Workshop on Visualisation in Environmental Sciences (EnvirVis) (2021)
S. Dutta and K. Feige and K. Rink and D. Zeckzer (Editors)

# Digital Earth Viewer: a 4D visualisation platform for geoscience datasets

V. Buck †1, F. Stäbler†1, E. González1 and J. Greinert2

1Digital Earth, GEOMAR Helmholtz Centre for Ocean Research Kiel, Kiel, Germany
2DeepSea Monitoring / Marine Geosystems, GEOMAR Helmholtz Centre for Ocean Research Kiel, Kiel, Germany

**Abstract**

*A comprehensive study of the Earth System and its different environments requires understanding of multi-dimensional data acquired with a multitude of different sensors or produced by various models. Here we present a component-wise scalable web-based framework for simultaneous visualisation of multiple data sources. It helps contextualise mixed observation and simulation data in time and space.*

## 1. Introduction

The Earth functions as a system where diverse phenomena of physical, chemical, and biological nature are interlinked on very different spatial and temporal scales. Environmental phenomena can cover entire oceans and span millions of years, or can take place on a meter scale within seconds [RUB18]. Geoscience studies rely on the analysis of such heterogeneous datasets. This is arguably the most diverse kind of data encountered in any scientific discipline.
Scientists studying the Earth System are confronted with the exploration of these data sources in their search for knowledge. The technological advances of the last decades have led to an explosion in data collection and generation [FD06]. If the mining of such data, the understanding of relationships, and the discovery of models that lead to knowledge should all grow at a comparable speed, geoscientists too must profit from the same advances in technology. One way in which digital technologies can support the cognitive tasks is through the creation of visualisation tools [CRA00]. However, the development of these tools is only justified if they will provide a cost vs. revenue ratio better than that of existing solutions [WIJ05].
The Digital Earth Viewer is a web-based visualisation platform capable of ingesting data from heterogeneous sources and performing spatial and temporal contextualisation upon them. It allows the parallel navigation of large datasets in a virtual 4D environment. The examination of further aspects of this data with the use of classic 2D graph overlays and user defined operations to combine single datasets into new data products are also implemented. In particular, we argue that these capabilities make the Digital Earth Viewer perform the tasks of map-like visualisation (identify location, retrieve values, assess distances, and trace paths) [HSS20] with higher efficiency than other existing workflows.

## 2. Related Work

Most commercial sensors provide software tools for data access and visualisation. These tend to be proprietary and single-use solutions. Using this kind of applications makes the contextualisation of different data sources a cumbersome task that requires the synchronisation of multiple unrelated visualisation tools.
Ocean Data View (ODV) is a software package for the analysis and visualisation of environmental datasets, with a focus on oceanographic data. Its ability to deal with multiple environmental variables, its compatibility with many common marine data formats, and its ability to create visual transformations in the form of commonly used 2D graphs and charts, have led to a wide adoption by the scientific community [SCH02].
Many geographic information system (GIS) software packages (e.g. ArcGIS, QGIS, Saga) are used to analyse and visualise geographic data. They provide strong analysis capabilities and can access most of the data types commonly used in the geosciences. However their 3D visualisation capabilities are often limited and, to the best of our knowledge, none of them treats time as a true dimension. WebGIS applications like the maps@awi[†] interface exemplify the benefits of uncomplicated, cross-platform organisation and display of geographical data. They too are limited to a two-dimensional data representation.

[†] equal contribution

[†] https://maps.awi.de/awimaps/catalog/

DOI: 10.2312/envirvis.20211081

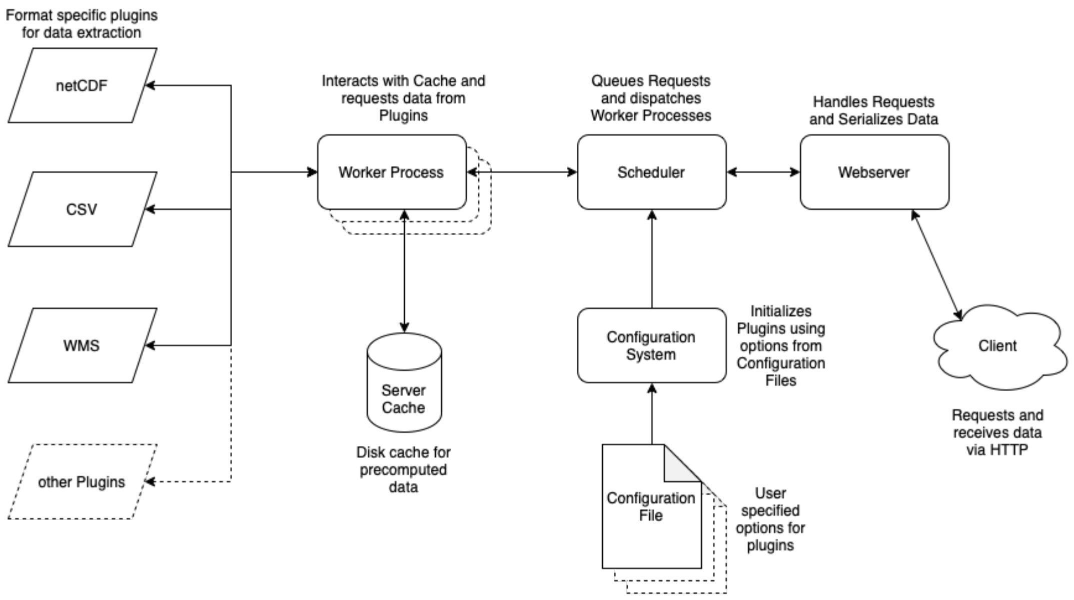

Format specific plugins for data extraction
netCDF
CSV
WMS
other Plugins
Interacts with Cache and requests data from Plugins
Worker Process
Server Cache
Disk cache for precomputed data
Queues Requests and dispatches Worker Processes
Scheduler
Handles Requests and Serializes Data
Webserver
Configuration System
Initializes Plugins using options from Configuration Files
Configuration File
User specified options for plugins
Client
Requests and receives data via HTTP

**Figure 1:** *Digital Earth Viewer server component architecture*

The Cesium engine[‡] is a commercial product based on open-source software components which can be used to visualise pre-processed geospatial data through a web browser. The visualisation uses existing data processing pipelines and leans on multiple online services, such as WMS tile servers like Microsoft's VirtualEarth or ArcGIS Online. Several applications for scientific visualisation using CesiumJS are currently under development. A prominent example is the NASAs Global Precipitation Measurement Program[§]. Due to the numerous requirements of the data pipeline, CesiumJS applications usually need to be supported by powerful back-end servers. Furthermore, visualisations in CesiumJS are based on so-called "3D-tiles", which can only be scaled uniformly. This helps with performance in regional visualisations, but limits the applicability of effects such as the exaggeration of terrain at global scales.
Another product which has seen widespread use for geospatial visualisation is the NASA WorldWind engine[¶]. It was originally developed for the use in the NASA Earth Observing System Data and Information System Project and allows for 2.5D visualisation of WMS-delivered maps. It offers a basic map navigation functionality and serves as a way to present preprocessed maps to an audience in an appealing way.
Summarising the main shortcomings of the visualisation tools mentioned above: many of these tools fail to capture the four-dimensional character of geoscience datasets. The 3D and 4D tools that fulfil this expectation, are often geared towards an online deployment, which is not always attainable (for example during an expedition). Furthermore, most solutions focus on presenting preprocessed data, which is usually the result of a scientific process and might only be of limited use during research.

## 3. Materials and Methods

The Digital-Earth-Viewer is a hybrid application, which is split into two parts: 1) a server backend which handles data extraction from different file formats as well as re-gridding into viewable areas and caching, and 2) a web frontend, which is responsible for visualising the data. The client controls the backend via a stateless HTTP interface [IET14].
The server backend is built as a plugin-oriented architecture around a central dispatcher and written in the rust programming language. Requests for data pages are handled by an asynchronous HTTP server, checked for consistency, and distributed to a set of worker threads as per the proactor pattern. File-format specific plugins extract the requested data and reformat it to be suitable for the client's requirements. Since some data sources require costly computations to extract pages and/or load information from network resources, the formatted data pages are stored in a database. Before dispatching a request to a plugin, the database is queried for the requested page and if available, the precomputed data is returned instead of recalculating it from the source (fig. 1).
The client is built as a dynamic MVVM [BAA*11] application written in TypeScript. It is built around the VueJS framework, and implements a custom WebGL rendering engine. Since the model needs to handle potentially long-running requests to the server, it is constructed as a Service Oriented Device Architecture [MLK10] with a set of services which are instantiated by a central service broker. These services handle different functions, such as requesting and caching the data that is to be visualised, holding state for the WebGL rendering cycle or storing application and visualisation configurations.

‡ https://cesium.com/platform/cesiumjs/
§ https://gpm.nasa.gov/global-viewer/
¶ https://worldwind.earth/

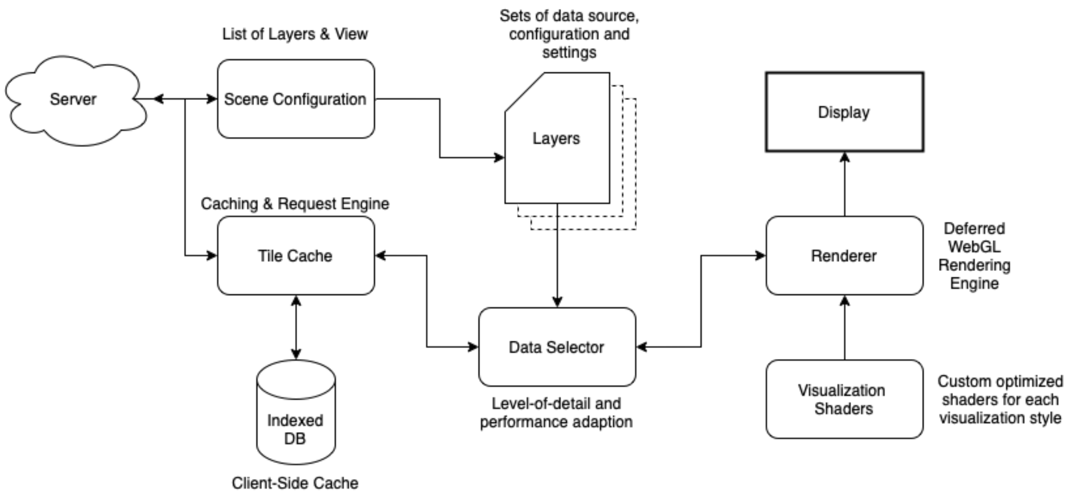

List of Layers & View
Sets of data source, configuration and settings
Server
Scene Configuration
Layers
Display
Caching & Request Engine
Tile Cache
Renderer
Deferred WebGL Rendering Engine
Data Selector
Indexed DB
Level-of-detail and performance adaption
Visualization Shaders
Custom optimized shaders for each visualization style
Client-Side Cache

**Figure 2:** *Digital Earth Viewer client component architecture*

A stored scene setup is retrieved from the server and converted into a set of presentation layers which reference a data source as well as display settings. A data selection algorithm then decides which sections need to be requested from the server. These requests are handled by the tile cache component: it queries a browser-built-in cache as well as the server component to retrieve the desired parts. The requested data is used by a renderer, which gets periodically called by the browser. It uses shaders specific to the type of visualisation requested (fig 2).
Rendering through the WebGL-API allows for dynamic visualisations, such as changing color mappings or terrain exaggeration without time-intensive recomputation; it also displays vector fields as animated tracers.
A further capability implemented in the client is the functionality to interactively interrogate the data. This can be done by selecting regions on the 4D map surface which then call out the data value at this specific point or area. The data selection can be forwarded to advanced graphing components built using d3.js.
The client requests data by letting the rendering engine decide "offline" what data will likely be visible. As the client cannot instantly get all the required data from the server, low resolution data is used as a stand-in for the requested data until it becomes available.
To efficiently address spatiotemporal regions in caches, the 4-dimensional datasets are split into temporal pages and spatial tiles. The spatial tiling system follows a recursive scheme that allows any region of the earth, including the poles, to be covered by a finite number of tiles for any given resolution (fig.3). Temporal quantisation of a contiguous dataset into what we call "time-pages" splits data to additionally reduce transfer sizes on the client-server interface. While spatially dense data is transferred as individual time steps, spatially sparse data is grouped in pages of a defined transfer size. These caching schemes also help limit the amount of data which needs to be processed by the clients GPU, since the different level of tiles implement a natural level-of-detail mechanism.

High-performance display of quickly configurable 3D graphics is enabled by offloading a large amount of the client side computation to the GPU through the WebGL API. This includes wrapping the geographic coordinates of the displayed entities onto the globe that can be seen at large view scales. As the accuracy of this operation is limited by single precision floating point computations, it becomes imprecise when viewing objects at scales of a few meters. At those scales, instead a pseudo-equirectangular projection is used that corrects for the length of the parallels at the viewed latitude. The transition between the two display styles is seamless.
Currently, general-purpose-sources are implemented for the netCDF, ESRI ASCII Grid, and CSV file formats, as well as fa-

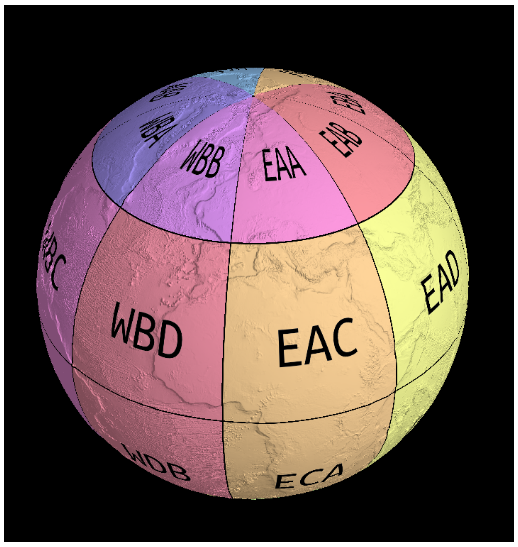
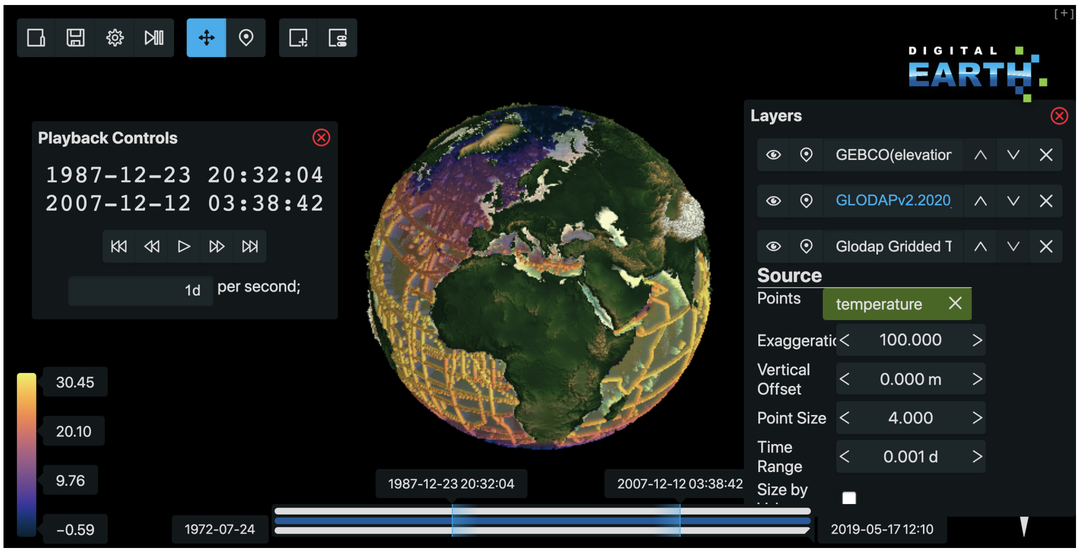


**Figure 3:** *The Earth's globe is recursively subdivided by the spatial tiling pattern with a progressive level of detail in order to be covered by a finite number of tiles.*



1987-12-23 20:32:04
2007-12-12 03:38:42

1d per second;

GEBCO(elevatior
GLODAPv2.2020
Glodap Gridded T

Exaggeratic 100.000
Vertical Offset 0.000 m
Point Size 4.000
Time Range 0.001 d
Size by

**Figure 4:** *The complete user interface of the Digital Earth Viewer includes the main 3D rendered environment as well as child windows for navigation through time and data layer control*

cilities to load data from WMS or Maptile-servers or to interact with SQLite databases. Specialised sources have been built to read Acoustic Doppler Current Profiler (ADCP) data and further application-specific formats. A user interface allows to perform predefined operations on multiple datasets to combine them into new data products.
The viewer is compatible with most web browsers as of April 2021 and its utilisation does not require the installation of any additional software components. Similarly individual server component executables can be configured and deployed in Windows, Linux, and MacOS systems.
The Digital Earth Viewer is conceived as an open source software and its source code will be available on an open repository.

## 4. Resutls

The Digital-Earth-Viewer takes advantage of the interplay between server and client technologies: the server component can handle native code for extracting data from complex file formats in an efficient manner, while the client component uses standardised interfaces for graphics and interaction. This enables the software to display 4D data visualisations in real time. It provides a 3D rendered representation of multiple heterogeneous data sources and its navigation treats each spatial dimension and the temporal component as first class citizens. UI controls in child windows implement functionalities like adding or removing layers in the visualisation, changing visualisation parameters and color schemes, plotting graph elements, and sharing of visualisation results.(fig.4).
Server executables can be compiled for all three mayor operative systems. This allows to deploy locally hosted instances for offline use.

## 5. Conclusions

The Digital-Earth-Viewer is capable of addressing the map-like visualisation tasks of location identification, distance assessment, and path tracing with higher efficacy than many other visualisation solutions. This is because it captures the spatiotemporal nature of geoscientific data more faithfully. The task of data retrieval is augmented by the contextualisation of heterogeneous data sources and by the creation of new data products from user defined combinations. The software deployment as a web-application makes it possible to access and visualise diverse data types commonly used in the Earth Science field from any modern computer. All of these features lead to an improved visualisation of effects, possibilities, and consequences, which in turn leads to the building of models that help better our understanding of the Earth System.

## References

[BAA*11] Baillie, G., Armour, B., Allan, D., Milne, R., Connolly,T.M., Beeby,R. (2011), "A Model-View-DynamicViewModel and its Performance in a Web-based Component Architecture." *Proceedings of the 23rd International Conference on Software Engineering & Knowledge Engineering* 2

[CRA00] Crapo, Andrew W., et al. (2000). "Visualization and the process of modeling: a cognitive-theoretic view." *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining.* 1

[DAL01] Dalrymple, G. Brent (2001). "The age of the Earth in the twentieth century: a problem (mostly) solved." *Special Publications, Geological Society of London.* 190 (1): 205–221.

[FD06] Fraser, S.J., Dickson, B.L. (2006) "Data Mining Geoscientific Data Sets Using Self Organizing Maps" *Australian Academy of Science Elizabeth and Frederick White Conference Mastering the Data Explosion in the Earth and Environmental Sciences* 1

[HSS20] Hogräfer, M., Heitzler, M. and Schulz, H.-J. (2020), "The State of the Art in Map-Like Visualization." *Computer Graphics Forum* 1

[IET14] Internet Engineering Task Force (IETF)(2014), “The State of the Art in Map-Like Visualization.” *Computer Graphics Forum* 2

[MLK10] Mauro, C., Leimeister, J.M., Krcmar, H. (2010), “Service Oriented Device Integration – An Analysis of SOA Design Patterns” *Proceedings of the 43rd Hawaii International Conference on System Sciences* 2

[PHS97] Pyarali, I., , Harrison, T., and Schmidt, D.C. (1997), “Proactor” *4th annual Pattern Languages of Programming conference*

[RUB18] Ruban, D. A. (2018), "Episodic events in long-term geological processes: A new classification and its applications." *Geoscience Frontiers*: 377-389. 1

[SCH02] Schlitzer, R. (2002), "Interactive analysis and visualization of geoscience data with Ocean Data View." *Computers & geosciences*: 1211-1218. 1

[WIJ05] van Wijk, J.J., (2005), "The Value of Visualization" *VIS 05. IEEE Visualization.* 1