SeaDataCloud Virtual Research Environment: Implementation and Technical Aspects

Merret Buurman, German Climate Computing Centre (Germany), buurman@dkrz.de
Charles Troupin, GHER/ULiège (Belgium), ctroupin@uliege.be
Alexander Barth, GHER/ULiège (Belgium), a.barth@uliege.be
Leo Bruvry-Lagadec, IFREMER (France), Leo.Bruvry.Lagadec@ifremer.fr
Sebastian Mieruch, Alfred-Wegener-Institute (Germany), sebastian.mieruch@awi.de
Naranyanan Krishnan, UKRI STFC (United Kingdom), naranyanan.krishnan@stfc.ac.uk
Giorgio Santinelli, Deltares (The Netherlands), Giorgio.Santinelli@deltares.nl
Fedor Baart, Deltares (The Netherlands), fedor.baart@deltares.nl
Peter Thijsse, MARIS (The Netherlands), peter@maris.nl
Filip Waumans, Vlaams Instituut voor de Zee (Belgium), filip.waumans@vliz.be
Themis Zamani, Greek Research and Technology Network (Greece), themis@grnet.gr

What is a Virtual Research Environment?

A Virtual Research Environment (VRE) is a collaborative environment to perform data-driven research. Gone are the days when you had to download all input files to process them on a slow, memory-lacking desktop computer! Instead, perform your analyses with ample data and the latest versions of the software tools on performant hardware in the cloud! The results can be shared and the processing workflow can be reproduced by other users.

The SeaDataCloud VRE (availble to authorized users at https://vre.seadatanet.org/) is a pilot application and supports research with marine data and SeaDataNet tools in the cloud. Its user-facing functionalities are described in the abstract "*Working with the SeaDataCloud VRE: what can we do for you?"*. This abstract focuses on the architecture and technical details.

How is the Virtual Research Environment deployed?

The VRE's components are deployed on servers across various data centres to distribute the load. The shared central components as well as the various processing services are deployed as Docker containers. Containers provide a standardized operating environment across all data centres. This significantly facilitates the deployment and the integration of heterogenous services. The services are loosely coupled to allow for easy extension: The VRE provides central services such as Single Sign-On and central web storage for users, but the services can also run as standalone services.

Shared components: Dashboard, private workspace and file selector

The dashboard component is the heart of the VRE, responsible for managing diverse aspects of the VRE. It provides the main user interface with entrypoints to the different services, and access to the user's private workspace, an online storage space for users to store their own datasets. Behind the scenes, it triggers data synchronisation mechanisms and ensures user authorization for the various services by a token-based authorization system. The dashboard is a web application

based on the PHP framework Laravel, which includes a user management system and state-of-the-art security measures. Marine-ID [https://www.marine-id.org/] is used to provide Single-Sign-On based on EUDAT's *B2ACCESS* service used by several European projects [https://eudat.eu/services/b2access].

The private workspace is a customized instance of the well-known open-source software NextCloud [https://nextcloud.com/], running on SeaDataCloud's servers. **The file selector** is a bridge between most VRE services and the NextCloud, allowing users to select data from their private workspace to be handed over to the processing service instances by the backend.

Two types of processing services

The actual processing services are what researchers use to perform their analyses. Two types of services are deployed in the VRE. The first type of service operates like a normal web server, where one instance serves many users. The *BioQC* tool, the *VIZ-ualisation* tool, and *webODV* are examples for this type. For these services, one Docker container is running continuously and waiting for users.

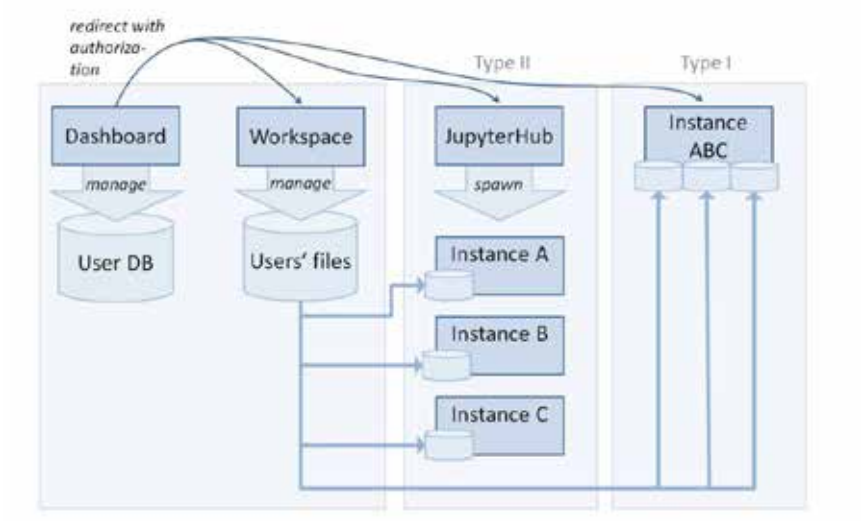

Figure 1: Loosely-coupled structure of VRE services

The second type of service is for the more computationally intensive tools, or for those that are not optimized for handling multiple users' sessions (this frequently occurs when desktop software is ported to the cloud). For this type of service, every user works with an individual instance. As services are packaged as Docker containers, this means that we need to deploy one container per user. We use JupyterHub [https://jupyter.org/hub] to spawn the container when a user logs in. JupyterHub was developed to serve JupyterNotebooks, but it can be used for any tool that is "dockerized" and interacts with the user via HTTP. JupyterHub's benefits include instance management, authentication/authorization, and web security measures such as reverse proxying and SSL termination. This solution is used for the services DIVAnd and ERDDAP subsetting.

DIVAnd is a tool designed to perform interpolation of oceanographic observations onto a n-dimensional grid. The code is written in the Julia programming language and is complemented by a set of Jupyter notebooks. The Docker image is publicly available on DockerHub includes the installation of the Julia language along with packages for plotting and manipulation of netCDF files, the master versions of the DIVAnd Notebooks, which users can adapt and use to prepare data products, and of DIVAnd itself [https://github.com/gher-ulg/DIVAnd.jl].

The VRE's **subsetting service** embeds an instance of ERDDAP [https://coastwatch.pfeg.noaa.gov/ erddap/index.html], configured dynamically and started on the fly according to a dataset selected by the user via the file selector. Its docker image includes the the ERDDAP web application running in an Apache Tomcat web server and a visualization frontend using VueJS framework.

This was a short introduction to the technical side of our VRE. The container-based deployment has proved stable and convenient to maintain. In future, we hope to be able to extend and improve the VRE. Besides incorporating user feedback, possible areas of improvement concern the container orchestration (e.g. by introducing technologies such as *kubernetes* or *Apache Mesos & Marathon*) and the synchronization of data between the data centres.