION TO ACCESS AND REASON
ABOUT ALL GLOBAL, CURATED, KNOWLEDGE
OF A DOMAIN**

How CDCS could possibly support FAIR manifests and containers now

- ▶ Define/manage component eco-system (such as through GitHub, DockerHub, manifest-files)
 - ▶ Define manifest files in some supported format (XML, JSON)
 - ▶ Register manifest files in CDCS system (registry) with PID
 - ▶ Translate out in whatever formats (exporters and exporter extensions for translation)
- ▶ Automate processes for finding, manipulating, editing components and component-structures via CDCS registry REST API
- ▶ Have tools, services, processes, UIs, workflows – call this API to interact with the registry (or services that interact with the registry)
- ▶ Organize registry with several types, PIDs, and interlinkages
 - ▶ Manifests
 - ▶ Specification versions, evolution

Container manifest workflow representation example



<https://github.com/usnistgov/fair-chain-compute-container>

Manifest configuration

manifest	name	Type	value	inputs	outputs	role
A	i-imgs	var		-	I (images)	Source
B	xDim	var	2	-	N (numeric)	Source
C	yDim	var	3	-	N	Source
D		function		I, N, N	I	Process
E	threshold	var	4	-	N	Source
F		function		I, N	I	Process
G	o-imgs	var		I	-	Result

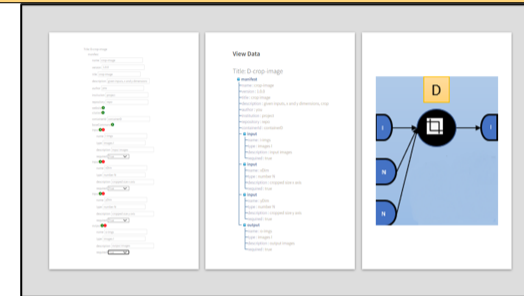
D: Crop-image

Inputs/ui:

- A: i-imgs: input-images
- B: xDim: Cropped-size-x-axis
- C: yDim: cropped-size-y-axis

Outputs:

- D: o-imgs: output-images



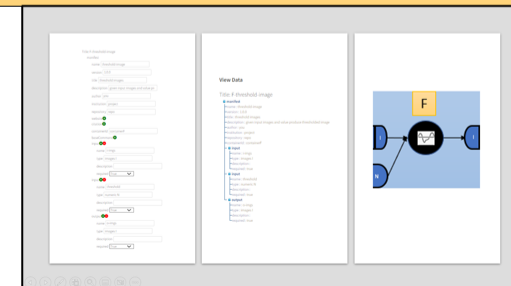
F: Threshold-image

Inputs/ui:

- D: i-imgs: input-images
- E: threshold: threshold-value

Outputs:

- G: o-imgs: output-images





Manifest curation examples with CDCS

Materials Data Curation System Home Data Curation Data Exploration Composer Help

Select Template

Select a template from the following table. Once you make your selection, start a new document or open an existing form or start from an uploaded document automatically load the appropriate form and display it on the next page.

Templates

Template	Actions
 manifest	▶ Select Template
 specification	▶ Select Template

Crop image example curation

Title: D-crop-image

manifest

name

version

title

description

author

institution

repository

website

citation

containerId

baseCommand

input

name

type

description

required

input

name

type

description

required

input

name

type

description

required

output

name

type

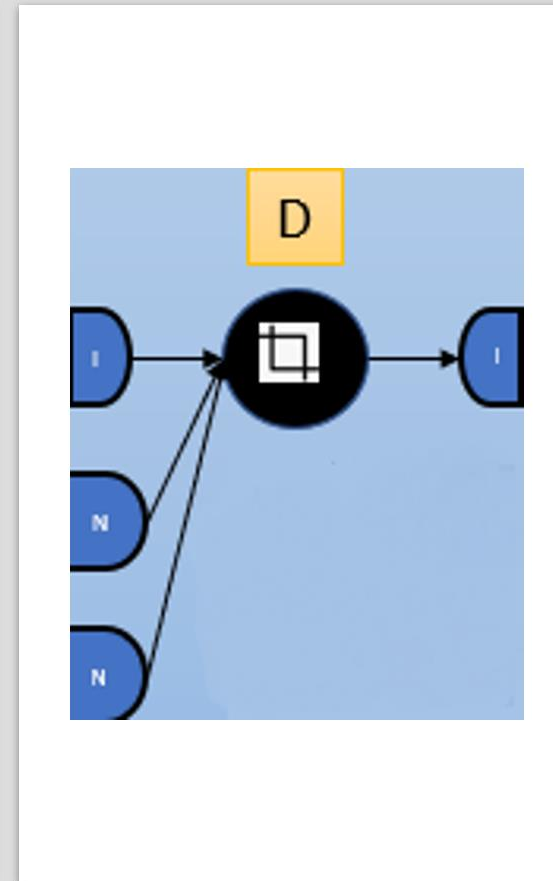
description

required

View Data

Title: D-crop-image

- manifest
 - name: crop-image
 - version: 1.0.0
 - title: crop image
 - description: given inputs, x and y dimensions, crop
 - author: you
 - institution: project
 - repository: repo
 - containerId: containerD
- input
 - name: i-imgs
 - type: images I
 - description: input images
 - required: true
- input
 - name: xDim
 - type: number N
 - description: cropped size x axis
 - required: true
- input
 - name: yDim
 - type: number N
 - description: cropped size y axis
 - required: true
- output
 - name: o-imgs
 - type: images I
 - description: output images
 - required: true



Threshold image example curation

Title: F-threshold-image

manifest

name

version

title

description

author

institution

repository

website

citation

containerId

baseCommand

input

name

type

description

required

input

name

type

description

required

output

name

type

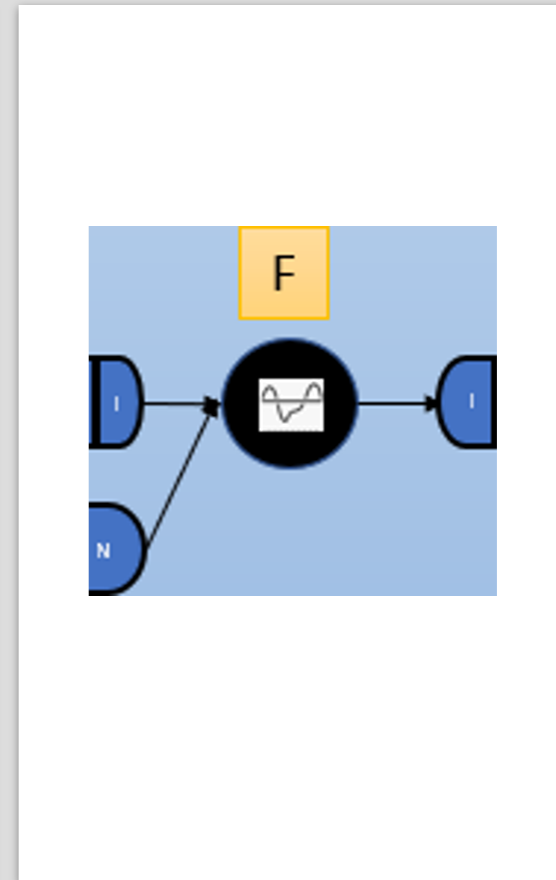
description

required

View Data

Title: F-threshold-image

- manifest
 - name : threshold-image
 - version : 1.0.0
 - title : threshold images
 - description : given input images and value produce thresholded image
 - author : you
 - institution : project
 - repository : repo
 - containerId : containerF
- input
 - name : i-imgs
 - type : images I
 - description :
 - required : true
- input
 - name : threshold
 - type : numeric N
 - description :
 - required : true
- output
 - name : o-imgs
 - type : images I
 - description :
 - required : true



Search component manifests in CDCS

Enter keywords, or leave blank to retrieve all records

Local Results

MDCS

Federated Search

No federated instance is available.

Filter by Template [Select All](#)

manifest

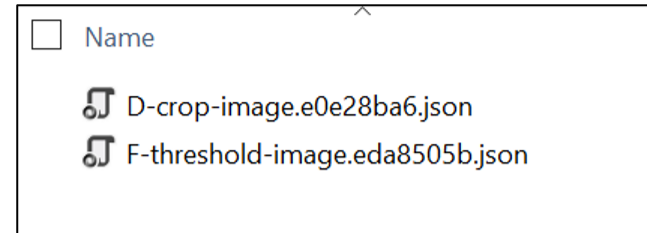
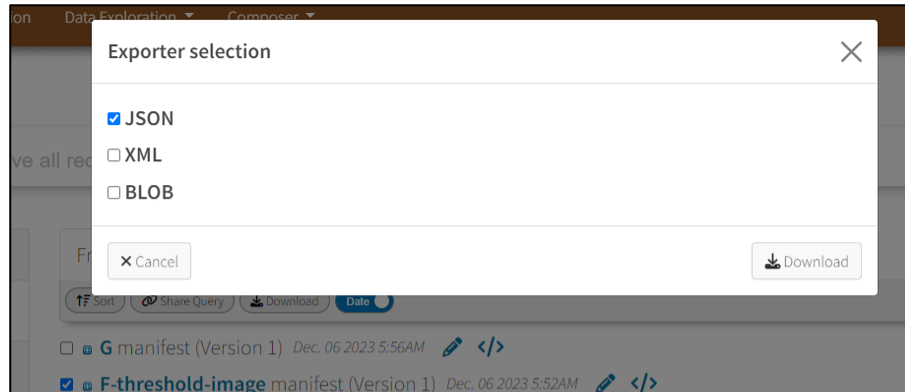
specification

From MDCS **7**

[Sort](#) [Share Query](#) [Download](#) [Date](#)

- G** manifest (Version 1) *Dec. 06 2023 5:56AM* [edit](#) [code](#)
- F-threshold-image** manifest (Version 1) *Dec. 06 2023 5:52AM* [edit](#) [code](#)
- E** manifest (Version 1) *Dec. 06 2023 5:46AM* [edit](#) [code](#)
- D-crop-image** manifest (Version 1) *Dec. 06 2023 5:41AM* [edit](#) [code](#)
- C** manifest (Version 1) *Dec. 06 2023 5:34AM* [edit](#) [code](#)
- B** manifest (Version 1) *Dec. 06 2023 5:31AM* [edit](#) [code](#)
- A** manifest (Version 1) *Dec. 06 2023 5:28AM* [edit](#) [code](#)

Export/translate in other formats (JSON, XML, others)



Access and manipulate via CDCS REST API (use to automate services, tools, UIs...)

```
analyze.ipynb Python 3 (ipykernel) Code
```

Access and manipulate via REST

```
[1]: import cdc
from cdc import CDCS
print('Notebook executed for cdc version', cdc.__version__)
curator = CDCS('http://127.0.0.1/', username='blong', verify=False)
print(curator.cdcversion)

Notebook executed for cdc version 0.2.2
Enter password for blong @ http://127.0.0.1: .....
(3, 6, 0)
```

show our schemas

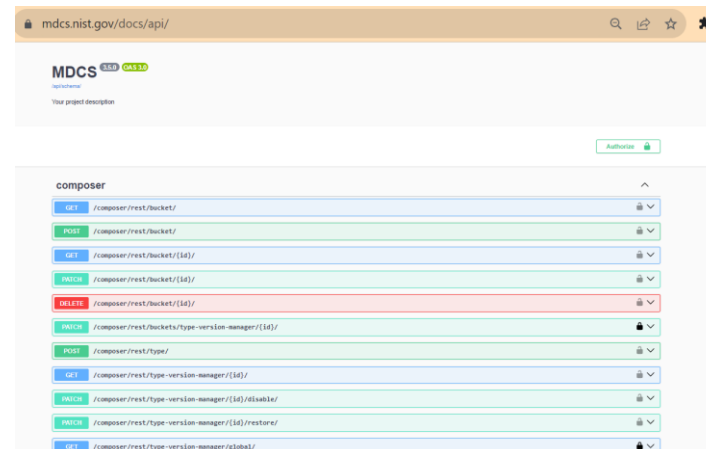
```
[2]: curator.template_titles

[2]: ['manifest', 'specification']
```

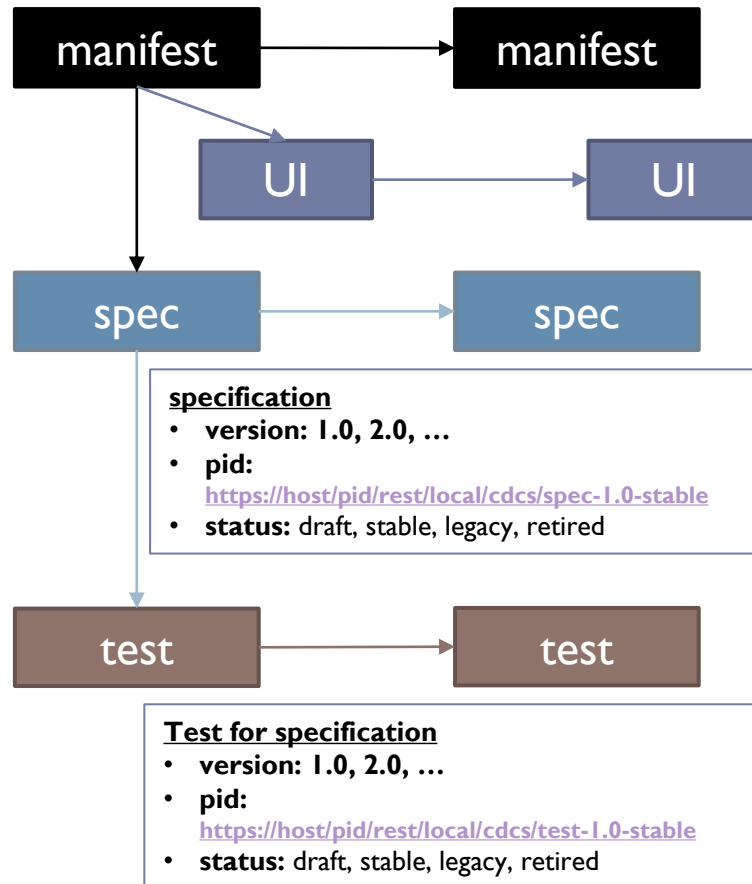
access our manifests via REST

```
[3]: curator.query(template='manifest', keyword='')
```

	id	template	workspace	user_id	title	creation_date	last_modification_date	last_change_date	xml_content	template_title
0	7	1	None	1	G	05:56:33.314000+00:00	2023-12-06 05:56:33.314000+00:00	2023-12-06 05:56:33.314000+00:00	<manifest xmlns:xsi="http://www.w3.org/2001/X...	manifest
1	6	1	None	1	F-threshold-image	05:52:11.022000+00:00	2023-12-06 05:52:11.022000+00:00	2023-12-06 05:52:11.022000+00:00	<manifest xmlns:xsi="http://www.w3.org/2001/X...	manifest
2	5	1	None	1	E	05:46:24.992000+00:00	2023-12-06 05:46:24.992000+00:00	2023-12-06 05:46:24.992000+00:00	<manifest xmlns:xsi="http://www.w3.org/2001/X...	manifest
3	4	1	None	1	D-crop-image	05:41:29.819000+00:00	2023-12-06 05:41:29.819000+00:00	2023-12-06 05:41:29.819000+00:00	<manifest xmlns:xsi="http://www.w3.org/2001/X...	manifest
4	3	1	None	1	C	05:34:11.695000+00:00	2023-12-06 05:34:11.695000+00:00	2023-12-06 05:34:11.695000+00:00	<manifest xmlns:xsi="http://www.w3.org/2001/X...	manifest



Each of these are linked with PIDs



Some useful considerations for specification evolution

- Can uniquely identify and store manifests, UIs, specs, and tests in distributed manner
- Any manifest can support 1+ specs (allow evolutionary compatibility of manifests based on supported specs)
 - Specs can have lifecycle phases: draft, stable, legacy, retired
 - Later specs can be backward compatible with previous ones
 - Components/manifests can claim compliance with 1+ specs users might use
- Every spec has associated tests to verify compliance
 - This enables users verify specification compatibility at a component/manifest level: important for interoperability
- A given manifest can refer to various supporting UIs

Conclusions

▶ **Needs**

- ▶ Interoperable containers/manifests have a number of needs regarding
 - ▶ curation, formatting, manipulation, unique identification, association with UIs, specifications, tests, and needs to be discoverable and accessible through UIs and automated processes

▶ **Challenges**

- ▶ These needs are not typically addressed in a unified way through existing processes, components, or platform infrastructure (such as GitHub/DockerHub) by themselves
- ▶ Without support from platforms/methods like CDCS, there can often be significant varieties/duplication of format, information, and more

▶ **Solutions**

- ▶ CDCS core functionality, including support and applications of PIDs, integrated into lifecycle usage can help facilitate eco-system level reuse, interoperability, automation, and consensus specification coordination/evolution
- ▶ Lifecycle use of FAIR containers can be facilitated by CDCS platform infrastructure (CDCS registries and repositories)
 - ▶ These follow best practices in software and data
- ▶ Example use-cases covered
 - ▶ Curation - definition and evolution of components
 - ▶ Export/translation – JSON, XML, other
 - ▶ Persistent identification
 - ▶ API-based automation and integration with services/UIs (REST)
 - ▶ Reuse and interoperability
 - Of manifests, specifications, tests
 - Within a shared eco-system

CDCS Project Information

Website	https://cdcs.nist.gov
Documentation / Tutorials	https://cdcs.nist.gov/cdcs-documentation/index.html
Software	Repository : https://github.com/usnistgov/mdcs Registry: https://github.com/usnistgov/nmrr Install: https://github.com/usnistgov/cdcs-docker
REST client	pycdcs https://github.com/usnistgov/pycdcs
Contact	Benjamin Long benjamin.long@nist.gov