Big Data Challenges

**VERACITY**  **VARIETY**  **VOLUME**

---

### Abstract

At the Biological and Chemical Oceanography Data Management Office (BCO-DMO) Big Data challenges have been steadily increasing. The sizes of data submissions have grown as instrumentation improves. Complex data types can sometimes be stored across different repositories. This signals a paradigm shift where data and information that is meant to be tightly-coupled and has traditionally been stored under the same roof, is now distributed across repositories and data stores. For domain-specific repositories like BCO-DMO, a new mechanism for assembling data, metadata and supporting documentation is needed.

Traditionally, data repositories have relied on a human’s involvement throughout discovery and access workflows. This human could assess fitness for purpose by reading loosely coupled, unstructured information from web pages and documentation. Distributed storage was something that could be communicated in text that a human could read and understand. However, as machines play larger roles in the process of discovery and access, distributed resources must be described and packaged in ways that fit into machine automated workflows of discovery and access for assessing fitness for purpose by the end-user. Once machines have recommended a data resource as relevant to an investigator’s needs, the data should be easy to integrate into that investigator’s toolkits for analysis and visualization.

BCO-DMO is exploring the idea of data containerization, or packaging data and related information for easier transport, interpretation, and use. Data containerization reduces not only the friction data repositories experience trying to describe complex data resources, but also for end-users trying to access data with their own toolkits. In researching the landscape of data containerization, the Frictionless Data Package (https://frictionlessdata.io/) provides a number of valuable advantages over similar solutions. This presentation will focus on these advantages and how the Frictionless Data Package addresses a number of real-world use cases faced for data discovery, advantages over similar solutions. This presentation will focus on these advantages and how the Frictionless Data Package addresses a number of real-world use cases faced for data discovery, advantages over similar solutions.

---

### What is Data Containerization?

**Problem:** Data can be distributed across multiple locations - databases, files on a server, files on the web, or available from web APIs, etc.

**The Frictionless Data Package is a set of extendible, lightweight formats for packaging data and metadata.**

**datapackage.json** can handle these various locations and sizes?

Q: How do we package data to

BCO-DMO wants:

- Simpler, expedited data ingest for submitters
- Data transformation that captures provenance
- Continuous integration testing of data holdings

---

### Continuous Integration Testing for Data

**Goodtables.io - Continuous data validation as a service.**

https://github.com/frictionlessdata/goodtables.io

Because Data Package Pipelines are declarative, `pipeline-spec.yaml` files are provenance records.

- Pipelines can be re-run to verify that the workflow is reproducible
- Data Packages can be validated
  - Is the datapackage.json well-formed?
  - Does it meet the JSON schema specification?
- Tabular Data Packages have deeper validation
  - Column header checking vs. datapackage.json defined fields
  - Methods for handling missing data values
  - Regex processing of a data cell for conforming to a pattern

---

### What about the work of a data manager?

**Data Package Pipelines - Framework for processing data packages in pipelines of modular components.**

https://github.com/frictionlessdata/datapackage-pipelines

- A pipeline has a list of processing steps, and it generates a single data package as its output.
- A pipeline is defined in a declarative way, not in code, stored in a file named `pipeline-spec.yaml`.
- Data Package Pipelines define some common processors, custom processors can be created.

**Data Managers can extend the datapackage.json to add semantic markup.**

The resulting `datapackage.json` can then be used to populate repository metadata catalog.