Beacon data lakes for federated, high-performance access to marine data in the Blue-Cloud2026 ecosystem

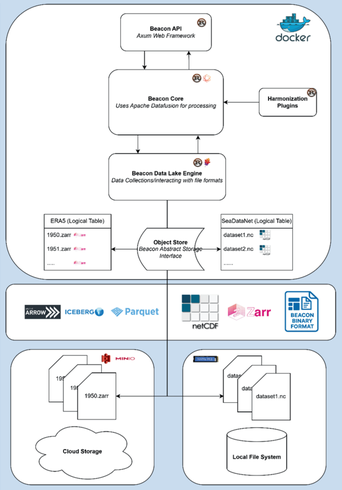
Sharing is encouraged

MARIS

BEACON

A software system for harmonising vast amounts of data in an instant

Efficient, parallel access to multi-dimensional data, with lossless compression



As part of several major European research initiatives, Blue-Cloud2026, EOSC-FUTURE, and ENVRI-Hub NEXT, MARIS has developed a powerful software system called Beacon. Designed to operate within the European Open Science Cloud (EOSC) ecosystem, Beacon enables fast and dynamic access to large-scale, heterogeneous environmental and oceanographic datasets.

At its core, Beacon uses a unique indexing system that allows it to dynamically extract subsets of data on-the-fly based on user queries. This means that users can request a specific subset from millions of files containing different types and structures of observational data, such as gridded datasets, timeseries, or cruise observations and Beacon will return a single, harmonised output file containing just the requested information.

Because the data sources used by Beacon (such as Euro-Argo, SeaDataNet, ERA5, and World Ocean Database) often consist of millions of individual files, two significant challenges arise:

1. Ensuring fast query performance
2. Managing extremely large storage requirements

To address these, MARIS introduced Beacon ATLAS, a purpose-built, high-performance binary format that significantly improves data storage and retrieval. ATLAS was designed with the goal of:

- Reducing the physical storage footprint of large datasets
- Maximizing read and query performance
- Preserving full data fidelity, ensuring that all original information is retained

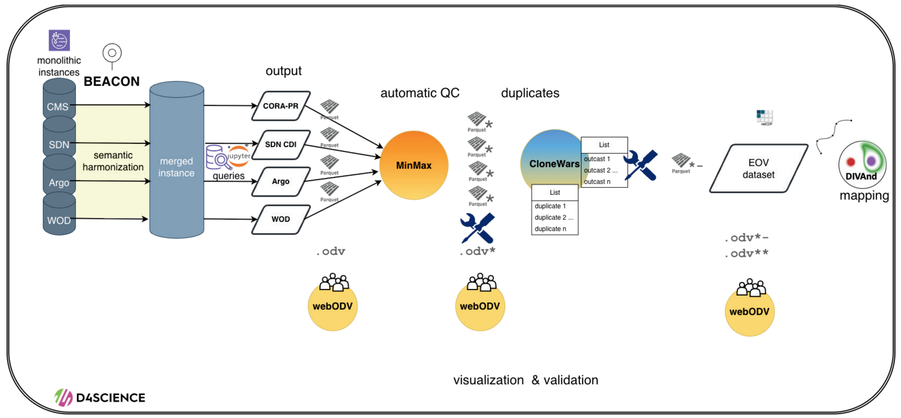
ATLAS achieves these goals through several key innovations:

- It stores multi-dimensional data as Apache Arrow arrays, which allows for zero-deserialization access. In practice, this means that the data can be read directly from disk as if it were in computer memory, eliminating the usual performance hit from format translation.
- It is non-blocking, meaning multiple processor cores can access and read from the file simultaneously. This allows for true parallel access to millions of datasets and enables data transfer speeds of multiple gigabytes per second, often shifting the performance bottleneck from software to hardware.
- It uses adaptive, block-level compression, meaning that each dataset is analyzed and compressed using the most efficient method for that specific data. This improves both storage efficiency and decompression speed, in some cases reaching speeds close to a machine's memory bandwidth.
- Despite being highly optimized, ATLAS maintains lossless integrity: all original data values and structures are retained. An ATLAS file created from a NetCDF file, for example, can be used to fully reconstruct the original NetCDF.
- ATLAS uses a pruning block based indexing system that allows you to push filters to the scan level and skip datasets that don't contain any data relevant to your filters.

This new format is benchmarked and compared with traditional data formats like NetCDF, CSV, and ASCII in terms of performance and storage efficiency, and consistently shows substantial advantages similar to parquet (Analysis Ready Cloud Optimized - ARCO). In January 2025, the Beacon software system was officially released as open-source software (version 1.0.0), making it available to the wider scientific and data management community. This allows any organization to deploy their own Beacon instance, improve access to their data, and reduce storage requirements, without compromising on data quality or completeness.

eosc | Blue-Cloud2026

A federated European FAIR and Open Research Ecosystem for oceans, seas, coastal and inland waters



Eutrophication Workbench Virtual Lab Workflow

https://blue-cloud.d4science.org/group/eutrophication-workbench/eutrophication-workbench

Monitoring usage in VRE

Under Blue-Cloud active data nodes

BDI data collection	Beacon instance	WorkBench
CORA Profile data, retrieved from Copernicus Marine Service: INSITU_GLO_PHY_TS_DISCRETE_MY_013_001	beacon-cora-pr.maris.nl beacon-cora-pr.d4science.org	WB1
CORA Timeseries data, retrieved from Copernicus Marine Service: INSITU_GLO_PHY_TS_DISCRETE_MY_013_001	beacon-cora-ts.maris.nl beacon-cora-ts.d4science.org	WB1
Euro-Argo data, retrieved from S3 bucket	beacon-argo.maris.nl beacon-argo.d4science.org	WB1
SeaDataNet data collection, synced and retrieved via the existing CDI system	beacon-cdi.maris.nl beacon-seadatanet.d4science.org	WB1
World Ocean Database (WOD) actual depths, retrieved from ncei.noaa.gov	beacon-wod.maris.nl beacon-wod.d4science.org	WB1 + WB2
EMODnet Chemistry North East Atlantic Collection (subset of the complete collection) retrieved from EMODnet Chemistry WebODV (Eutrophication_Atlantic_profiles_2023_unrestricted.odv)	beacon-emodnet-chem.maris.nl beacon-emodnet-chemistry.d4science.org	WB2
BGC data, retrieved from Copernicus Marine Service: INSITU_GLO_BGC_DISCRETE_MY_013_046	beacon-cmems.maris.nl beacon-bgc.d4science.org	WB2

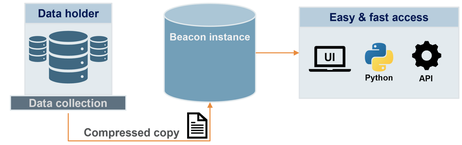
Common Metadata and Data Profile for merged Beacon instances, including target vocabularies

Key metadata	Description		
source BDI			
unique identifier in the source BDI	SeaDataNet/EMODnet Chemistry Local CDI+EDMO	CMEMS CORA/EuroArgo DC_REFERENCE+id	WOD Accession number/ database ID
access date to source BDI	YYYY-MM-DD		
platform	L06 (types), B76 (models) and C17 (instances)		
instrument	L05 (types) and L22 (models)		
variable	P01		
variable standard name	P07 (if required) - could be derived from P01 or I-ADOPT mappings		
units	P06		
quality flag	L27 for identification of source flag scheme, and L20 (SeaDataNet) and/or L34 (IODE) for harmonisation		
time	YYYY-MM-DD THH:MM:SSZ in 24 hours mode and UTC		
geographic position	reference coordinate system WGS84		
depth			
organisation (originator, holding centre)	EDMO code		
project	EDMERP code		
cruise	CSR		
date update			

Overview of deployed merged Beacon instances

Beacon instance, merging BDI data collections	Merged Beacon instance	WorkBench
• CORA Profile data, retrieved from Copernicus Marine Service: INSITU_GLO_PHY_TS_DISCRETE_MY_013_001 • CORA Timeseries, retrieved from Copernicus Marine Service: INSITU_GLO_PHY_TS_DISCRETE_MY_013_001 • Euro-Argo data, retrieved from S3 bucket • World Ocean Database (WOD) actual depths, retrieved from ncei.noaa.gov • SeaDataNet data collection, retrieved from SeaDataNet CDI service	beacon-wb1-ts.maris.nl beacon-wb1-ts.d4science.org	WorkBench 1
• EMODnet Chemistry North East Atlantic Collection (subset of the complete collection) retrieved from EMODnet Chemistry WebODV (Eutrophication_Atlantic_profiles_2023_unrestricted.odv) • World Ocean Database (WOD) actual depths, retrieved from ncei.noaa.gov • BGC data, retrieved from Copernicus Marine Service: INSITU_GLO_BGC_DISCRETE_MY_013_046	beacon-wb2-eutrophication.maris.nl beacon-wb2-eutrophication.d4science.org	WorkBench 2

Contact MARIS
Download now!
Apache V2 license
beacon.maris.nl
github.com/maris-development/beacon



Beacon nodes are easily accessible via a Jupyter Notebook

In order to request data from the WOD Beacon endpoint, the query body needs to be constructed. In the image below you can find an example of how such a query can look. In this case we are querying temperature (parameter), its units, time, depth, longitude and latitude, while filtering on time and depth.

```
query_builder = tables['default'].query()

query_builder.add_select_column("time", alias="TIME")
query_builder.add_select_column("lon", alias="LONGITUDE")
query_builder.add_select_column("lat", alias="LATITUDE")
query_builder.add_select_column("Temperature", alias="TEMPERATURE")
query_builder.add_select_column("Temperature_WODflag", alias="TEMPERATURE_QC")
query_builder.add_select_column("Temperature.units", alias="TEMPERATURE_UNIT")
query_builder.add_select_column("z", alias="DEPTH")
query_builder.add_select_column("z.units", alias="DEPTH_UNIT")
query_builder.add_select_column(".featuretype", alias="FEATURE_TYPE")
query_builder.add_range_filter("TIME", "2022-01-01T00:00:00", "2023-01-01T00:00:00")
query_builder.add_is_not_null_filter("TEMPERATURE")
query_builder.add_equals_filter("TEMPERATURE_QC", 0.0)
query_builder.add_range_filter("DEPTH", 0, 10)

df = query_builder.to_pandas_dataframe()
df
```

8.4s

A user can then send this query body to the Beacon endpoint, as seen in the image below. This provides us with the requested data in a dataframe containing all the data fitting the provided filters above in only 8 seconds! Extra metadata columns can be added to the dataframe by extending the query body above with extra elements.

```
Creating JSONQuery with from: FromTable(table='default')
Running query: {"output": {"format": "parquet"}, "select": [{"column": "time", "alias": "TIME"}, {"column": "lon", "alias": "LONGITUDE"}, {"colum
```

	TIME	LONGITUDE	LATITUDE	TEMPERATURE	TEMPERATURE_QC	TEMPERATURE_UNIT	DEPTH	DEPTH_UNIT	FEATURE_TYPE
0	2022-01-01 00:02:01.287	-145.503998	80.081703	-1.5947	0	degree_C	3.957628	m	Profile
1	2022-01-01 00:02:01.287	-145.503998	80.081703	-1.5948	0	degree_C	5.936413	m	Profile
2	2022-01-01 00:02:01.287	-145.503998	80.081703	-1.5947	0	degree_C	7.915180	m	Profile
3	2022-01-01 00:02:01.287	-145.503998	80.081703	-1.5947	0	degree_C	9.893927	m	Profile
4	2022-01-04 00:02:01.824	-156.945602	73.264198	-1.5380	0	degree_C	3.958745	m	Profile
...	...	...	...	...	...	...	...	...	...
3540982	2022-12-30 15:45:00.000	-52.298000	-47.342499	14.6800	0	degree_C	4.958282	m	Profile
3540983	2022-12-30 15:45:00.000	-52.298000	-47.342499	14.6800	0	degree_C	5.850760	m	Profile
3540984	2022-12-30 15:45:00.000	-52.298000	-47.342499	14.6790	0	degree_C	6.842398	m	Profile
3540985	2022-12-30 15:45:00.000	-52.298000	-47.342499	14.6780	0	degree_C	7.834031	m	Profile
3540986	2022-12-30 15:45:00.000	-52.298000	-47.342499	14.6770	0	degree_C	8.924802	m	Profile

3540987 rows × 9 columns

In this example we can then very quickly plot this subset of the temperature data between 0-10 meters depth from the WOD collection of the year 2022.