1st Conference on Research Data Infrastructure

Enabling RDM

https://doi.org/10.52825/CoRDI.v1i.404

Published: 07 Sept. 2023

The Aruna Object Storage

A distributed multi cloud object storage system for scientific data management

Marius Alfred Dieckmann[1‡][https://orcid.org/0000-0001-5130-546X], Sebastian Beyvers[1‡][https://orcid.org/0000-0002-9747-7096], Jannis Hochmuth[1‡][https://orcid.org/0009-0004-4382-4760], Anna Rehm[2][https://orcid.org/0009-0000-5063-486X], Frank Förster[13][https://orcid.org/0000-0003-4166-5423], and Alexander Goesmann[1][https://orcid.org/0000-0002-7086-2568]

[1]Bioinformatics and Systems Biology, Justus Liebig University, Giessen, Germany

[2]Algorithmic Bioinformatics, Justus Liebig University, Giessen, Germany

[3]Bioinformatics Core Facility, Justus Liebig University, Giessen, Germany

[‡] Authors with equal contributions

Abstract: The exponential growth of scientific data has led to an increasing demand for effective data management and storage solutions. Academic computing infrastructures are often fragmented, which can make it challenging for researchers to leverage cloud-native principles and modern data analysis tools. To address this challenge, a new distributed storage platform called Aruna Object Storage (AOS) was developed. AOS is a cloud-native, scalable, and domain-agnostic object storage system that provides an S3-compatible interface for a variety of data analysis tools like Apache Spark, TensorFlow, and Pandas. The system uses an underlying distributed NewSQL database to manage detailed information about its resources and can be deployed across multiple data centers for geo-redundancy. AOS is designed to support modern DataOps practices, including the adoption of FAIR principles. Resources in AOS are organized into Objects, Datasets, Collections and Projects, which represent relations of data objects. Additionally, these can be further annotated with key-value pairs called Labels and Hooks to provide additional information about the data. The system's event-driven architecture makes it easy to automate actions and enforce data validation checks, significantly improving accessibility and reproducibility of scientific results. AOS is open source and freely available via https://aruna-storage.org.

Keywords: data management, storage, FAIR, multi-cloud, cloud-native, data mesh

1 Introduction

In recent years, significant progress has been made in the field of information technology, resulting in decreasing costs for data processing and data storage [1]. At the same time, the volume and importance of data has increased dramatically. Cloud computing infrastructures have gained popularity and are now widely used for data analysis in the commercial sector. This has led to the development of a variety of novel tools

and frameworks. However, many research communities have been slow to adopt cloud computing and have missed out on many of the benefits of these infrastructures and frameworks, partly due to the fragmented landscape of academic computing infrastructures. As part of the NFDI4Biodiversity and NFDI4Microbiota consortia, it was therefore decided to build a new storage platform that integrates the heterogeneous offerings into a common system and allows researchers to manage their data similarly to FAIR Digital Objects [2]. Our goal is to help scientists use cloud-native principles to improve the quality of their data while simplifying data access and analysis procedures.

2 Goals

Our project aims to create a cloud-native, geo-redundant, scalable, and domain-agnostic object storage based data mesh system for scientists [3]. It aims to leverage the heterogeneous computing and storage infrastructures of the scientific community enhanced by additional data management capabilities. In addition, it will support researchers using modern data analysis tools such as Apache Spark [4], Nextflow [5] or Pandas [6] by providing an S3-compatible interface. It is also intended to enable researchers to adopt modern DataOps practices for better and more efficient data handling throughout the data life cycle. Consequently, the system will facilitate the application of the FAIR principles [7]. Primary access to the system should be API-based, supporting OAuth2 and OIDC for authentication and Attribute Based Access Control (ABAC) for authorization [8].

3 Results

Based on these goals, the Aruna Object Storage (AOS) was developed. It is implemented in Rust and provides multiple access methods for end users, such as a gRPC [9] and JSON-over-REST API, as well as pre-built client libraries for multiple programming languages. The system uses an underlying distributed NewSQL database to manage detailed information about its resources (Fig. 1). The database can be deployed across multiple data centers and scaled horizontally to keep pace with the growth of the data stored. Data submitted by users is stored using data proxies, which provide an S3-compatible API with additional functionality to abstract from existing storage infrastructures. This allows a variety of different academic computing and storage providers to be integrated into the system, enabling easy and automated offsite backups and site-local caches, while allowing participants to retain full data sovereignty.

3.1 Objects and higher-level resources

All data uploaded and stored by users is stored as an Object, represented as a sequence of bytes without any semantic information. Once uploaded, these Objects are immutable. Updates create new Objects that reference the original Object, resulting in a history of changes. Objects are organized into Collections and optional Datasets. A Dataset consists of closely related Objects and is used to combine data and metadata for easier access and organization. Collections and Projects, on the other hand, contain a set of Objects and Datasets that represent a scoped view of the data. Collections, Datasets and Projects can be snapshotted, capturing the current state and providing a persistent, versioned identifier. This allows other researchers to accurately reproduce results based on a specific version, while allowing for continuous modification of the current data. All resources and their relationships form a directed acyclic graph (DAG) with Projects as roots and Objects as leaves (Fig. 2).

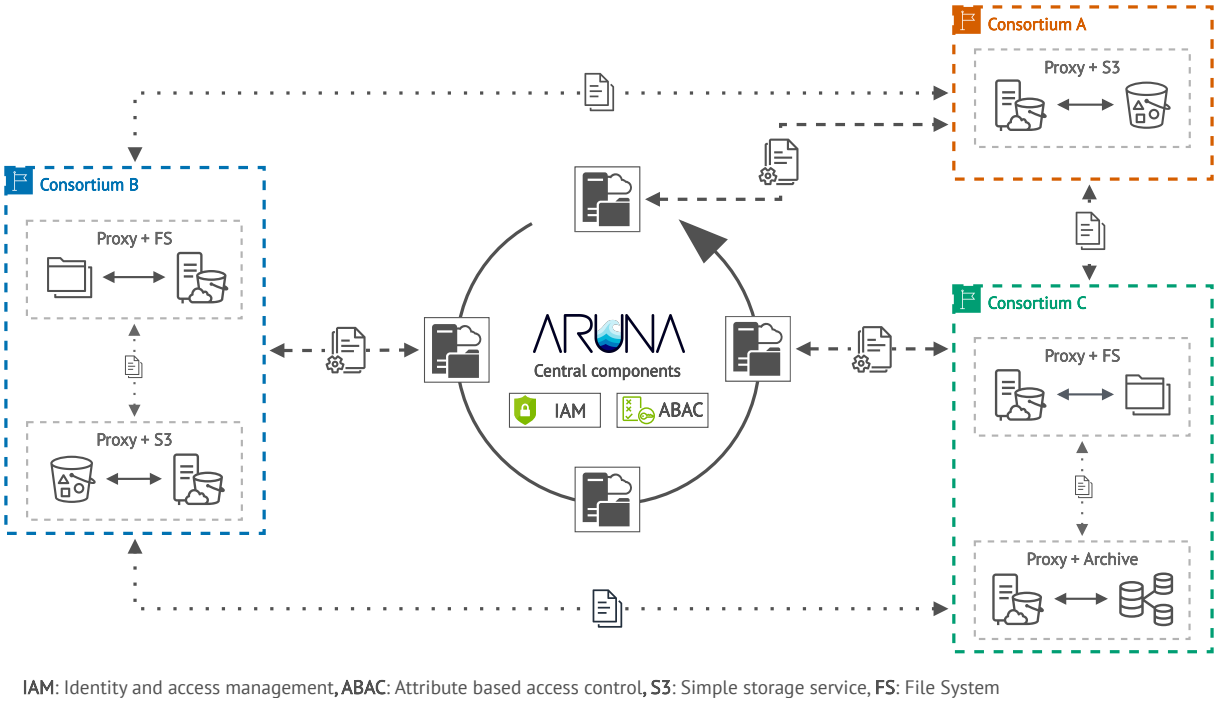

Figure 1. Schematic overview of centralised and decentralised AOS components. The centralised AOS components handle authentication and authorisation by integrating existing IAM providers in combination with user-specific attributes (ABAC). The central components also provide a registry with meta-descriptions and locality information making records discoverable. The decentralised components consist of data proxy applications that expose existing data structures via a common S3 interface and enable data exchange and caching in a peer to peer network within and between participants.

3.2 Data life cycle and events

Resources, and in particular Objects, have their own lifecycle, expressed by states. These can be used to apply certain checks to the data before it is made available. Using webhooks, a user could request that a particular piece of data conforms to a particular standard. After the data is uploaded, the system checks it against the specified standard before making it available. Only if the validation is successful is it made available, otherwise it is placed in an error state that only the user in question can see, without being visible to external users by default. In addition, all state changes emit events that can trigger additional external automation.

4 Discussion

AOS is a modern data management and storage system that has a number of distinct advantages over other tools. First and foremost, it aims to provide a cloud-native solution that is scalable and flexible to connect to a variety of cloud and non-cloud compute and storage infrastructures. By providing an S3-compatible interface, AOS is able to interact with a variety of modern data analysis tools such as Pandas or the Hadoop ecosystem. Its additional features such as labels, webhooks and events make it easy for scientists to adopt DataOps practices to make their work more efficient. Using a

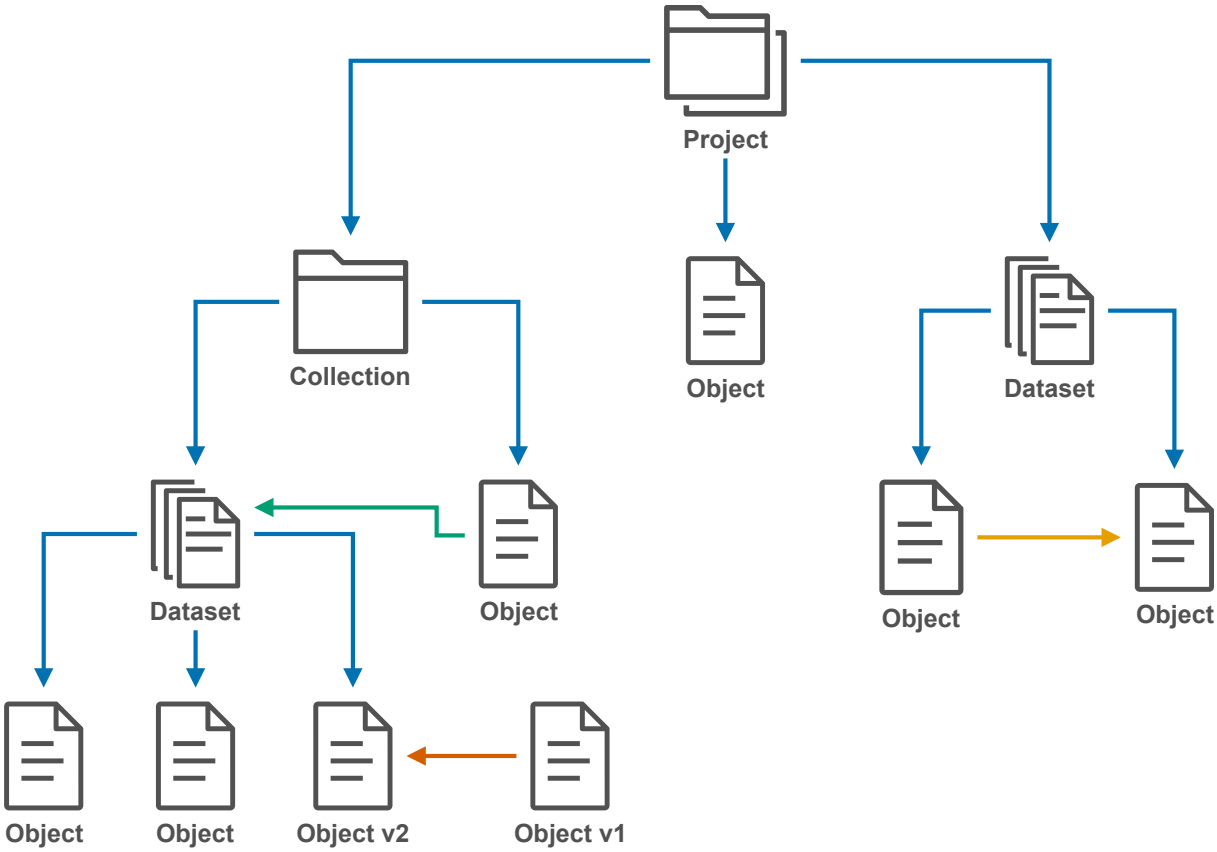

Figure 2. Hierarchical structure of AOS resources. Resources form a directed acyclic graph of *belongs to* relationships (blue) with Projects as roots and Objects as leaves. Resources can also describe horizontal *version* relationships (orange), *data/metadata* relationships (yellow) or even custom user-defined relationships (green).

distributed database that can be deployed across multiple data centers in Germany reduces the risk of the system being unavailable due to problems in a single data center. AOS is open source and freely available; more information, including documentation and source code, is available at https://aruna-storage.org

Author contributions

Conceptualization: MAD, SB, JH, FF; **Data curation:** JH; **Formal analysis:** MAD, SB; **Funding acquisition:** AG; **Investigation:** MAD, SB, JH, AR, FF; **Methodology:** MAD, SB, JH, FF; **Project administration:** MAD, FF, AG; **Resources:** MAD, SB, JH, FF, AG; **Software:** MAD, SB, JH, FF; **Supervision:** MAD, FF, AG; **Validation:** MAD, SB, JH, FF; **Visualization:** MAD, SB, JH, AR, FF; **Writing – original draft:** MAD, SB, FF; **Writing – review & editing:** MAD, SB, JH, AR, FF, AG

Competing interests

The authors declare that they have no competing interests.

Funding

MAD is funded by the German Research Foundation DFG under the National Research Data Infrastructure–NFDI4BioDiversity(NFDI 5/1)–442032008. AR and FF are funded by the German Research Foundation DFG under the National Research Data Infrastructure–NFDI4Microbiota(NFDI 28/1)–460129525. SB and JH are funded via FAIR Data Spaces by the German Federal Ministry of Education and Research BMBF (grant FAIRDS08).

Acknowledgements

Thanks to Dr. Karina Brinkrolf for proofreading the manuscript. We thank Amazon Web Services for distributing the icons used in Fig. 1 & 2 under the CC-BY-ND 2.0 license.

References

[1] C. L. Borgman, "The conundrum of sharing research data," *Journal of the American Society for Information Science and Technology*, vol. 63, no. 6, pp. 1059–1078, 2012. DOI: https://doi.org/10.1002/asi.22634.

[2] K. De Smedt, D. Koureas, and P. Wittenburg, "Fair digital objects for science: From data pieces to actionable knowledge units," *Publications*, vol. 8, no. 2, p. 21, 2020. DOI: https://doi.org/10.3390/publications8020021.

[3] I. A. Machado, C. Costa, and M. Y. Santos, "Data mesh: Concepts and principles of a paradigm shift in data architectures," *Procedia Computer Science*, vol. 196, pp. 263–271, 2022. DOI: https://doi.org/10.1016/j.procs.2021.12.013.

[4] S. Salloum, R. Dautov, X. Chen, P. X. Peng, and J. Z. Huang, "Big data analytics on apache spark," *International Journal of Data Science and Analytics*, vol. 1, pp. 145–164, 2016. DOI: https://doi.org/10.1007/s41060-016-0027-9.

[5] P. Di Tommaso, M. Chatzou, E. W. Floden, P. P. Barja, E. Palumbo, and C. Notredame, "Nextflow enables reproducible computational workflows," *Nature biotechnology*, vol. 35, no. 4, pp. 316–319, 2017. DOI: https://doi.org/10.1038/nbt.3820.

[6] W. McKinney *et al.*, "Pandas: A foundational python library for data analysis and statistics," *Python for high performance and scientific computing*, vol. 14, no. 9, pp. 1–9, 2011.

[7] M. D. Wilkinson, M. Dumontier, I. J. Aalbersberg, *et al.*, "The fair guiding principles for scientific data management and stewardship," *Scientific data*, vol. 3, no. 1, pp. 1–9, 2016. DOI: https://doi.org/10.1038/sdata.2016.18.

[8] E. Yuan and J. Tong, "Attributed based access control (abac) for web services," in *IEEE International Conference on Web Services (ICWS'05)*, IEEE, 2005. DOI: https://doi.org/10.1109/ICWS.2005.25.

[9] K. Indrasiri and D. Kuruppu, *gRPC: up and running: building cloud native applications with Go and Java for Docker and Kubernetes*. O'Reilly Media, 2020, ISBN: 9781492058335.