NEON's Streaming Processing Pipeline for Eddy-Covariance Raw Data

Greg Holling, David Durden, Andy Fox, Hongyan Luo, Natchaya Pingintha-Durden, Cove Sturtevant, and Stefan Metzger

National Ecological Observatory Network, Boulder, Colorado, USA

National Ecological Observatory Network

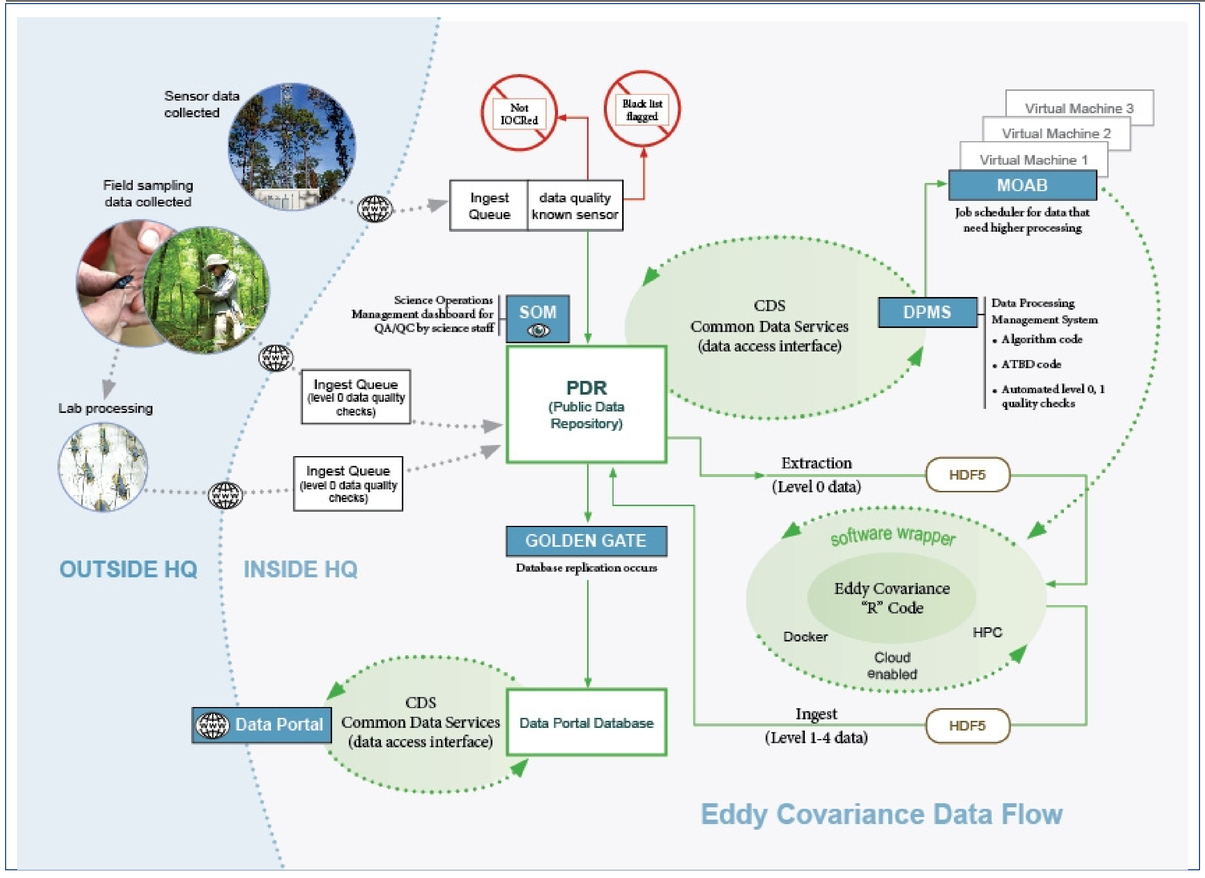
Background

Automatically retrieving, analyzing and storing complex datasets along their contextual information can pose a substantial challenge. Tower networks such as ICOS, Ameriflux, and NEON exemplify this challenge through the growing size of datasets from dispersed measurement sites: Eddy-covariance raw data across the NEON network are expected to amount to 100 Gigabytes per day. The streaming processing and dissemination of this data requires a seamless integration of reproducible, extensible and portable scientific code, in combination with efficient data flows, scalable processing and data discovery tools.

Hence, the eddy-covariance raw data processing at NEON differs from a pre-existing pipeline in some fairly significant ways:

- The eddy-covariance code is being developed by a consortium of scientists and software engineers both inside and outside of NEON; most other NEON data products rely on code solely developed by NEON software engineers.
- The eddy-covariance code is written as publicly available R-packages; most other NEON data product code is written in Java and used NEON-internally only.
- The eddy-covariance R-packages and workflows are compiled into portable, publicly available Docker images and run in parallel Docker containers.
- I/O for the eddy-covariance code uses self-describing HDF5 files; most other NEON code interacts directly with the NEON production database and outputs data and contextual information in separate plain text files.

Data Flow – CI View



eddy4R Source Code

The eddy-covariance source code (eddy4R) is being developed by a consortium of scientists and software engineers both inside and outside of NEON.

The initial development source code resides in a private Github repository and is gradually being released to the public following NEON's data product schedule. Use and contribution are strongly encouraged.

Source code is written as modular R functions and workflows, and provided as a family of R-packages (eddy4R). Source code and dependencies are bundled in a Docker image.

Contact Information: Gholling@BattelleEcology.org

Docker

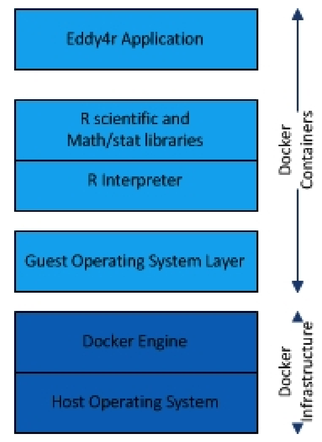
"Docker containers wrap a piece of software in a complete filesystem that contains everything needed to run: code, runtime, system tools, system libraries – anything that can be installed on a server. **This guarantees that the software will always run the same, regardless of its environment**." *What is Docker, from www.docker.com, emphasis added*.

Docker containers are running instances of a Docker image, and are simpler and more lightweight than virtual machines. They rely on shared kernel capabilities of the host operating system, rather than providing wrapper libraries to emulate the guest operating system.

In the case of eddy4R, this guarantees reproducibility. Current and future researchers can be guaranteed that, using the same eddy4R-Docker image and the same input data, they will always obtain the same results, regardless of the host operating system.

One other advantage of using the eddy4R-Docker image is that any number of eddy4R-Docker containers can easily be deployed to various physical environments, including Amazon Web Services or Moab/Slurm or other HPC job schedulers.

Because Docker images are lightweight, they can be freely and easily distributed via a web-based portal such as Dockerhub (https://hub.docker.com/) .The eddy4R-Docker image is based on the Rocker-Docker image (https://github.com/rocker-org/rocker), which in turn is based off an existing Debian-Docker image. As a result, the downloads can be logically separated.



HDF5

HDF5 (https://www.hdfgroup.org/hdf5/) is a platform-independent open-source data model, which can be used to represent very complex data objects and object hierarchies. HDF5 supports a number of compression mechanisms, so data can be represented compactly.

An HDF5 data model is represented as a file. An HDF5 viewer or source code library is required to read the file. HDF5 readers are available for a number of programming languages and operating systems. HDF5 libraries for R and Python are very well supported.

All input and output data for eddy4R is encapsulated in HDF5 files. The output files will contain L1-L4 data for a specific site and time range. These files will be the primary data representation for eddy-covariance data on the NEON data portal.

Moab

Moab (http://www.adaptivecomputing.com/products/hpc-products/moab-hpc-basic-edition/) is used for job control, monitoring, and scheduling. Moab is used to dispatch eddy4R-Docker jobs to configured virtual machines, and to monitor job status.

All of the configured virtual machines will have the Docker client installed, and will download the specified version of the eddy4R-Docker image on demand.

NEON is a project sponsored by the National Science Foundation and operated under cooperative agreement by Battelle.

Battelle
The Business of Innovation