High-Level Design of a Data Carousel for the Basic Fusion Files

May, 2020

Mattias Carrasco-Kind[1]
Gregory Daues[1]
Benjamin Galelwsky [1]
Aleksandar Jelenak[3]
Margaret Johnson[1]
Ryan Kolak[2]
Donald Petravick[1]
John Readey[3]

[1] National Center for Supercomputing Applications, University of Illinois at Urbana-Champaign;
[2]Amazon Web Services
[3]The HDF Group.

Sometimes data is large enough that the resources needed to merely hold the data can severely strain budgets. When resource constraints are severe, and the alternative is not having access to the data at all, an alternative is to 1) use a cheaper storage solution and 2) mitigate any problems that arise from the use of this type of storage. 3) deal with the restrictions that are present in the solution. We present a white paper based on limited prototyping, reflecting our current thinking on the high-level design and operational model using the Data Carousel Access pattern, applied in the context of Amazon Web services, for the 2.4 PB Basic Fusion Dataset.

Terra Basic Fusion Use Case

The use case under consideration is the Basic Fusion files arising from the Terra Access project, primarily consisting of Level 1 data products from observations in the period from 2000 - 2015 from five instruments on the Terra Satellite. The satellite continues to operate. The dataset has the potential to be extended to include observations through 2021. The volume of the fused data set for the period 2000 - 2015 is 2.4PB, implying that a dataset covering the period 2000 - 2022 would require about 3.4 PB of storage in total.

The Terra Basic Fusion data for the period 2000-2015 is packaged into approximately 84,000 HDF5 files of an average size of 32 GB. Each file contains data from a single orbit from the Terra Satellite. Files vary in size because some instruments are *tasked,* i.e. not operated continually. The Terra Satellite is in a Low Earth Orbit. Unlike Geostationary Satellites which are able to monitor a large fraction of the Earth for each observation, Terra is operated in such a way to continually trace out 233 orbital paths every sixteen days, which provides global coverage every two days for the wider swath Terra instruments. Adjacent paths are not imaged in successive orbits.

Terra has made a significant 20-year record of the Earth system. Uses of the data include analysis of the entire globe over the duration of the mission. These analyses need to see the complete record in order to build a condensed derived data set that does not require seeing all the data simultaneously. Other access patterns include access to specific locations over long periods of time (many orbits), data from multiple instruments imaging the same region, and observations of large fractions of the earth over short periods of time.

NASA's initial commercial cloud deployment is on Amazon Web Services. 150 TB of the Basic Fusion files, the Terra Fusion Sampler, are a public data set on AWS (https://registry.opendata.aws/terrafusion/). The full 2.4 PB of Basic Fusion files reside on Amazon Glacier under NASA's Management. However, these files are not known to be available for science use. The barriers to science use are the high cost of hosting the data on S3, and potentially unbounded data access costs for accessing the data for AWS infrequent use or Glacier storage tiers.

Data Access and Amazon Deep Glacier

Characteristics of Amazon Deep Glacier:

- Ingest itself of data is "free"
- Unlike Amazon S3, there is a per byte access charge. However, data in Deep Glacier can be accessed approximately 91 times each year before the total charges equal S3 charges.
- The latency of access can be hours, and perhaps days
- The storage of data is the lowest cost available on AWS.
- Like S3, data is stored in objects.
- 24 hours of access to data is provided without additional charges with semantics like conventional S3.
- Deep Glacier is engineered to support 40 accesses/day/PB.
- The maximum size of an Object is 5 TB.
- Objects cannot be partially restored.

Using only the technical characteristics of AWS, we obtain the following: 40 requests per PB-day implies 96 requests/day. 2.4 PB can be forced into ~480 5TB objects. A year's storage of the data is less than $29,000. The 2.4 PB data volume permits 96 requests/day. A tour through all the data can be completed in five days. One tour through the data costs ~$13000. The retrieval fee provides for each restored object to be usable for 24 hours. Maintaining availability after that time incurs S3 charges.

Characteristic	**Value**
Restore requests per day	96
Minimum number of objects	480 5 TB objects
Minimum days for a tour	5
Movement costs for a full tour	$13000
Duration of useful access to data once staged	24 hours

The limit on the number of restores requires that the 84000 of approximately 32-GB objects from the existing repository be re-packaged into ~4TB objects needed for the data service. Options range from simple approaches akin to the concatenation of the existing HDF5 files to repacking the data into a Cloud Optimized Geotiff (COG) format.

Native HDF access of S3 objects is best accomplished using HDF Group's Highly Scalable Data Server (HSDS). This product was developed in coordination with NASA and is a server that can run inside AWS and offers a Python interface that matches the popular h5py library. HDF5 files are indexed once at ingest time. The index is used to inform range queries on the restored S3 objects.

For the Basic Fusion Product, the proposed approach is to re-pack the existing files into a ZIP file of the requisite ~4TB size. ZIP uncompressed format (ZIP_STORED),would be used as the data are already compressed within the HDF files themselves. Each original format HDF5 file can then be accessed by a range request on the ZIP file. These ranges can be integrated into the HSDS server. Users can run an HSDS server, and not incur lengthy overheads of re-building server meta-data from scratch.

- The HSDS server provides optimized data movement directly into scientists' programs providing efficient access to just the data they need from the ~4TB object.
- The composite HDF5 files, and sidecar HSDS metadata files, while large, are usable in a stand-alone environment, should they be exported.
- The approach requires minimal changes to existing code.

- Application code wanting to access the files directly can either access the whole zip file or extract individual files of interest.

The primary design considerations for packing Basic Fusion files into Zip files, are those that might minimize staging for carousel cycles which do not require the whole data set. Packing options include spatial (packaging by orbital path) or temporal. Another type of option is simply to choose the largest number of zip files consistent with the designated maximum time for a tour (in our case two weeks -- 1400 files) to maximize the chance that that a zip file would not be needed..

The Data Carousel Access Design Pattern

Repackaging the data for science use in Amazon Deep Glacier would serve very occasional use by a small group of interested scientists, who access the data infrequently. The scientists would likely have to coordinate informally to:

- Not exceed the 40 requests/PB-day limit,
- Devise some triggering mechanism to run code once the data are staged.
- Avoid redundant data stages.

Clearly the scientists need to embrace a “measure twice, cut once” philosophy due to the latencies involved, and the costs of any computation on such large datasets.

A brief survey of scientists at Illinois indicates that, given the constraint, Two weeks is an acceptable cadence to provide Scientific access to file on Amazon Glacier, though further optimizations of the basic carousel scheduling would allow for quickly accessing files for users who request access to a relatively small amount of data. Two weeks allows for slightly smaller objects with an average size of 3.7 TB, with two such objects holding all the data for one complete coverage of the earth. This gives a technology-limited cadence of under 7 days and easily allows for a two-week cadence should the dataset be completed with data through 2021.

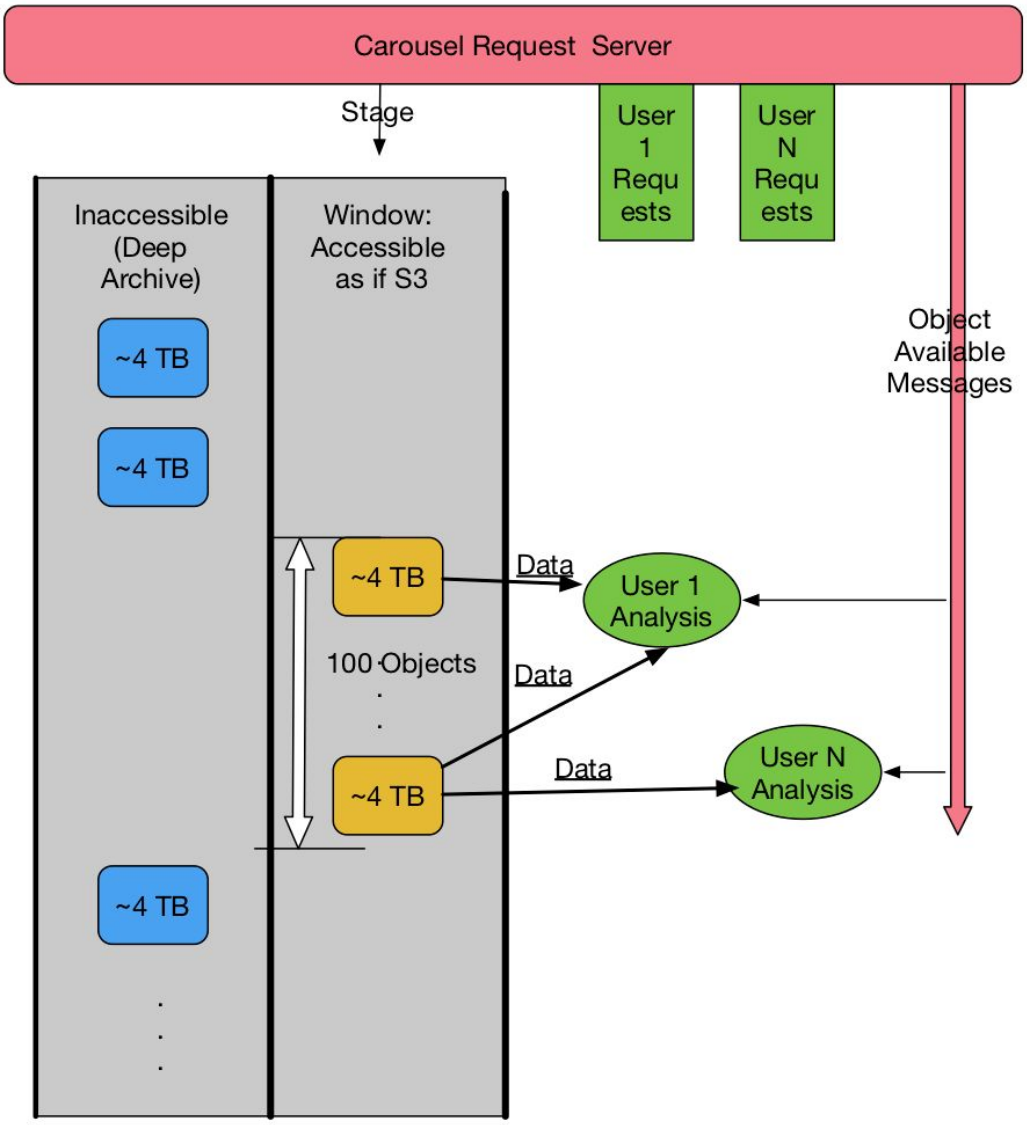
A data carousel is a ***data access design pattern*** that queues up requests for data, stages requested data, and triggers running users’ codes when the data becomes available. This access pattern can address larger and diverse user communities which need to rely on a software system to coordinate their access, rather than informal communications.

The characteristics of a data carousel are:

- Users independently declare data of interest to the Carousel.
- The carousel plans a sequence of stages to place data into an accessible window.
- As the ~4TB objects are staged, each user is notified that the data is now accessible and would be available for 24 hours. This conserves and caps the staging and data movement costs.
- All requested data is eventually presented in the window.

- Tours through the data occur on a cadence. This document uses a two-week window for illustrative purposes. Therefore,
    - The maximum latency is 2 weeks for any particular object to appear in the window
    - The costs of data movement to the carousel operator are absolutely capped.

Figure: Conceptual Operation of a Data Carousel.



Initially, user1 through user N have declared data needed for their analysis to the Carousel Request Server. To help plan an analysis run, the users have

- Data generally available about orbits of the Terra Satellite available within the greater NASA ecosystem.
- The Meta-data collected to support the HSDS server, which would reside in distinct buckets inside AWS.
- The packing meta-data, indicating which files are in which Zip Archives.

Then, using a carousel project supplied software stack, users start a serverless listener, which consumes (practically) no resources. The Carousel identifies the subset of all objects residing in Deep Glacier that are needed by any of the users. Each day, the carousel server makes ~100 requests to AWS Deep Glacier to make the data accessible for a day. As objects become available, the Carousel Server broadcasts availability to the serverless listeners associated with each user. The listener acquires compute resources and runs software that accesses the data. Each user has 24 hours to complete their analysis of the data in that object.

The carousel project supplies a software and data stack to users.

- Software to listen for availability messages,
- Software to run an HSDS server
- HSDS meta-data for each object
- A template framework for running user software.

Users are free to exploit the template or to choose a variety of AWS solutions to run their codes.

In this model, the carousel operator bears the costs of staging the data. Each User bears the cost of running their own software stacks as well as the costs of saving and managing their outputs.

AWS Costs

The table below shows approximate annual operations costs based on AWS list prices for US-Virginia, as of May 13, 2020. AWS costs are dominated by storage costs. **Operating costs are maximum costs, assuming all data are accessed every-other-week. It is likely that initial costs will be lower, as it will take time for the analysis community to adapt to exploit the mechanism. An initial AWS budget of $80,000 seems more than adequate.** The carousel framework can use serverless computing, at little costs. Analysis costs, including HDF5 server costs would be borne by the end-user.

Storage Method	Volume (PB)	Storage $/GB/Month	Annual Storage Cost	#full accesses / year	Retrieval ($/GB)	Max Annual Retrieval cost	**Max Total Annual Cost**
S3	2.4	0.021	$604,800	Unlimited	0	$0	**$604,800**
S3 infrequent access	2.4	0.0125	$360,000	26	0.01	$624,000	**$984,000**

S3 One Zone Infrequent Access	2.4	0.01	$288,000	26	0.01	$624,000	**$912,000**
Glacier	2.4	0.004	$115,200	26	0.0025	$156,000	**$271,200**
Deep Glacier	2.4	0.00099	$28,512	26	0.0025	$156,000	**$184,512**