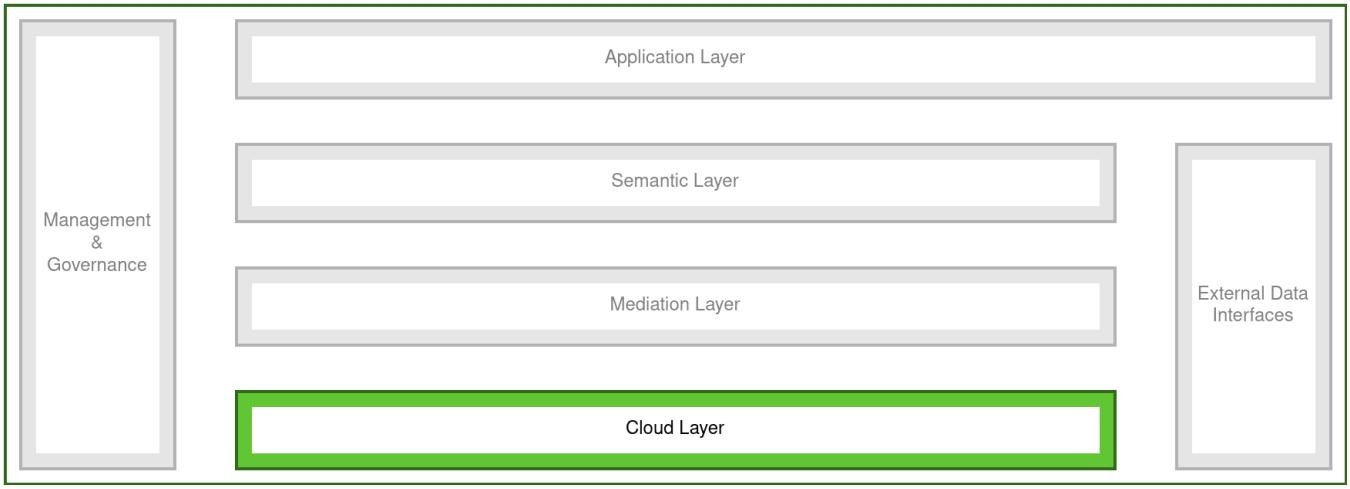
Cloud Layer

The **Cloud Layer** of the RDC is the technical backbone based on a multi-cloud infrastructure, notably consisting of the de.NBI cloud and GDWG. It provides near-infinite storage and storage services such as Aruna Object Storage (AOS), which manages the RDC's raw datasets. In addition, the layer provides basic compute services to the upper layers to facilitate applications to run in the cloud. The cloud layer is not designed for the end user but for those who implement and deliver the services on the upper layers.



Zooming into the Cloud Layer

The Cloud Layer provides support for storage and compute resources. For the storage resources, we distinguish between object storage systems, with its reference implementation Aruna Object Storage, and semantic storage systems, where dedicated database systems are offered to manage the data products of the Semantic Layer. The compute resources consist of tools, such as Docker and Kubernetes, to facilitate the depolyment of services of the upper layers in the cloud.

Object Storage - Infrastructure-as-a-Service

The Aruna Object Storage (AOS), developed at Justus Liebig University Giessen (JLU), is the only implementation of an object storage in RDC. An object storage offers several advantages: As indicated by the name, an object storage organizes data in units of objects (not for example as files as known from a file system). Objects have a system-generated unique identifier and metadata to describe the contents of the objects with key-value pairs. The object creator is responsible for defining access credentials and attaching suitable metadata that describe the contents of unstructured data like text, images, audio, and video and semi-structured data expressed in JSON format to support finding objects. To retrieve objects from the object storage, there are two options. One is to use the unique identifier to retrieve the associated object and the other employs search labels returning all objects with metadata matching one or multiple search labels. Second, one of the key functionality of an object storage is the so-called RESTful APIs to retrieve the objects from the storage via HTTP requests. Thus, it is possible to use either an arbitrary programming language or simply a web browser to access the data. While object stores might be located on an ordinary desktop computer, it is more commonly used in distributed cloud infrastructures. This gives at least three advantages: First, the cloud offers a nearly unlimited storage capacity. Second, it is possible to distribute objects among multiple nodes such that access to objects and processing can be parallelized resulting in performance improvements. Third, the data objects are replicated to avoid the problem of system failures. When one of the nodes in the cloud is out of service, there will be other nodes offering a copy of the objects.

The object storage of a research infrastructure like RDC gives another advantage by sharing objects among many users. Instead of managing an individual copy of an object for each of the users, it is now possible to share the object among users with suitable access credentials. Consider for example a picture collection of dragonflies that might be of interest to many communities, but the locations of threaded species are visible only to a few experts. Moreover, domain scientists do not have to care about the management of systems, but simply use the unified access interface of the object storage of RDC or the higher-level services on top of the services of the Cloud Layer.

In general, object storage requires that objects are static and updates of an object are seldom. However, in biodiversity, there are also highly dynamic data sets such as time series that need to be updated continuously. To address this issue, RDC introduces a version concept that manages static versions, also known as snapshots, of a data set, where one of them is the current version. Instead of applying updates to the actual version immediately, it is possible to collect updates and apply them once when the next version of the data set is created. Versions of data sets are then managed for processing and never deleted from the object store. Thus, such a version approach is also important for reproducibility reasons of research results.

Cloud Computing - Platform as a Service

The RDC is not limited to storage only but strives to offer scalable cloud-based computing services that are easy to deploy and maintain. Examples of such services are specific database systems like PostgreSQL and Elasticsearch that are used for managing the data products offered in the Semantic Layer or specific software tools to create a workflow and to check the data quality provided in the Mediation Layer. While these services will be in the upper layers of the RDC, it is important to offer basic infrastructure tools within the Cloud Layer. Probably the most essential tool is Docker supporting the containerization of services. A Docker container is a standalone, and executable software package that includes everything needed to run a service (within a cloud infrastructure like de.NBI). It encapsulates the dependencies of a service (avoiding version mismatches) and isolates it from the underlying hardware. In addition, Kubernetes and OpenStack support the orchestration and management of Docker containers within a cloud infrastructure. For example, they are responsible for resource utilization, load balancing, and scalability.

Example Components

- Aruna Object Storage (AOS)
- Docker, Kubernetes, and Openstack

Perspectives

Centrally managed Semantic Storage Systems - Platform as a Service

Unlike object storage, a semantic storage system provides functionality for managing and searching specific kinds of data. For example, users are often interested in managing tabular data in a relational database system such as PostgreSQL with its native SQL interface. In addition, PostgreSQL is also well known for managing geospatial data for species observations or environmental parameters. For the search index, Elasticsearch seems to be the most suitable system while Virtuoso is a so-called triple store for managing a knowledge graph. So far, every service is responsible for the installation and administration of these quite complex systems. Moreover, multiple instances of these systems may exist and need to be administrated. To avoid this redundant work, we strive to offer a selection of ready-to-use semantic storage systems within the Cloud Layer such that users only need to create their databases, but are not responsible anymore for the system administration and maintenance. Such kind of centrally administrated systems are not yet implemented and ideally, these services might be also a task for Base4NFDI. We plan to initiate a working group in Base4NFDI for hosting database system services like PostgreSQL, NoSQL database systems (ElasticSearch), and triple stores (Virtuoso) in the cloud.

Associated Services:

- Aruna Object Storage