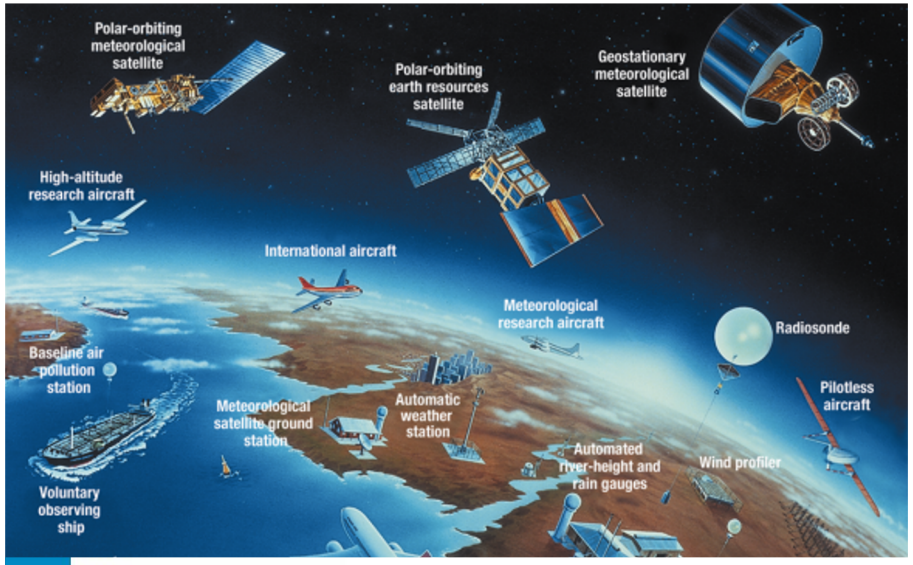
The National Weather Service's Implementation of the OGC Environmental Data Retrieval API

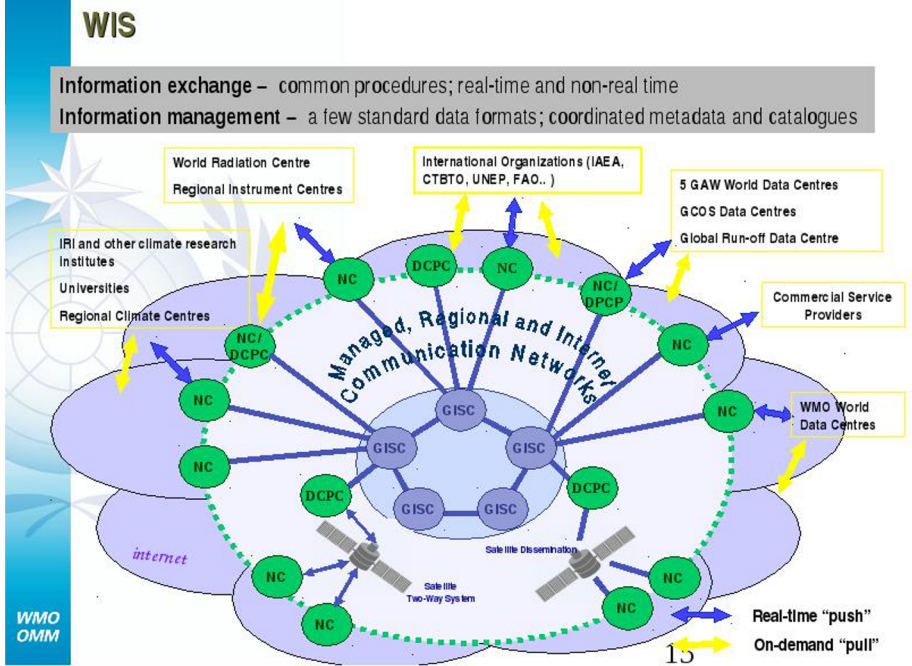
Pangeo Showcase

September 15, 2021

Background regarding OGC EDR-API

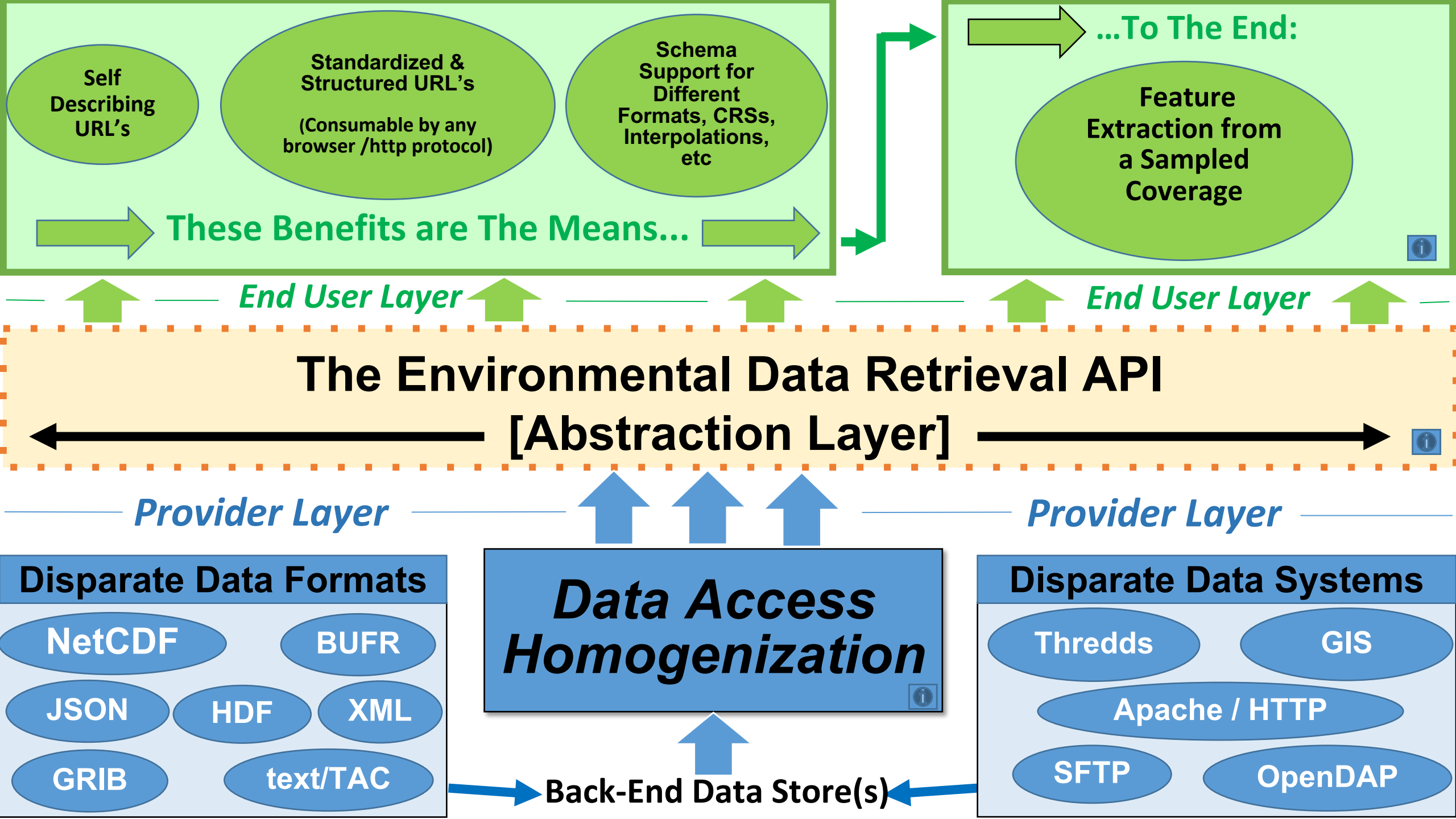
- NWS began in Partnership with UK Met Office
- 2020 OGC sets EDR-API data standards working group and encourages more development - SPRINTs, Working groups (MetOcean DWG/SWG), Github
- Global Met Centers have growing interest and backing including US leads: NASA,USGS, & Unidata
- Help WMO realize goal of data sharing to broader community & WIS 2.0 technical specification





Relevant OGC Links/Resources

- OGC BLOG
  - https://www.ogc.org/blog/3211
- OGC EDR API website
  - https://ogcapi.ogc.org/edr/
- OGC EDR API Standards Working Group
  - https://github.com/opengeospatial/ogcapi-environmental-data-retrieval
- EDR API Deployments
  - https://github.com/opengeospatial/ogcapi-environmental-data-retrieval/blob/master/deployments.md


Self Describing URL's
Standardized & Structured URL's
(Consumable by any browser /http protocol)
Schema Support for Different Formats, CRSs, Interpolations, etc
These Benefits are The Means...
...To The End:
Feature Extraction from a Sampled Coverage
End User Layer
End User Layer
The Environmental Data Retrieval API
[Abstraction Layer]
Provider Layer
Provider Layer
Disparate Data Formats
NetCDF
BUFR
JSON
HDF
XML
GRIB
text/TAC
Data Access Homogenization
Back-End Data Store(s)
Disparate Data Systems
Thredds
GIS
Apache / HTTP
SFTP
OpenDAP

The NWS EDR API under the hood .. A look at the technologies

- Xarray

Xarray uses labels in the form of dimensions, coordinates, and attributes on top of raw numpy like arrays (N-Dimensional Indexed Arrays). These labels allow for the organization of data into “collections” or groups of weather parameters that share dimensionality.

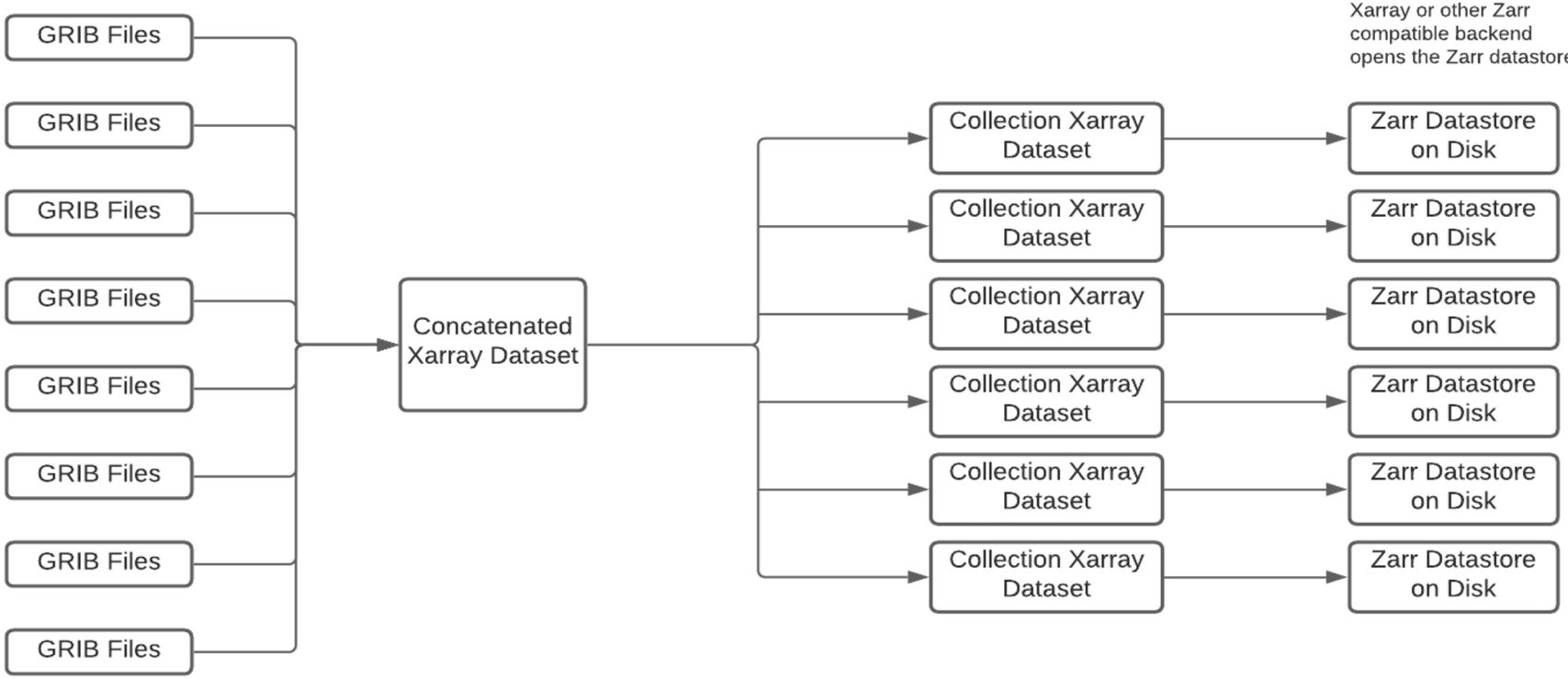
- Dask
  - Dask is a flexible library for parallel computing in Python. We use Dask to chunk the data located in the Xarray object to improve performance. Xarray’s use of Dask also allows for “lazy loading” of data so that only the data needed by the user is loaded into memory and everything else is ignored.

- Zarr
  - Zarr provides an implementation of chunked, compressed, N-Dimensional arrays. Once the Xarray objects are chunked by Dask, they are stored as Zarr objects, from which are reopened by the API when a user makes a query. The EDR-API is efficient with multi dimensional data because the data is already organized as collections of shared dimensionality.

The NWS EDR API under the hood ..
EDR API Homogenization example for GRIB Data

Legacy Data Store
- NOMADS service where model data is disseminated as several GRIB files, each GRIB file associated with a forecast time

Optimized Data Solution
- The data is organized how we want.
- Zarr allows us to store on disk, S3, etc.
- Fast performance when an Application with an Xarray or other Zarr compatible backend opens the Zarr datastore.



Ingestion into Xarray

Reorganization and Binning into Collections of Weather Parameters with Common Dimensions

Re-chunking of Data with Dask and Transfer from Xarray Dataset in Memory to Storage of Zarr on Disk

EDR API specification provides complete versatility in constructing collections

- Some data (i.e. text products) doesn't require a zarr backend
- EDR-API is most efficient when collections are constructed by common dimensionality.
- However, EDR allows collections to be constructed to suit any specific use case or need.

WSUP Usecase (1 collection)

Blendv4.0_conus
Weather Params:
appttemp (t,x,y)
ceil (t,x,y)
cprbfzrain (t,x,y)
cprbrain (t,x,y)
cprbsleet (t,x,y)
cprbsnow (t,x,y)
dewpoint (t,x,y)
freezingspray (t,x,y)
hindex (t,x,y)
maxrh (t,x,y)
maxt (t,x,y)
minrh (t,x,y)
mint (t,x,y)
pop01 (t,x,y)
pop06 (t,x,y)
pop12 (t,x,y)
qpf01 (t,x,y)
qpf06 (t,x,y)

Somewhat unique to WSUP usecase is that t could differ by weather parameter, so technically, these are not weather parameters grouped by common dimension.

Automated GFS Usecase (many collections)

gfs_lat_lon_time_lv_ISBL0_Isobaric
Weather Params:
ABSV_P0_L100_GLL0 (lat,lon,time,lv_isbl)
DZDT_P0_L100_GLL0 (lat,lon,time,lv_isbl)
HGT_P0_L100_GLL0 (lat,lon,time,lv_isbl)
O3MR_P0_L100_GLL0 (lat,lon,time,lv_isbl)
RH_P0_L100_GLL0 (lat,lon,time,lv_isbl)
SPFH_P0_L100_GLL0 (lat,lon,time,lv_isbl)
TMP_P0_L100_GLL0 (lat,lon,time,lv_isbl)

gfs_lat_lon_time_lv_DBLL11_Depth_Below
Weather Params:
SOILL_P0_2L106_GLL0 (lat,lon,time,lv_dbll)
SOILW_P0_2L106_GLL0 (lat,lon,time,lv_dbll)
TSOIL_P0_2L106_GLL0 (lat,lon,time,lv_dbll)

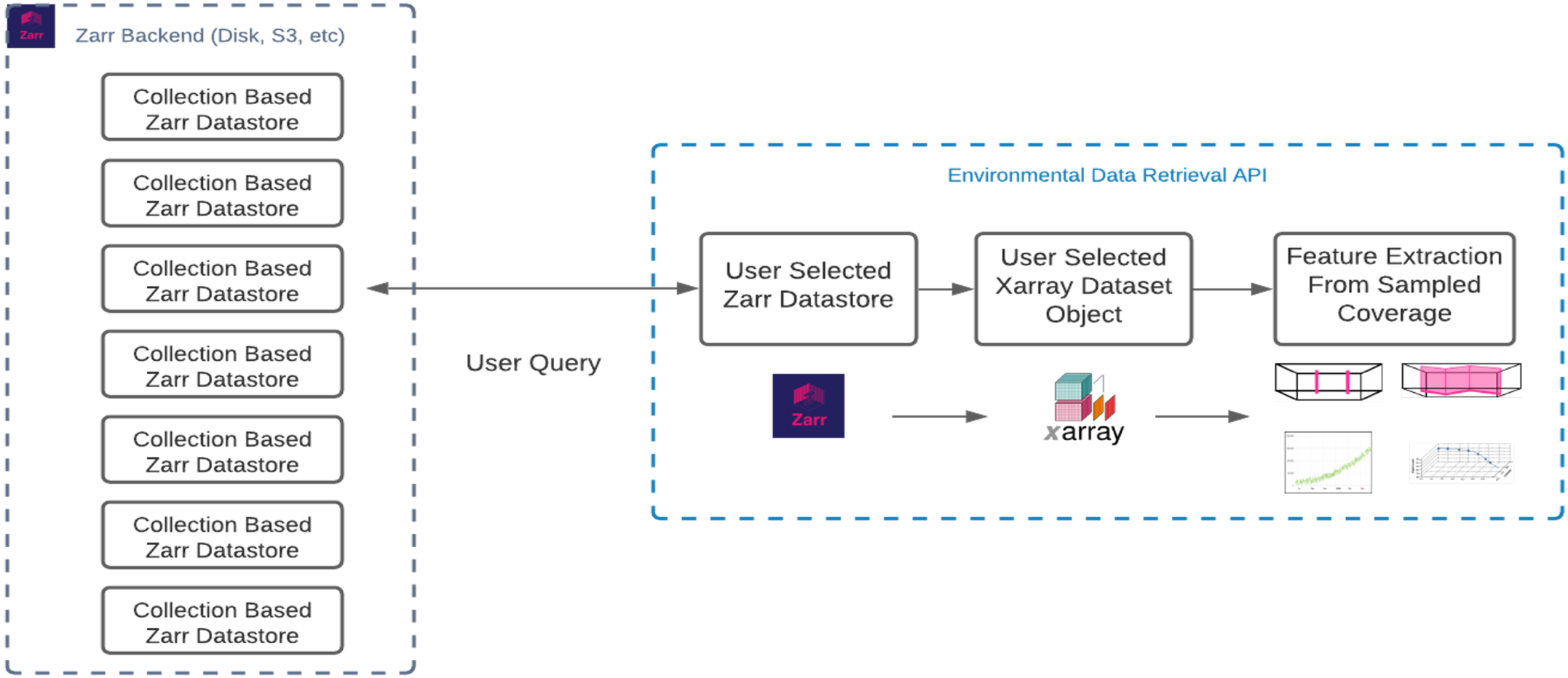
gfs_lat_lon_time_Mean_sea_level_Pa
Weather Params:
MSLET_P0_L101_GLL0 (lat,lon,time)
PRMSL_P0_L101_GLL0 (lat,lon,time)

gfs_lat_lon_time_Entire_Atmosphere
Weather Params:
TCDC_P8_L10_GLL0_avg6h (lat,lon,time)

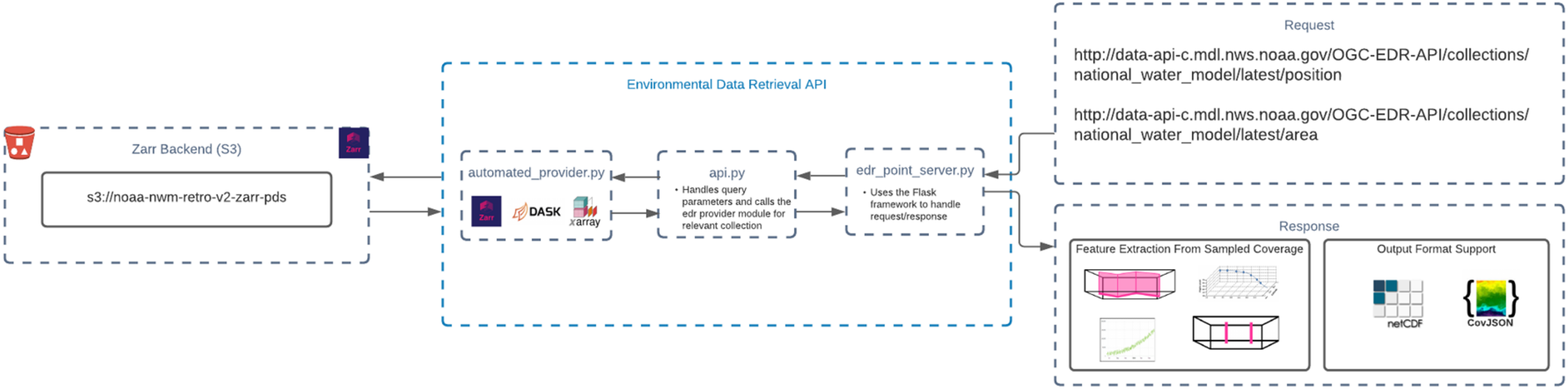
...etc.

Meanwhile, the automated GFS collection approach groups weather parameters into collections that share common dimensionality and level type.

Basic Example of EDR-API interacting with Backend Zarr



National Water Model EDR-API Implementation



Demo

- Open API overview
- Sampling Geometry Type Examples
- Multiple Output Format Examples
- National Water Model Example
- Dask Dashboard during an EDR-API Request to S3

Key Takeaways

- The NWS EDR-API implementation uses similar if not the same technologies that the Pangeo community use.
- This EDR-API implementation is an implementation of an international standard through the OGC.
- For those interested in implementing the EDR-API, the pygeoapi implementation is a great starting point. In addition, the EDR-API Standard Working group will be looking at putting together documentation and a Best Practices document to aid developers.
- We look forward to becoming more engaged and involved with the Pangeo community. It is exciting to see how we can streamline the access to big data to make life easier for our users.