ASCR Workshop January 2022

# Basic Research Needs for **Management and Storage of Scientific Data**

U.S. DEPARTMENT OF ENERGY | Office of Science

# TABLE OF CONTENTS

# 1. Executive Summary

Data management technologies have, for decades, provided foundational capabilities for scientific computing. Just as storage, input/output (I/O), and data management have been fundamental to simulation-based science for many years, so too are capable data-management technologies key to the success of today's scientific workflows utilizing data-intensive and machine learning (ML) techniques. The U.S. Department of Energy (DOE), Office of Science, Advanced Scientific Computing Research (ASCR) program has invested broadly in data management research focused on high-performance computing (HPC) systems, from parallel file systems that store data to application software that makes these systems more productive. Still, advances in technology combined with growing diversity of supported science strongly motivate continued investment in this area.

In January 2022, ASCR convened a workshop to identify priority research directions in data management for high-performance and scientific computing. Attendees were challenged to identify promising approaches that would support the breadth of the DOE mission, including the explosion of artificial intelligence (AI) uses and the growing needs of experimental and observational science. Technological and science drivers were identified and considered as they relate to key aspects of data management such as interfaces, architectural design, and FAIR (Findable, Accessible, Interoperable, and Reusable) principles. The thoughts of the workshop participants were distilled into a set of four priority research directions with the potential for high impact on DOE science.

**High-productivity interfaces for accessing scientific data efficiently.** A redesign of data access interfaces is critical for locating and accessing data in deep memory and storage hierarchies and across systems (e.g., memory, file systems, archives, online repositories, edge devices, and cloud storage). New interfaces are needed for enabling data management in complex AI workflows. Interfaces are also needed to capture user intent (e.g., metadata and provenance, data usage pattern) for optimizing workflows, performing automated data movement, and extracting important information from datasets.

**Understanding the behavior of complex data management systems in DOE science.** Understanding the behavior of complex data management systems, including user behavior, underlying hardware behavior, and associated compute and networking activities, is key to maximizing the reliability and performance of these systems. Through improved understanding we can eliminate application bottlenecks and unlock the potential of AI to enable the next generation of self-tuning data management services.

**Rich metadata and provenance collection, management, search, and access.** Metadata and provenance are critical for supporting the FAIR principles for reproducible science. R&D efforts are needed to enable management of the voluminous metadata inherent in modern science, to identify metadata and provenance that are effective for supporting FAIR principles, and to understand how to best collect and use metadata and provenance for improving data management systems and scientific discovery as a whole.

**Reinventing data services for new applications, devices, and architectures.** New science endeavors and approaches require specialization of how data are accessed, organized, and retained. New networking and storage devices, including ones with computational capabilities, merit revisiting data service design in order to maximally exploit these technologies. New architectures, including scenarios in which data lives across sites or across administrative domains or is generated at the edge, similarly place new requirements on data services. Co-design of these services with scientists, hardware architects, and facility operators is needed to unlock the potential of data in these unique environments and ease porting to new ones.

In the future, scientific activities will encompass an increasingly broad range of domains and span both HPC resources and advanced scientific instruments. Scientists and facility operators working together to co-design data management architectures will ensure that we have the most capable and robust tools for managing these troves of valuable scientific results. Improvements in how we describe and structure this data will enable greater sharing of data than ever before and will facilitate automation of science with artificial intelligence.

# 2. Background/Motivation

Since the early 2000s, the model of "the parallel file system is the data management system" has been dominant in high-performance computing (HPC) facilities, with file systems such as Lustre and GPFS (now Spectrum Scale) being the trusted persistent store for science data near the platform. At the same time, outside of HPC platforms, various technologies have emerged, including GridFTP and data transfer nodes for moving data between sites, metadata catalogs such as iRODS for finding data across multiple locations, and many different forms of data services (e.g., noSQL, document stores, streaming data services) catering to different use cases. While HPC storage research continued largely to focus on how to make best use of these parallel file systems, other communities moved in new directions.

In September 2018, the U.S. Department of Energy (DOE), Office of Science, Advanced Scientific Computing Research Program convened a workshop to identify key challenges and define research directions that will advance the field of storage systems and I/O over the next 5–7 years. The workshop participants concluded that addressing these combined challenges and opportunities requires tools and techniques that greatly extend traditional approaches and require new research directions.

In the past few years, technologies have matured, the importance of artificial intelligence (AI) has become more obvious, and the needs of experimental and observational science have multiplied. Additionally, the recognition of the value of science data beyond its initial uses encourages us to embrace the challenge of enabling FAIR data principles (findability, accessibility, interoperability, and reusability) [Wilkinson, 2016][Wilkinson, 2019]. At the same time, the high performance, enormous capacity, and resiliency properties that have made HPC storage a success must not be sacrificed. All these factors motivate a re-examination of topics related to data management for DOE science.

# 3. Priority Research Directions

Workshop discussions were organized into four research priorities: (1) High productivity interfaces for accessing scientific data efficiently, (2) Understanding the behavior of complex data management systems, (3) Rich metadata and provenance collection, management, search, and access, and (4) Reinventing data services for new applications, devices, and architectures. Each of these priorities is organized into a subsection below that contains a brief background to the research area, state of the art, and workshop findings. Each subsection also lists priority research directions in the respective area. Themes that crosscut multiple priority directions are discussed at the end of this section: AI for data management and data management for AI, co-design, and the FAIR principles.

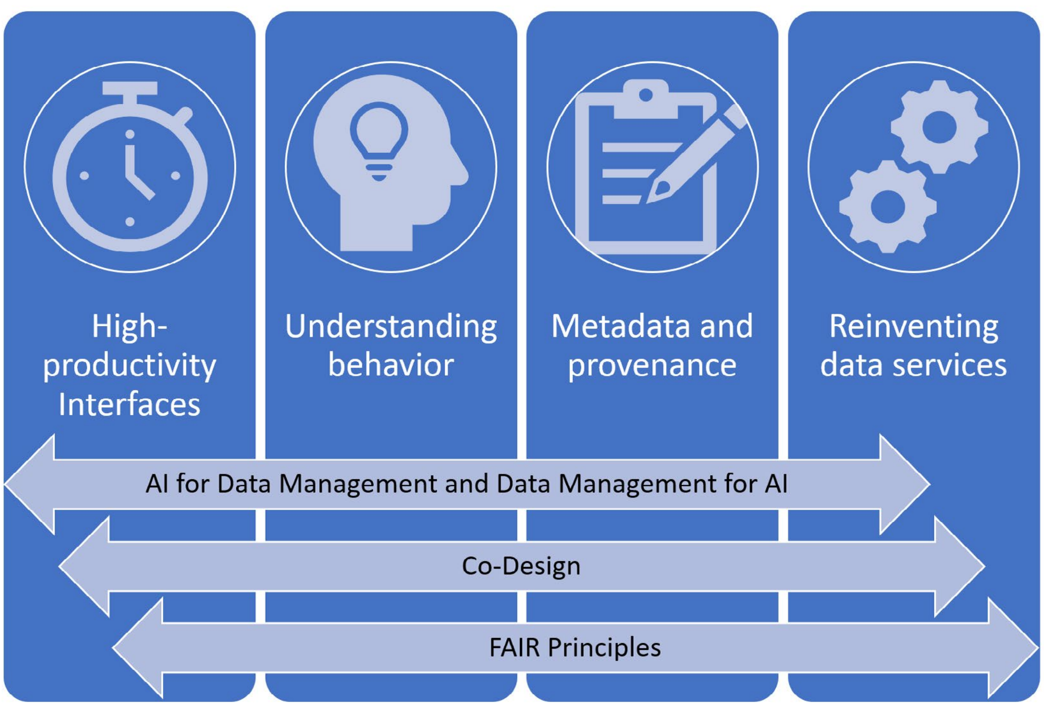


Figure 1: Our workshop identified four priority research directions: (1) High productivity interfaces for accessing scientific data efficiently, (2) Understanding the behavior of complex data management systems, (3) Rich metadata and provenance collection, management, search, and access, and (4) Reinventing data services for new applications, devices, and architectures. We also identified three crosscutting topics: AI for data management and data management for AI, co-design, and FAIR principles.

# 3.1. Priority Research Direction 1: High-productivity interfaces for accessing scientific data efficiently

## Background

A key need identified at the workshop was to scale up and scale out scientific data access to match the scaling of computational science applications. Industry has focused its data-scaling solutions on large collections of small and mostly independent (log or image) items. Scientific data, conversely, is generally highly interdependent and rich in not only current but also future connections. Data storage, transmission, and retrieval are of course a common need in all types of computing, but they are particularly critical in the HPC space where inefficient I/O is a significant impediment to efficient utilization of HPC resources. This situation has become even more critical as the modes of usage of scientific data have exploded with the advent of large-scale artificial intelligence/machine learning (AI/ML), real-time connection of experimental and computational instruments, and the integration of HPC into cloud-scale application environments.

To address this critical situation, we must rethink the user-level abstractions and storage and I/O technologies used for scientific data. As we look forward to large and complex datasets being fed into larger and more complex workflows that contain ML, simulation, and even experimental control aspects, we observe that operations at the level of “read” and “write” are at an unsatisfyingly low level of abstraction for interacting with scientific data. As a result, modern scientific data management solutions have been forced into a number of challenging trade-offs. At the performance level, the POSIX tape-based access patterns give high performance only to specific read patterns. This serialization for performance forces libraries and end users to adopt data structure and metadata tagging on “write” that may be unclear, poorly maintainable, fragile, and even pathological for emerging access patterns, such as AI/ML training for digital twins.

Correspondingly, hardware technologies such as non-volatile memory, object-based storage solutions, and converged network services are leading to new requirements and opportunities when optimizing data management. The evolutions in hardware, changing user needs, and revolutionary new scientific algorithms all underscore the need to rethink the fundamental interfaces for data storage, access, and management.

Accordingly, we identified three core research needs for moving toward a new approach to scientific data management: achieving a deeper understanding of scalable search and access for massive scientific datasets, investigating the new interfaces required to optimize AI and mixed simulation–AI workflows, and better satisfying user intents to contextualize and optimize performance. The state of the art in data interfaces is deep and broad, and we provide a quick summary of some of the relevant examples later in this section. The key supporting conclusions from the basic needs discussions then follow to highlight areas that need fundamental research rather than broader adoption of existing technologies. These findings are then synthesized into a consensus on the three priority research directions at the end of the section.

## State of the Art

The computing environment has become complex with many different interfaces that span the machine architecture. Currently, different low-level interfaces may be found for accessing data within the CPU, within the memory bus, or within other hardware elements such as SATA and NVMe. For software, this interface variety provides the potential for much more access sophistication than does a traditional generic interface. The newer specialized interface systems, including databases of all varieties and more specialized systems such as those developed for cloud-based data analytics, add further access advances. This report provides a brief survey of this vast landscape.

**Low-level interfaces:** At the lowest level in the hardware, modern systems may support DAX (direct access) [LKF, 2014] as a way to accelerate hardware I/O paths. Others rely on NVMe devices [Coughlin, 2013] across the PCIe bus [Mayhew, 2003] for fast device access. Older technologies, such as SATA [Grimsrud, 2003], may host cheap devices but with strict performance limitations. Each of these low-level interfaces, from DAX to SATA, offers faster access but for more limited devices and hardware support. Specialized software interfaces, such as SPDK [Yan, 2017] or PMDK [W. Wang, 2018], are needed to get full advantage of offered bandwidth. In general, NVMe devices currently dominate without much of the software able to take advantage fully of available bandwidth.

The next level up the stack provides a storage system interface that offers support for abstracting away the hardware specific characteristics to something meeting the general POSIX interface. Examples include zfs [Rodeh, 2003], xfs [Wang, 1993], and ext4 [Mathur, 2007]. These interfaces are also used for container systems, since a container image is a file system in a file.

Above this are the storage systems typically used for HPC systems. These include traditional systems that primarily use a POSIX interface such as Lustre [Braam, 1999], SpectrumScale (GPFS) [Barkes, 1998], and OrangeFS [Bonnie, 2011]/PVFS2 [Latham, 2004]. These are parallel storage systems focused on offering support for very large single files striped across many different storage targets to gain aggregate performance. Distributed storage systems, such as HDFS [Mackey, 2009], Ceph/RADOS [Weil, 2006], and WekaIO [Liran, 2018], focus on supporting simultaneous access to files from many locations simultaneously. The former systems tend to be write performance optimized, while the latter are more read performance optimized.

Newer systems have emerged that focus on niches, such as burst buffers, where reliability may be less critical since data will migrate off these devices shortly after being pushed there [Tang, 2017]. BeeGFS [Heichler, 2014] offers a commercial grade system while GekkoFS [Vef, 2018] offers a more experimental system.

**Rethinking the linear array of bytes model:** Systems such as MadFS [Lu, 2009] and OceanStor [Huawei, 2018] are further rethinking the storage stack. MadFS builds on the ideas of GekoFS but uses a distributed key-value store for metadata and a node-level storage system, such as zfs, to store the data blocks.

Key-value stores and object stores in general include systems such as MDHIM [Greenburg, 2015], DAOS [Lofstead, 2016], and pMEMCPY [Logan, 2021]. The cloud interface has largely settled on S3 [Amazon, 2006] given the early dominance of Amazon's cloud platform. More exotic systems, such as Labios [Kougkas, 2019], may use a different access system. In this case, a label/tag is added to an I/O request that the system can then schedule to best manage I/O performance overall.

**Moving from *serial* to *parallel* and *federated:*** HPC has a long history of work that addresses the shared file pointer problem of POSIX. POSIX I/O provides the traditional I/O interface to read and write data from/to files. However, it is traditionally designed for files to be accessed serially–it lacks support for multiple processes/ranks writing data concurrently. Thus, HPC applications that use POSIX I/O have to explicitly address how to store distributed data structures and avoid contention issues. In general, using POSIX I/O directly in HPC applications is considered cumbersome. The MPI-IO interface provides an API for concurrent access to a file. Collective functions allow multiple MPI ranks to participate in a function call. Ranks can describe access to non-overlapping regions of the file, and the underlying MPI library performs the actual I/O and coordinates access to the file.

Sophisticated libraries and data management frameworks such as ADIOS [Lofstead, 2008], HDF5 [Folk, 2011], and PnetCDF [Li, 2003] provide features to store distributed data structures to files. They provide collective function calls that are used to describe the decomposition of data among MPI ranks. In addition, they provide custom self-describing file formats to store data. Different libraries differ in the design of their abstractions and data formats. HDF5 provides a hierarchical data format in which raw data is written into HDF datasets, and datasets can further be grouped into HDF groups. It supports the MPI-IO interface for concurrent access to a HDF5 file. An HDF5 file is represented on a file system as a single file. However, its Virtual Object Layer (VOL) [HDFGroup, 2012] allows storing data in a different format so that users can use the HDF5 abstraction with the option of storing data in a different format. ADIOS provides a publish-subscribe interface for reading and writing distributed "variables," which are typically used to represent science quantities such as temperature and velocity. Its pub-sub interface allows data to be written to files or streamed in-memory to other processes or streamed over the wide-area network. Like HDF5, this is a pluggable interface. An ADIOS file is a container; it is represented by a directory on the file system consisting of raw data and metadata files. PnetCDF is a 64-bit extension to the NetCDF3 [Rew, 1990] API/format widely used in the climate community. Maximum backwards compatibility and similarity was the goal imposing some limitations, such as adding a new variable to an existing file is expensive, while support for "no value present" offers a way to avoid consistency issues when data is absent. NetCDF4 [Rew, 2006] is built on HDF5 to achieve similar goals.

**Using *object* as a fundamental construct:** While file systems manage data using a hierarchy of files, object stores store data as objects that are identified by unique identifiers. Popular object-storage based file systems used in HPC centers include the Ceph storage system and the Lustre file system. For Ceph, the RADOS layer can be exposed as a native object store, whereas with Lustre, the object storage is hidden behind the POSIX API. With the advent of non-volatile memory (NVM), interfaces such as Intel's DAOS have emerged to provide an asynchronous key-value store on top of commodity NVM. UNITY provides a data storage abstraction that places the entire memory hierarchy, including both traditionally separated memory- and file-based data storage, into one storage continuum using a publish/subscribe model based on objects [Jones, 2017]. Research-oriented tools such as Hermes [Kougkas, 2018], DataElevator [Dong, 2016], and UniViStor [T. Wang, 2018] provide a transparent way to utilize the tiered storage hierarchy on modern supercomputers that consists of system memory, local and remote NVM devices, parallel file systems, and tape devices. Proactive Data Containers (PDC) [Byna, 2018] provides an object-centric API that supports asynchronous and transparent data movement in memory and storage hierarchy [Tang, 2018][Mu, 2020]. Complicating this strictly stacked memory/storage hierarchy is accelerator-specialized memory/storage, such as GPU memory. Interfaces such as NVIDIA GPUDirect Storage (GDS) [Ravi, 2020] are opening up these memory tiers for more direct access with the greater storage hierarchy and are becoming increasingly important as the use of accelerators continues to expand. With the growing popularity of cloud computing, systems such as Amazon's S3 and interfaces such as Kubernetes [Mcluckie, 2014] have emerged to provide automated provisioning, deployment, and scaling of applications. For portable execution of applications on a variety of platforms and infrastructures, container technologies such as Singularity [Kurtzer, 2017] and Docker [Turnbull, 2014] are being used by science teams.

**Contextualizing data:** Database technologies provide a highly structured way to store and retrieve data. Relational databases are used to design a schema to store and associate data and objects. Access to data is provided through the Structured Query Language (SQL) [Chamberlin, 1974], a declarative language designed for querying and modifying database systems. No-SQL (Not Only SQL) databases are used for storing larger objects that are unsuitable for expressing in a relational schema. The development of No-SQL databases was pushed by the ability to relax consistency and tolerate having parts of the total database be unavailable and still generate an acceptable result. Systems such as Cassandra [Hewitt, 2010] focus on a fixed set of columns representing common values with an essentially limitless set of additional columns that optionally contain other data—in effect, a set of explicit, valued tags on rows that can offer more information but cannot be used for optimized data selection (i.e., searching by these columns requires a table scan). MongoDB [MongoDB, 2009] offers a way to store documents alongside a set of standardized attributes more easily than a traditional RDBMS.

In-memory distributed data stores such as Redis [Kakola, 1996] focus on accelerated access patterns using a key-value structure. Key-value stores, in general, offer a way to have a large data store that is more resilient to failures in the namespace hierarchy. With a hierarchical namespace, failures in servers that host metadata toward the root may block access to data lower in that part of the hierarchy. With a key-value store, simply the portion of the key space hosted by that server will be unavailable. Locking similarly is localized.

**Interfacing with various analysis tools and in-memory tools:** Tools and ecosystems built around the R [Venabels, 2000] and Python [Sanner, 1999] programming languages are used heavily for data analysis and visualization. Traditional interfaces include managing data in comma-separated values (CSV) format, whereas modern interfaces include tools such as Pandas [McKinney, 2011] and dataframes [Embley, 1980] that provide a tabular-like interface to data. Pandas provides a dataframe-driven object interface to data and can interface with a variety of storage formats that includes CSV files and SQL databases. Apache Arrow [Apache Arrow] defines a language-independent columnar memory format that can be used as an intermediate format for interoperability between several tools and storage formats. Jupyter [Perez, 2007] notebooks are increasingly being used in the DOE community and national laboratories for reproducible analysis of data in spite of how fragile they are for this purpose [J. Wang, 2020].

More feature-rich and scalable tools for analysis include the Map-Reduce framework for parallel processing of big data, and frameworks such as Apache Kafka [Kreps, 2011] offer a message-queue style interface intended for widespread distribution rather than persistent storage. Unlike storage systems with consistency guarantees, Kafka and similar systems offer only best effort, which is sufficient for their target environments. Use of these kinds of systems when data delivery must be guaranteed is currently not advised, limiting their applicability in many important HPC workloads [Wu, 2009].

## Workshop Findings

The workshop discussion covered many areas of scientific interfaces. Participants noted several key themes around scientific data interfaces, including changing application types, difficulty mingling scientific data models with legacy I/O interfaces, emerging challenges in using modern storage hierarchies, and more complex and performance-critical application couplings.

**Expanded roles for HPC:** Participants noted that applications are diversifying from traditional simulation and modeling methods toward complex workflows involving unstructured data, machine learning, and higher-level programming models. These pose challenges because they are often incompatible with each other and with traditional programming techniques. New applications that combine, for example, high-performance numerical simulation (e.g., for climate modeling) with record-oriented datasets (e.g., for population modeling) stored in databases would mix data access interfaces and create programming complexities. Researchers desire to rapidly integrate datasets, software libraries, and algorithmic techniques but are stymied by poor support for new techniques in HPC environments.

New scientific interfaces are needed to continue development of capabilities that bring HPC closer to users and scientific instruments. Currently, HPC systems are heavily isolated from external data sources and sinks, but this situation is changing as new distributed systems are being developed that support the security protocols required. This implies that application developers face additional interface challenges. Emerging scientific instruments will be capable of generating data at rates that outpace computer hardware performance gains, necessitating a careful strategy of filtering, compression, feature extraction, and so on that will be common across many application types. Participants discussed how better scientific interfaces and reusable capabilities could accelerate the integration of HPC with advanced scientific instruments, improving the utility of both systems. More generally, edge computing applications that connect HPC sites to devices and users outside the HPC complex could benefit from improved interfaces, as non-traditional data access models such as streams, subscriptions, and novel human-computer interfaces challenge existing capabilities.

**Relegating POSIX:** The POSIX interface is used to move memory-resident data to and from persistent storage hardware. Participants noted, however, that the changing hardware environment poses challenges to this familiar two-mode model. Modern systems consist of a complex hierarchy, including accelerator devices and associated memory, traditional RAM, node-local high-capacity storage devices, parallel file systems with varying performance capabilities, and archival tape storage. Developing portable, efficient applications against all of these system features is currently impossible. Participants suggested that unified programming interfaces, declarative data approaches, advanced runtimes, and workflow systems could be applied to improve this situation. Such efforts are seen as critical because exascale and post-exascale systems may become increasingly reconfigurable, able to be specialized to particular scientific learning tasks, and hardware will become more diversified, with rapid changes and dynamic deployment roadmaps.

The default approach provided by HPC vendors is the traditional POSIX interface for accessing disk-resident data. Participants noted, however, that this model has many limitations for research today. Its consistency model, impossible to get around, does not always enable the best behavior for large-scale computing. Its metadata model is also often a performance bottleneck, while not being extensible enough for modern metadata goals such as FAIR. Thus, higher-level third-party libraries are brought in, but often there are gaps and incompatibilities when application data models interact with the high-level models provided by the libraries. Similarly, databases (SQL or otherwise) face challenges in HPC contexts for many reasons and result in similar programming challenges.

**Elevating data lifecycle**: The relationship of application teams with data is changing, and new conventions are becoming the norm. Providing one's data along with one's written publication has become standard practice in scientific research. An implication of machine learning is that the purpose of data may be more for machine consumption than human consumption, or some combination thereof. Traditional computational studies commonly had simple patterns from creation to analysis and publication. Today, the lifecycle of data is becoming more complex, with additional stages of reuse, sharing, learning, and validation. In modern scientific campaigns, data moves across an expanding scope of community and custom tools, in which downstream reuse by other users and systems is becoming more critical, and the end goals of publication and reporting may be secondary to such a team.

**FAIR as the new paradigm**: The increased importance of the data lifecycle and the expanded roles of HPC are underscoring the increasing importance of FAIR (findable, accessible, interoperable, and reusable) [Wilkinson, 2016] conventions. Participants noted that there is limited support for these features in vendor-supplied filesystems; thus, databases and other systems must be integrated into the data ecosystem to provide these features. Such metadata can also inform automated systems about the intended purpose of datasets and thus how to best access and modify them in the future. These capabilities are needed in light of overall computer system complexity but are intractable using the default systems of today. Participants described how advanced metadata capabilities (e.g., search, sharing, reproducibility, annotations) will be critical for data-driven studies going forward. FAIR is particularly critical for AI consumers of data [Fagnan, 2019], and the AI-for-science process will require and may generate large volumes of high-quality, well-curated scientific data.

**Co-design:** Participants discussed that co-design is needed to span necessary high-level data model capabilities, while enabling users to accurately specify what operations the underlying system should perform. A variety of low-level capabilities could be brought to bear on scientific problems, including native support for indexing, key-value databases, and coordination and consistency features. Co-design is needed to explore how high-level patterns can utilize these capabilities without unnecessary overheads and programming complexities.

The future of HPC systems is likely to be highly dynamic. Scientific applications teams will need appropriate tools to manage sudden changes in computing capabilities as well as scientific project goals. Scientific data interfaces anchor the programmer to the computing environment and will need to become more flexible to enable the dynamic computational investigations and data explorations that will generate insights and breakthroughs in large-scale scientific problems. The fraction of software written by scientific teams continues to shrink, and a range of community tools must be brought in to manipulate, integrate, analyze, and learn from data. Rapid prototyping applications will be possible only if improved data interfaces make these couplings low-effort and efficient.

**Employing helpful abstractions and models:** Another challenge is the underlying data models. In many cases these models evolve over time and are kept for compatibility. However, creating data models supporting the particular use cases and automatic transformation between models (and representations) can improve the way data is used.

The same is true for the metadata. In many cases no metadata models or ontologies are available, or they are very domain or application specific. Improving collection and usage of metadata will greatly benefit from common ontologies.

**Expanding the notion of *"data center."*** In current HPC systems, data is often an afterthought and managed by the user. Changing this model, by putting data and data management center stage, will lead to the design of HPC systems around the data itself, rather than the (legacy) applications that are/have been dominating the HPC space. Transforming the HPC centers into true "data centers" and providing the necessary tooling to work with the data have a great potential to accelerate new discoveries. Future data centers could help advance science by providing their user community with "data fusion" in/for multimodel simulations, by adding content-based addressing instead of traditional location-based addressing, by automatically managing seamless data movement, and by providing data versioning/lineage on a selectively chosen granularity.

## Summary of Priority Research Direction 1

High-productivity interfaces for accessing scientific data efficiently:

1. How can application developers search and access important information seamlessly in massive amounts of scientific data?
2. What changes are needed to existing I/O application programming interfaces (APIs) to enable complex AI workflows?
3. What are effective interfaces and abstractions for capturing user intent for optimizing data management?

A redesign of data access interfaces is critically important to locate and to access data in deep memory and storage hierarchies and across systems (e.g., memory, file systems, archives, online repositories, edge devices, and cloud


Figure 2. Our workshop noted several key themes around scientific data interfaces, including changing application types, difficulty mingling scientific data models with legacy I/O interfaces, emerging challenges in using modern storage hierarchies, and more complex and performance-critical application couplings. The above figure illustrates three research areas that were identified as core needs in order to move toward a new approach to scientific data management.

Back to TOC

storage). New interfaces are needed for enabling data management in complex AI workflows. Interfaces are also needed to capture user intent (e.g., metadata and provenance, data usage pattern) for optimizing workflows, performing automated data movement, and extracting important information from datasets.

# 3.2. Priority Research Direction 2: Understanding the behavior of complex data management systems in DOE science

## Background

Advancements in data management and storage technology are not possible without first establishing a firm foundation for observing and *understanding* I/O behavior. This is true not only for computer science researchers but also for end users, facility operators, and any practitioner of data-intensive scientific computing. Extraction, storage, and analysis of I/O instrumentation enable these stakeholders to measure performance changes, identify root causes, make better use of resources, and interpret performance in a broader system context. Understanding I/O behavior is an active field of research with a successful track record in improving scientific productivity, but much work remains. Scientific workloads are diversifying to leverage new runtime systems and computational techniques (e.g., AI, workflows, and big data) while the systems themselves are growing in scale and complexity (e.g., additional storage hierarchy layers and new services) to meet the demand. At the same time, the community is facing pressure to more effectively extract actionable results from instrumentation in practice with minimal effort. These factors and others present an array of challenges and opportunities for DOE research.

## State of the Art

Understanding I/O is a broad topic that encompasses many aspects of scientific data management. We begin our summary of the state of the art by surveying the storage technologies that we need to understand from two perspectives: directional data movement and layers of software. We then consider crosscutting, purpose-built methodologies and resources that can be used to enhance our understanding of those technologies.

## Direction of data movement.

Data movement in HPC systems can be conceptualized in two dimensions as illustrated on the left-hand side of Figure 3. Vertical data movement refers to movement of data within a node using locally attached storage resources (fine grained) or remote storage resources (coarse grained). Horizontal data movement refers to movement of data across nodes within a single platform (fine grained) or across platforms (coarse grained)

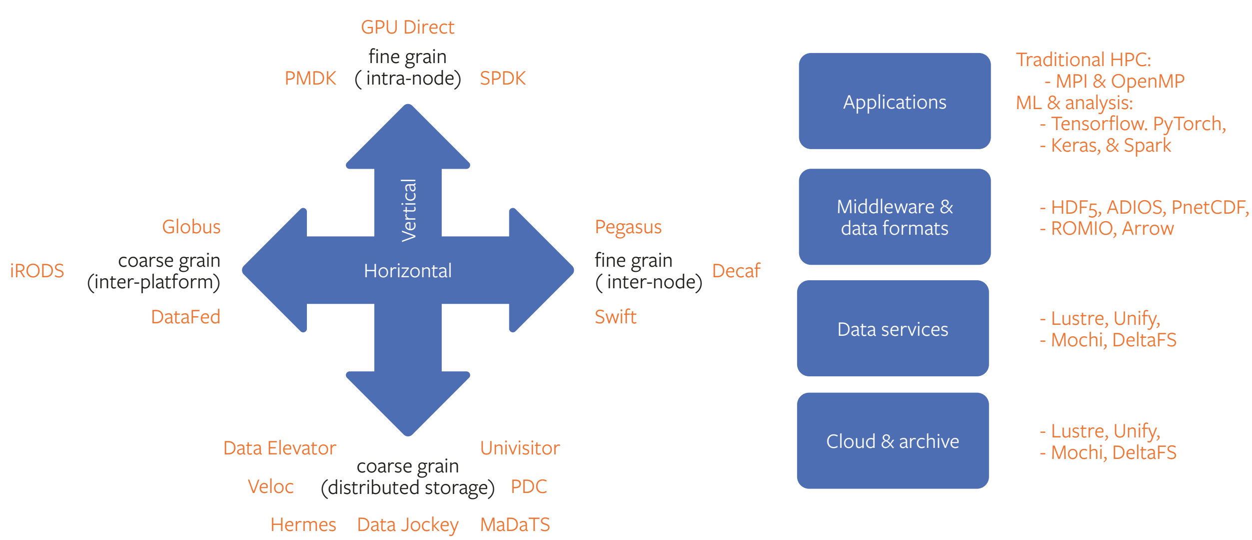


Figure 3: A high-level taxonomy of HPC storage infrastructure in terms of directional data movement and layers of software. Exemplar state-of-the-art technologies are shown in orange.

Vertical fine-grained data movement is driven by the need to effectively harness emerging hardware, including NVMe, persistent memory, and computational storage, whereas vertical coarse-grained data movement is driven by the need to simplify usage of complex storage systems. Horizontal fine-grained data movement is driven by the need to transfer data between workflow components, whereas horizontal coarse-grained data movement is used to federate datasets and resources that span HPC facilities [da Silva, 2021].

Data management involves methods for efficient data access to and from the storage system, but it also includes any data movement within modern scientific workflows. With data staging, I/O-intensive coupled applications can run in steps, exchanging large amounts of data between each step without saving all steps to permanent storage. This is the case, for example, for a visualization and analysis application running in parallel with a simulation. Similarly, scientific instruments can generate a large volume of data at high velocity in parallel with digital twins or analysis that must parse the data during the experiment or before the next experiment (e.g., the Korea Superconducting Tokamak Advanced Research (KSTAR) fusion reactor [R. Wang, 2020] ). Industry software, such as Apache Flink [Katsifodimos, 2016], has been designed to offer continuous streaming eliminating periodic import and query execution in order to perform analytics in a real-time fashion. Similarly, numerous current I/O libraries designed for HPC are able to offer streaming for various applications (e.g., ADIOS [Kube, 2021], MPIStream [Peng, 2014], Decaf [Dreherand, 2017], Henson [Morozov, 2016]).

**Layers of software.** The right side of Figure 3 conceptualizes HPC storage technologies in terms of layers of software used by applications to store and retrieve their data. The application layer includes a growing diversity of runtime systems. In previous decades, HPC applications were dominated by message passing (e.g., MPI) and node-level parallelism (e.g., OpenMP), but today's application portfolio has expanded to leverage breakthroughs in machine learning and data analytics. This diversity of applications is in turn supported by a rich ecosystem of middleware and file formats that simplify organization of and access to scientific data. The middleware components use distributed data services that enable high-performance concurrent access to a collection of distributed storage devices. The complete data management lifecycle includes not only archival systems for data stewardship but also cloud storage resources that allow scientists to bring additional computing resources to bear on their problems.

## Crosscutting technologies for better understanding

**Instrumentation and profiling**. A rich ecosystem of computational instrumentation and profiling tools is routinely used to extract as much productivity as possible from constrained computational resources. Examples include parallel application instrumentation tools such as Tau [Shende, 2006], Pin [Luk, 2005], HPCToolkit [Adhianto, 2010], and Score-P [Knüpfer, 2012]; node-level tools such as nvprof [NVIDIA, 2022], Intel VTune and other tools from the Intel PAT suite [Intel, 2022; and platform-wide system data tools such as LDMS [Agelastos, 2014]. Most of these include some degree of I/O instrumentation support, but the community often turns to purpose-built I/O instrumentation tools such as Darshan [Carns, 2011] and Recorder [C. Wang, 2020] to understand HPC I/O behavior in depth. Darshan is a transparent modular tool for capturing concise summaries of I/O behavior. Recorder is a multilevel I/O tracing tool that captures I/O access in fine detail.

Specific platforms, storage systems, and frameworks may also provide in-depth but less generalizable instrumentation capabilities. For example, the Lustre LMT tool [LMT, 2022] can be used to understand file-system-level detail. In recent years storage vendors have also provided proprietary tools that have advanced features but do not necessarily interoperate easily with open-source tools. Middleware libraries may provide capabilities such as ADIOS's Skel [Logan, 2012] to understand middleware usage and reconstruct workloads for further study. Modular APIs such as the HDF5 VOL [Byna, 2020] interface have also opened up opportunities to interpose new instrumentation methods. Emerging machine learning frameworks provide capabilities such as TensorBoard [Tensorboard, 2022] as well, although they are not well integrated with HPC platform tool chains at this time.

**Analysis of instrumentation data**. Once data has been collected, analysis and visualization tools are needed to derive intuitive findings from the data. Each of the instrumentation and profiling tools described above includes at least some limited capabilities in this area. Additional tools can be layered on top of collected data for advanced functionality such as interactive trace navigation (e.g., DXT explorer [Bez, 2021]), platform-level context (e.g., TOKIO [Lockwood, 2018]), clustering and comparison of workloads (e.g., Gauge [Del Rosario, 2020]), and nested workflow behavior (e.g., IOBAT [IOBAT, 2022]). In addition to tool development, research activities have included methodologies for analysis of I/O data using graph techniques [Dai, 2016] and statistical learning

Back to TOC

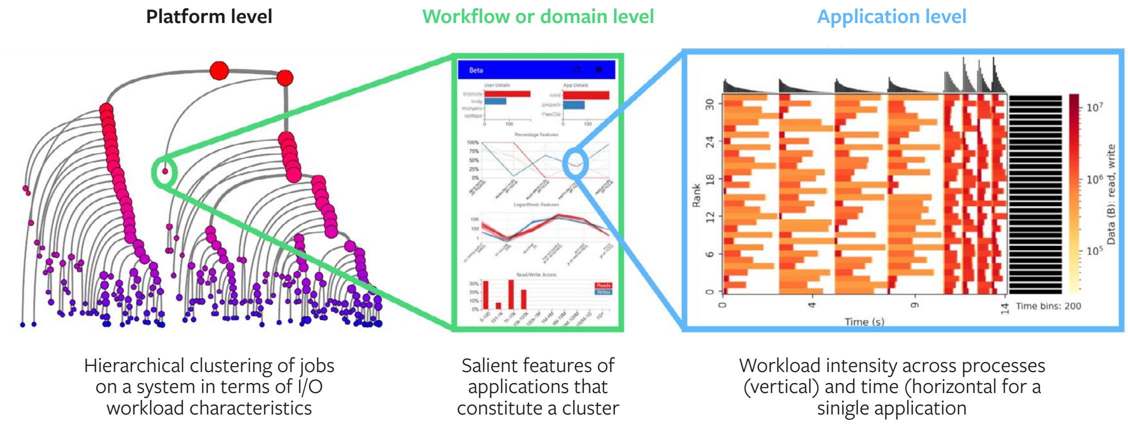


Figure 4: Analysis of instrumentation data: this conceptual figure shows how analysis and visualization methods can be used to navigate and link platform-level workloads, workflow or domain-level workloads, and application-level workloads. Different perspectives on the same data set may be necessary for different stakeholders or use cases. Figure credits: Isakov et al. [Isakov, 2020] (left), Del Rosario et al. [Del Rosario, 2020] (middle), Awtrey et al. [Awtrey, 2021] (right).

and ML modeling [Isakov, 2020][Costa, 2021][Patel, 2019] [Madireddy 2018][Agarwal, 2019][Xie 2021].

**Active benchmarking and probing**. In addition to observing I/O behavior, one often wishes to be able to *reproduce* I/O behavior in a controlled fashion for "what if" investigations. The state of the art in this area involves a variety of synthetic benchmarks such as ior [Hedges, 2005] and fio [fio, 2022], API-specific benchmarks such as h5bench [Tonglin, 2021], AI-oriented benchmarks such as dlio [Devarajan, 2021] and mlperf [Reddi, 2020], and application proxies such as MACSio [Dickson, 2016]. The Wemul [Chowdhury, 2020] framework provides a system for reproducing broader workflow I/O patterns. Any of these benchmarks could also be harnessed for use in recurring regression tests or probes of system behavior to observe platform variability over time [Lockwood, 2018].

**Data repositories**. The final piece for understanding I/O-related work is the ability to archive characterization data for later use. Such data could be leveraged to understand platform trends, apply new analysis techniques to existing data, or enable research by teams that do not have direct platform access. Examples of existing public repositories include the SNIA traces [SNIA, 2022], the anonymized Darshan data repository [ALCF, 2013], and large-scale disk health monitoring data [Lu, 2020]. Active effort is also under way in methods for standardizing data, such as the Common Workflow Language project for workflow systems [Crusoe, 2021]. Recent FAIR initiatives have focused on how to effectively share scientific data [Wilkinson, 2016], but advancements in that space can often equally apply to computer science data and computational workflows [Goble, 2020]. As in other fields, privacy and security concerns often dictate data repository functionality.

## Workshop Findings

The recognition that workflows, not individual jobs, ultimately drive scientific discovery expands the scope of understanding data movement and I/O performance. The state of the art in tools that reflect this workflow-oriented view has not caught up, and significant effort is required to solve the "data fusion" challenge of multiple connecting job-level insights into a holistic end-to-end view of data movement. For example, in order to minimize runtime overhead, application profiling tools often rely on summary statistics rather than real-time tracing, while system-level tools often generate time series data since they do not have insight into the boundaries between distinct user workloads. Workflow composition is fundamentally heterogeneous in time, however, and reconciling point-in-time or aggregated statistics application with time-resolved system statistics, especially amid the backdrop of large, eventually consistent systems, requires new approaches.

The nature of data movement is also aggravating an innate tension between simplifying data management and understanding data movement. In complex, tiered storage hierarchies, the goal of data management systems is to hide the underlying complexity to the greatest extent possible and ensure that users' data is in the right place at the right time. Achieving this requires the system itself to move data transparently, making it difficult or impossible to understand where data accesses are originating, what data paths are at play, and how these factors are affecting the perceived performance. Bridging this gap—simplified data management without compromising the intuitiveness of performance—remains a difficult challenge.

Another obstacle to better understanding of workflows is the limited ways in which users can express their full workflow. HPC has long provided imperative primitives to launching individual jobs, but such approaches limit system's ability to optimize data movement and management because the system is largely reactive to imperative operations and, at best, must guess about the most optimal data placement or I/O path for the next imperative statement. Providing declarative, intent-based means to express workflow construction would enable new end-to-end optimizations that the storage system could employ and would provide much better interpretability of the optimizations it makes. Intent-based workflow specification also enables I/O researchers to better understand user needs as workflows change without having to guess the goals of different imperative data management operations on any given system.

Even if these challenges—obtaining unambiguous, quantitative data across an entire workflow and being able to effectively combine the data—were overcome, such rich data generates better understanding only when combined with expert knowledge in what such data means. Translating that understanding into actionable steps for improvement is a step beyond, and deriving such actionable insights from data is currently heavily reliant on humans in the loop. In addition to the obvious fact that I/O expertise is in short supply relative to demand, such human experts' efforts are often trapped in boutique solutions developed for specific communities or systems. This challenge only gets worse as storage systems and workflows generate more telemetry data, underscoring the need to develop more human-scalable ways to connect rich data sources to actionable outcomes.

### Summary of Priority Research Direction 2

Understanding the behavior of complex data management systems in DOE science

Key Questions:

1. How can disparate information from multiple sources regarding data management activities be fused into useful knowledge?
2. In what ways can people and software leverage this knowledge to improve the reliability and performance of data management systems?

Understanding the behavior of complex data management systems, including user behavior, underlying hardware behavior, and associated compute and networking activities, is key to maximizing the reliability and performance of these systems. Through improved understanding we can eliminate application bottlenecks and unlock the potential of AI to enable the next generation of self-tuning data management services.

## 3.3. Priority Research Direction 3: Rich metadata and provenance collection, management, search, and access

### Background

Occasionally the terms *metadata* and *provenance* are conflated in the research community (e.g., because provenance information is certainly metadata), and so we find it useful to distinguish between them in our discussion.

Metadata, or data about data, can describe logistics data (e.g., file names, dates, format), access permissions, scientific content (e.g., variables, grid information, atomic coordinates), data useful for discovery (keywords, attribution, location, etc.), data describing the production of datasets from simulations and experiments (environment metadata), and much more. Metadata can take various forms (e.g., singular objects or key value pairs) and can be organized in community-accepted schemas and or in files themselves with self-describing formats such as HDF5 and netCDF. Although not a recommended practice, metadata is sometimes embedded directly into filenames. Metadata schemas can themselves be formally described by ontologies: collections of definitions, logical rules about their application and organization, and formal constraints on those rules.

Provenance in computer science is defined as the record of data lineage and software processes operating on the data that enable interpretation, validation, and reproduction of results [Miles, 2007][Freire, 2008].

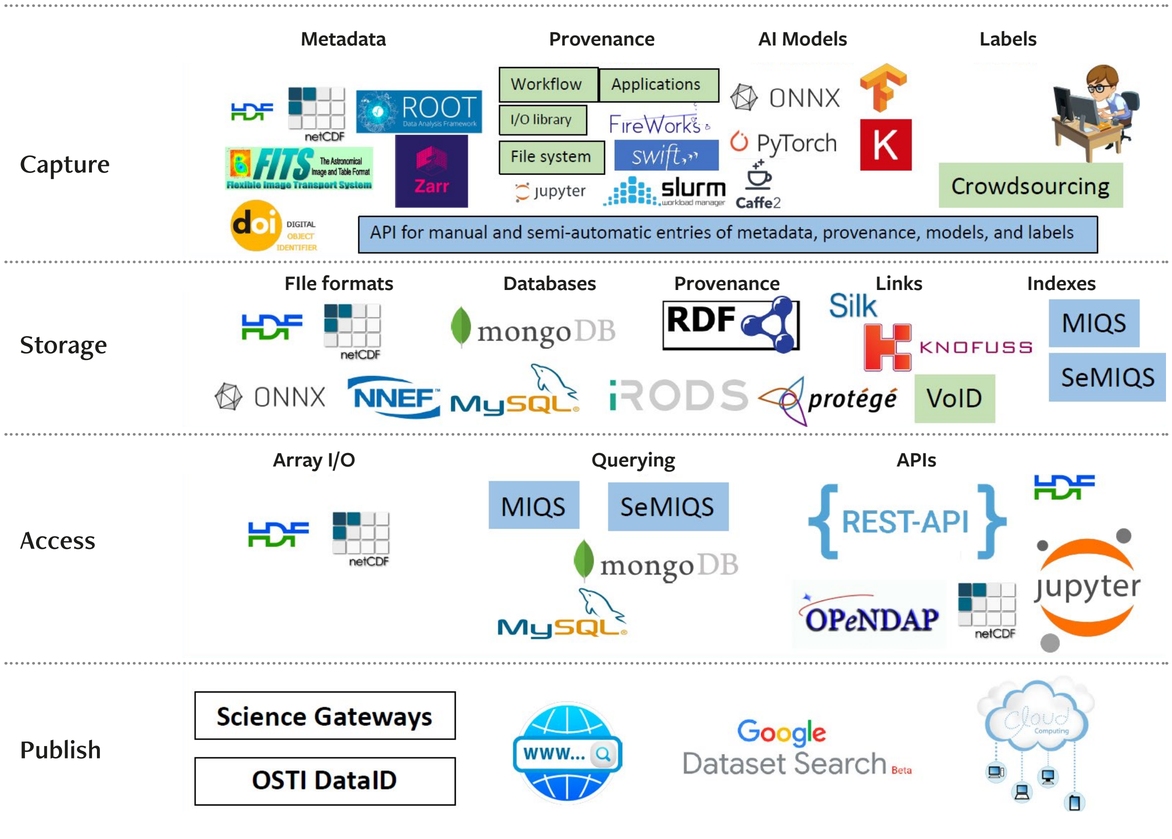


Figure 5. Software and tools related to capturing, storing, and accessing of metadata and provenance to support compliance with FAIR principles.

[Pouchard, 2020]. Provenance is the organized, systematic record that includes metadata associated with datasets in an experiment, their relationship to data and to each other, and the use of metadata schemas and ontologies. Provenance of data and software is crucial to the study of computational workflows that often orchestrate scientific and computational experiments and is an integral part of workflow management systems.

The workshop participants discussed "metadata management support to support the FAIR principles" and "capturing and using provenance" topics in two breakout sessions each.

## State of the Art

**FAIR principles**: The FAIR guiding principles [Wilkinson, 2016][Wilkinson, 2019] heavily rely on metadata and provenance and emphasize machine actionability in the itemized list of concepts and practical recommendations for data management and stewardship. Many communities—in particular the research data management communities within campus libraries and data centers—have embraced FAIR by developing numerous tools, repositories, and protocols to help make data FAIR. In spite of this abundance of research, few data-intensive tools exist that address issues specific to high-performance computing. In HPC contexts, enormous volumes of data and metadata are collected by new high-resolution instruments and sensors in DOE facilities and at computing edges, while streaming and in situ applications produce extremely heterogeneous data at unprecedented rates. Nevertheless, the FAIR principles remain useful to guide the development of data management, storage, and stewardship tools for applications to ensure the automatic capture, storage, organization, discoverability, and reuse of data and metadata produced in DOE-funded research. The application of FAIR principles to the needs and requirements of DOE scientific research and the development of new tools to ensure support of FAIR at scale is crucial to future scientific discovery. FAIR for Research Software (FAIR4RS) emphasizes quality metadata for software findability, interoperability, and reuse [Katz, 2021]. In addition, the application of FAIR to scientific AI has begun, but it is still in its infancy and requires new research to explore the applicability of FAIR concepts and the development of FAIR-minded frameworks, practices, and benchmarks customized to AI. Figure 5 shows various software, tools, and libraries used to support FAIR principles.

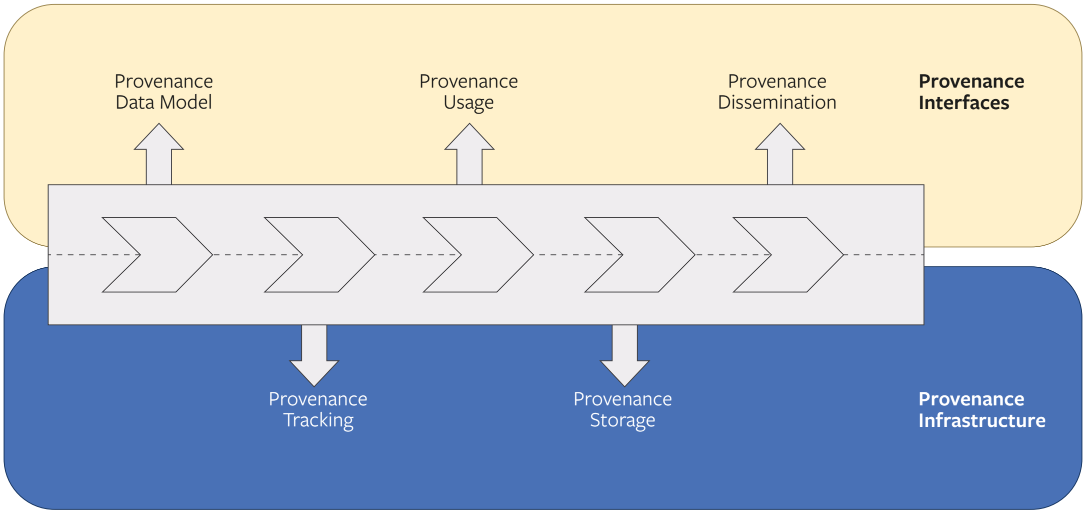


Figure 6: A taxonomy of provenance data collection and usage life-cycle. taxonomy

**Rich metadata and metadata ontologies.** Traditional file system metadata is insufficient to support FAIR principles in scientific applications. HPC storage libraries, such as HDF5, ADIOS, and NetCDF, allow users to describe dimensions of data and attributes to provide scientific and logistic context for data objects. Different scientific domains further propose their data models or formats with rich metadata integrated, such as RO-Crate for Research Object Crate [Carragáin, 2019]; NeXus for neutron, x-ray, and muon science [Konnecke, 2015]; ROOT and Fair4HEP for high energy physics [ROOT][FAIR4HEP]; MASplus for magnetic fusion energy programs [MASplus]; Casacore for radio astronomy data processing [Casacore]; and EFFIS for high-fidelity coupled simulations [Suchyta, 2022]. Because of the large number of rich metadata dialects, Interoperability becomes a key challenge for operating data from different platforms. Systems such as DataFed [Stansberry, 2019] and ScienceCapsule [Ghoshal, 2021] were proposed to provide a federated data management system across domains.

**Metadata data model.** Various data models have been used to manage the rich metadata needed for complex data management tasks. Relational databases are widely used for both traditional parallel file systems (e.g., PVFS [Carns, 2000]) and domain-specific scientific data management tools (e.g., Metacat [Jones, 2001]). NoSQL data models used in IndexFS [Ren, 2014] and HopsFS [Niazi, 2017] have been introduced to address the scalability issues. Storage systems built on graph-based models, such as QMDS [Ames, 2013], GraphMeta [Dai, 2016], and LiFS [Ames, 2005], have been investigated to model metadata in a more flexible manner. Customized models, such as DataStates [Nicolae, 2020], further allow users to tag datasets to facilitate data management tasks. As the Interoperability among data models or formats becomes an issue, standard vocabulary and formats such as Apache Parquet [Parquet], Apache Arrow [Arrow], and Damsel [Koziol, 2014] have been introduced to bridge multiple metadata schemas.

**Metadata scalability.** Scalability of metadata management is a long-lasting research topic. Building distributed metadata services compatible with POSIX semantics is a major research theme; examples are Ceph CRUSH [Weil, 2006], IndexFS [Ren, 2014], DeltaFS [Zheng, 2015], and GIGA+ [Patil, 2011].

Capturing and using provenance are critical steps of implementing FAIR principles in scientific computing. A large body of provenance-related studies and systems exists. As shown in Figure 6, we categorize and summarize these systems based on the life cycle of provenance data, in other words how provenance metadata is modeled, tracked, used, stored, and disseminated.

**Provenance data model**. The W3C PROV [Missier, 2013a] is currently the most commonly used provenance standard (developed based on the Open Provenance Model [Moreau, 2008]). It contains a set of recommendations, such as data abstractions, schemas, ontology, and representations. More specific provenance data models designed based on these standards are also widely seen, such as PAV [Ciccarese, 2013], PROV-AI [Azevedo, 2020], D-PROV [Missier, 2013b], and PROV-IO [Han, 2022].

**Provenance tracking/collection**. Provenance is built based on runtime information such as how data was accessed by applications or users and how applications executed. Such metadata need to be tracked and collected at runtime. Many scientific workflow platforms [Amstutz, 2022], such as Kepler/Komadu [Suriarachchi, 2015], Pegasus [Deelman, 2015], Makeflow [Albrecht, 2012], Galaxy, ProvLake [Azevedo, 2020], and ScienceCapsule [Ghoshal, 2021], support automatic provenance extraction and management. Although workflow-based provenance management is effective, it is often limited to a single environment and lacks the ability to integrate provenance across multiple systems. The general-purpose provenance systems, such as CDE [Guo, 2011], PASS [Reddy, 2006], and ProTracer [Ma, 2016], probe standardized system calls to transparently track program executions and build provenance based on them. One can collect provenance across systems in this way; for instance, PROV-IO [Han, 2022] probes both POSIX and high-level HDF5 I/O calls to collect provenance across layers. However, how to connect the collected low-level system events with high-level domain applications is still a challenge. Providing programmable APIs to users to manually add important provenance metadata sometimes becomes necessary. Containers, such as Singularity/Apptainer [Apptainer] or Docker [Turnbull, 2014], provide a unique opportunity for scientific reproducibility. Frameworks such as Binder [binder], WholeTale [wholetale], and Sci-unit [sciunit] track provenance data at the container level to help scientists connect low-level events and high-level container-based workflows to reproduce experimental results.

**Provenance usage.** Provenance has been investigated and used for a variety of purposes across different communities for many years. For example, Buneman et al. derive provenance in relational databases for understanding the dependencies between materialized views and table updates [Buneman, 2001]; Muniswamy-Reddy et al. intercept system calls via customized kernel modules to capture data dependencies in the OS kernel [Reddy, 2006]; Alvaro et al. use provenance to guide fault injection to improve fault-tolerance protocols [Alvaro, 2015]; and Simmhan et al. propose a publish-subscribe architecture for computing the provenance of sensor data [Simmhan, 2006]. Provenance is also widely used in intrusion detection in security domains [Pasquier, 2018][Bates, 2015][Ma, 2016]. More recently, Azevedo et al. used IBM ProvLake [Azevedo, 2020], and Wozniak et al. developed Braid-DB [Wozniak, 2021] to capture the data lineage across programs in AI workflows, using provenance to detect the performance anomaly in workflow executions and optimize their performance [Kelly, 2020], [Thavasimani, 2019]; Han et al. capture I/O related provenance for understanding the lineage of data products and configuration dependencies [Han, 2022]. Such diverse usages reflect the great potential of leveraging metadata and provenance for managing FAIR-compliant scientific data at scale.

**Provenance storage.** The collected provenance data needs to be properly stored for future usage. Typically provenance is stored in SQL databases [Gehani, 2012], RDF storage [Han, 2022][Dividino, 2009], or graph databases [Dai, 2017][Gehani, 2012][Dai, 2014][Dai, 2018]. The key factor is the scalability of the underneath storage layers, especially for large-scale HPC environments. Another important factor for storing provenance is security. To ensure provenance can be trusted, researchers propose to store provenance using blockchain technology to avoid tampering with the provenance [Neisse, 2017]; others leverage new trustworthy computing infrastructure (e.g., Intel's SGX and AMD's SEV) to redesign the provenance storage systems in HPC [Prowell, 2021].

**Provenance dissemination.** The provenance data needs to be effectively disseminated to users for both exploitation and exploration. Visualization is one way to enable both purposes and has been integrated into many provenance systems, such as VisTrails [Callahan, 2006], Orbiter [Macko, 2011], SPADE GraphViz [Gehani, 2012], Probe-it [Rio, 2007], and ZOOM UserViews [Biton, 2007]. In addition to visualization, programmable query interfaces based on SQL, graph query languages, or RDF query languages are widely used. Provenance retrieval APIs, such as Disclosed Provenance API [Reddy, 2009], Core Provenance Library (CPL) [Macko, 2012], and IPAPI [Carata, 2013], are seen in existing provenance frameworks as complementary to domain scientists.

## Workshop Findings

Storage and I/O technologies have traditionally focused on efficient data storage and access. Given that history, metadata usage long was limited to descriptions of data components, such as the name of a data object or a file or access restrictions. Self-describing file formats allowed storing and providing more descriptions about data objects. For instance, HDF5 and NetCDF allow describing dimensions of data and attributes to provide scientific and logistic context for data objects. More recently, lookaside solutions powered by relational and "NoSQL" database technologies have made it possible to create and associate arbitrary annotations; they also decouple metadata from data, enabling more straightforward discoverability and reuse.
Data provenance is one of several crucial resources to ensure the trustworthiness of data. Provenance has several benefits, including strategies to optimize data movement, avoid reinvention of wheels in scientific exploration, and identify sources and users of data. Despite these benefits for scientific data, collection and utilization of provenance have been sparse or limited to specific scientific repositories. Undocumented changes to data are common and can lead to false conclusions

in science [Hills, 2015]. As HPC resources are increasingly used to act on experimental, observational, and sensor data, provenance gathering and use throughout the data life cycle become a requirement.

The workshop participants discussed many aspects of metadata and provenance, including the variability and ambiguity in standards, the diversity of use cases and the need for clear definitions, the vision of building scalable infrastructure and runtime for rich metadata and provenance, and the roles of individuals and communities in addressing the challenges. The key findings are organized as follows.

**Standardization and specifications.** A few well-known concepts exist for guiding the description, collection and usage of metadata and provenance (e.g., FAIR principles [Wilkinson, 2016] and W3C PROV models and ontologies [Moreau, 2013] [Lebo, 2013]). Unfortunately, the interpretation and adoption of these concepts vary across fields and communities, limiting the potential benefits to the scientific communities in general. This diversity is largely due to the inherent ambiguity in the high-level definitions as well as the complexity of HPC ecosystems. For example, it is straightforward to assign a unique ID to static data (e.g., DOI numbers or an URL) to support the findability principle in FAIR, but it is unclear how to associate a persistent ID around dynamic data or in situ data streams. Similarly, reusability may encompass a broad spectrum of topics including repeatability, reproducibility, and repurposing, all of which are context-sensitive and may imply different levels of metadata or provenance information (e.g., data formats, workflow parameters, library dependencies). Clearly needed is a common vocabulary that concretizes the high-level concepts and enables effective communication and sharing across communities.

**Usage of metadata and provenance.** The importance of metadata and provenance has been well recognized across communities, and diverse use cases have been demonstrated to varying degrees (e.g., lineage-based fault injection in databases, capturing of hyperparameters in ML or AI workflows [Souza, 2019b] described earlier). Nevertheless, the individual use cases and the associated solutions tend to be application-specific and thus cannot be easily translated to enable FAIR-compliant usage of scientific data in general. Because of the variety of data and metadata that could be generated from HPC ecosystems, it is difficult for most domain scientists today to specify precisely what specific metadata or provenance information is needed or how it may help. Such ambiguity limits the adoption and usage of existing metadata and provenance solutions, which in turn makes clarifying the ambiguity and addressing real scientific needs difficult. Consequently, the explosion in data and metadata generation is outpacing what one human or team can hope to process or interpret, despite the large set of tools described in the preceding subsection. Therefore, methodologies are desired that can clarify the ambiguity, bridge the semantic gaps across fields and communities, and enable precise definition and measurement of heterogeneous use cases.

**Scalable infrastructure and runtime support.** The explosion of data heterogeneity and data sizes, coupled with an increasing speed of accumulation, has resulted in a corresponding explosion of the metadata and provenance information necessary to keep up with the data. Consequently, a scalable infrastructure and runtime support is needed that can efficiently handle not only the storage of metadata and provenance information, but also a variety of complex queries that can extract meaningful insight from that information. In this context, the workshop attendees identified a need for better organization and distributed indexing, which natively favors concurrent access, both for reading and writing. This aspect is linked with managing the volume of metadata and provenance information (pruning, frequency of capture and granularity, aggregation and summaries), consistent exposure (persistent identifiers), and trust (curation). Starting from this foundation, a simple yet flexible query support that leverages declarative, extensible, and self-explaining aspects of metadata/provenance and the relationships between them is critical at application-level. Furthermore, given that users are overwhelmed with the data itself, metadata and provenance need to avoid becoming a burden. Hence, its automated capture, monitoring of performance overhead vs. usefulness, and autotuning are desired.

**Human factors and cross-community collaboration.** Clear roles and collaboration are critical to support the creation and distribution of FAIR-compliant data. For example, the metadata and provenance needed for achieving Interoperability heavily depend on specific system characteristics and use cases across facilities and scientific communities. Collective efforts are thus essential to establish a common vocabulary and enable FAIRness at scale. Also, similar to data security (which relies on metadata and provenance), metadata and provenance should be exposed only to the right people. In other words, security policies (e.g., authentication, authorization, access control) must be specified and enforced for metadata and provenance, processes involving both technical (e.g., auditing mechanisms) and nontechnical (e.g., ethics) perspectives. These topics are increasingly important because more and more scientific data today is generated or consumed beyond a single HPC center (e.g., across institutions and cloud or edge sites) and shared globally.

### Summary

The explosion in data generation has been accompanied by one in metadata generation. Mature parallel file storage systems (and much research I/O software) are optimized for bulk data output, but I/O patterns of accessing metadata are typically random and small and often start with a query. Data provenance, namely, the lineage of data in its life cycle, plays a critical role in providing integrity of data and reproducibility of scientific results. In the age of artificial intelligence helping numerous fields of science in extracting patterns in large amounts of data, trustworthy data is essential. The emergence of the FAIR principles has highlighted a growing need for managing metadata and provenance in a principled manner. Rich metadata and provenance adhering to agreed-upon semantic standards can speed up discoverability of data and hence the process of scientific discovery.

New and enhanced methods are needed for capturing, storing, searching, and accessing machine readable and actionable metadata. R&D efforts are needed to develop standards, tools, and technologies to support more capable metadata management: improving findability of data, searching massive amounts of heterogeneous metadata, increasing the value of data using metadata, maintaining relationships among data objects and datasets from different data sources, and maintaining metadata even when the data is no longer available (or the creators of that data no longer directly manage it).

Research and software development are needed to drive advanced provenance capabilities in computational science, such as documentation of the lineage of data lifecycle and workflows, annotation of relationships across datasets within a repository and across multiple repositories across institutional boundaries, storage of vast amounts of provenance metadata using efficient data structures, searches for the stored provenance metadata, use of provenance information for various optimizations, and automatic generation of ontologies using AI technologies.

### Summary of Priority Research Directions 3

Rich metadata and provenance collection, management, search, and access

Key Questions:

1. What metadata and provenance are needed to support FAIR principles?
2. How do we support collection, storage, and search of rich metadata and provenance?
3. How can we use rich metadata and provenance for optimizing data management?

Metadata and provenance are critical for supporting the FAIR principles for reproducible science. R&D efforts are needed to enable management of the voluminous metadata inherent in modern science, to identify metadata and provenance that are effective for supporting FAIR principles, and to understand how to best collect and use metadata and provenance for improving data management systems and scientific discovery as a whole.

## 3.4. Priority Research Direction 4: Reinventing data services for new applications, devices, and architectures

### Background

Data management architectures and services encompass the hardware and software that together provide data management to scientific workflows: storage and networking devices, file systems, databases, object stores, and others. Successful architectural and service designs enable productive interactions with data while simultaneously making best use of the capabilities of the hardware resources. To this end, data management architectures and services must account for both the variety of applications of HPC systems and the intricacies of cutting-edge HPC hardware. HPC data management architectures and services have not rapidly adapted to new workloads including AI and experimental data analysis, nor is it clear that new technologies such as SmartNICs and computational storage can be readily incorporated into their designs.

The workshop participants discussed this topic in two breakout sessions, with additional conversations occurring in related breakout sessions.

### State of the Art

**Cloud and HPC.** HPC clouds are becoming an alternative to on-premise clusters for executing scientific applications [GoogleCloud][Azure][AWS][IBMCloud]. Outside HPC, ongoing work is aiming to better understand how to deploy applications

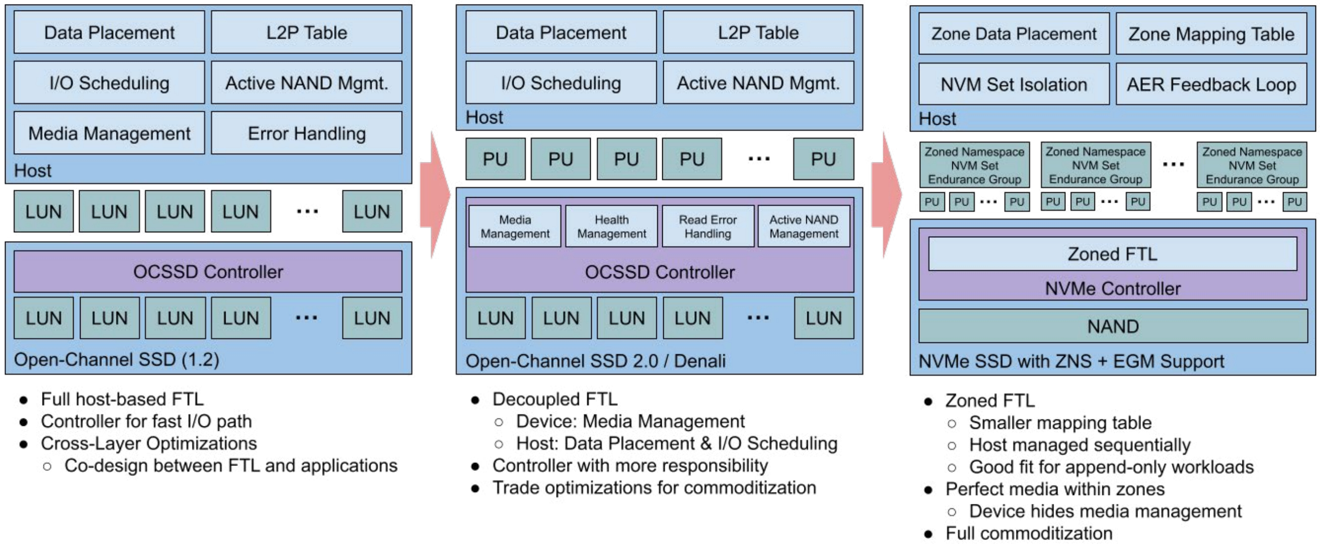


Figure 7: Evolution of SSD architectures that do away with the on-device firmware Flash Translation Layer, from Open-Channel SSDs to NVMe Zoned Namespaces, or ZNS. Early Open-Channel architectures relegated all flash translation layer (FTL) functionality to the host, making software development and upkeep difficult. The newest ZNS architecture strikes a balance between an application-friendly interface and granular data management that allows for lower capacity costs and predictable performance. Source: Flash Memory Summit, 2019.

to the public cloud in a cost-effective manner [Netto, 2019] [Mahgoub, 2020], but a better understanding of both HPC workflows that are eligible for public cloud deployment and how existing research applies to them is lacking.

**Storage Interfaces.** HPC storage interfaces such as HDF5 [Folk, 2011], ADIOS [Liu, 2014], PnetCDF [Li, 2003], Zarr [Zarr, 2021], and MPI-IO [Thakur, 1999] hide differences in backend storage and provide data structures that HPC tooling is compatible with but does not naturally translate to block and object cloud storage.

**HPC storage stacks.** Bespoke solutions, while efficient, make it difficult to provide general solutions for cloud, in-memory processing, and disaggregated or other architectures. Projects such as the Mochi storage microservice ecosystem [Ross, 2020] aim to design building blocks for storage stacks, an approach that can reduce development time and encourage innovation in the form of reconfigurable storage stacks.

**Emerging storage technologies.** Storage devices have gained richer interfaces and capabilities, including zoned namespaces [Bjørling, 2021] and embedded functions in the form of key-value stores [Pitchumani, 2020] or more general computational storage [SNIA, 2021]; but understanding the value that these devices can add for scientific applications is an open problem, and adding support for these devices in HPC storage stacks could be a significant undertaking that requires community effort.

**Indexing technologies.** Handling massive amounts of data efficiently is shaping up as one of the cornerstones of exascale computing. At these scales, data scans are to be avoided at all costs, and recent works have focused on the development of data indexes or other auxiliary data structures that can be used to efficiently navigate massive datasets and guarantee low query latencies. FastQuery [Chou, 2011] creates bitmap indices in a parallel postprocessing stage, and DeltaFS [Zheng, 2018] partitions and indexes data in situ, as it is written to storage.

**New hardware interconnects.** CXL [CXL] looks to become the state of the art for commodity system interconnects for high-performance environments including data centers. It subsumes Gen-Z [Gen-Z]—the main competing standard—and provides for resource pooling and memory coherence at high data rates. Beyond the scope of a single system, Ethernet—the most prevalent open network interconnect—is being standardized for 1.6 Tbps links [Ethernet].

**In-network/memory/storage computing.** In-network computing [NetCompute, 2018] is an emerging computing paradigm that leverages programmable network hardware – switches as well as NICs. The state of the art for current switch products (e.g., the Tofino line from Intel or Trident from Broadcom) involves packet processing at 10 Tbps and above, and the next generation of designs is intended to double that. Techniques are being developed for network-wide programming of such a fabric [Sultana, 2021].

Commercial products also exist for processing-in-memory [Ghose, 2019], building on the ideas of computational RAM [CRAM], and are being adapted for in-storage computing [Ruan, 2019], for which commercial products also exist. As with in-network computing, techniques are being developed to aid with optimizing the placement of computing tasks in relation to the data being processed. Recent commercial storage systems also are beginning to provide interfaces and support for near- and in-storage processing [Computational Storage, 2022] (referred to as active storage or computational storage) —a desired capability that initially was proposed and demonstrated in the context of in situ data analytics for HPC workflows almost a decade ago [Tiwari, 2013][Kang, 2013].

**Ecosystem for big data processing.** During several sessions, participants referenced the ecosystem of big data processing tools—several of which were developed at cloud companies—that are being used, or could be useful, in HPC environments. This ecosystem includes Apache products for analyzing structured data [Apache Arrow], for MapReduce-style computing jobs [Apache Hadoop], for storing huge tables [Apache Iceberg], and for distributed stream processing [Apache Kafka]. Severless computing [Castro, 2019] is the cloud-based nomenclature for what in HPC would be regarded as comprehensive middleware for storage and compute. The momentum of serverless computing is evidenced by its commercial uptake and the variety of open frameworks that implement the paradigm—the primary two being Apache OpenWhisk, which enables users to perform functions in response to events [Apache OpenWhisk], and OpenFaaS, which enables developers to deploy event-driven functions and microservices to Kubernetes without repetitive coding [OpenFaaS]. Researchers have started exploring the targeting of HPC jobs to public cloud services [Roy, 2022].

**Metadata and reproducibility of research.** Attendees in several of the breakouts expressed an interest in FAIR principles, workflows, and tools [Ghoshal, 2021]. One project mentioned was FAIR4HEP, a state-of-the-art initiative related to FAIR, HEP, and AI models [FAIR4HEP]. This is a point of connection with rich metadata and provenance (S.3.3), and data services will likely need adaptation to capture the metadata and provenance necessary to enable FAIR.

**I/O performance modeling, prediction, and control.** The scale and complexity of storage and processing systems used in HPC involve I/O access patterns that are still being understood [Patel, 2019][Patel, 2020]. To avoid the suboptimal use of I/O leading to congestion and bottlenecks, researchers have proposed various schemes, including: abstraction interfaces [Costa, 2021][Ghoshal, 2017], inference of data movement [Shin, 2019], autotuning by relying on genetic algorithms [Behzad, 2013], explainable AI models [Isakov, 2020], and paradigm-specific workload characterization for serverless [Roy, 2021].

**Co-design of hardware and services**. By offering unprecedented programmability, emerging computational devices force us to rethink the storage architecture and services holistically. Co-designing the hardware and software stack becomes essential to harness the power. For example, Microsoft co-designs host networking with SmartNICs and achieves scalable network in the Azure cloud [Firestone, 2018]; the Alibaba pushdowns table scans across database engines, file systems, and device drivers to computational storage and enables scalable cloud-native OLTP services [Cao, 2020]; and most recently, the processing of large-scale graph neural networks has been accelerated via computational SSDs, outperforming GPU-based solutions multiple times [Miryeong, 2022]. These advances have demonstrated promising co-design opportunities at scale. However, understanding the potentials and achieving the anticipated benefits in the HPC context remain an open challenge due to the different system stacks and scientific needs.

## Workshop Findings

**Metadata instastructure with data life-cycle and query support.** With increasing complexity and variety of data, dealing with the explosion of data sizes alone is not enough. Needed in addition is a scalable metadata infrastructure that exposes an easy way to index, search, and query large data. Starting from questions such as what metadata to capture (namespaces, labels, content and structural properties, intent and constraints), the attendees noted a need for data life-cycle and query support that addresses aspects such as storage abstractions that facilitate easy and/or automated extraction of metadata, efficient indexing that avoids scanning the whole data/metadata, rich semantic/ontological queries that complement simple filtering/aggregation queries, tracking of the dataflow and provenance, and optimizing of data location, layout, and representation based on metadata.

**Heterogeneous storage and emerging devices.** HPC systems continuously add new layers of heterogeneous memories and storage devices that can be exploited both locally on compute nodes and remotely through dedicated I/O servers. In this space, not enough effort has been made to unify data management across these layers and simplify access for users. Specifically, the attendees noted the need for capabilities such as auto-tiering, data placement and caching, dynamic provisioning, control interfaces and synergies with the metadata infrastructure. Furthermore, emerging devices such as persistent memory blur the line between memory and storage, thus introducing the need for unified access models (e.g., reconcile memory-oriented APIs vs storage-oriented APIs) and hardware customization.

**Near-data processing capabilities.** With increasingly complex data preprocessing and metadata queries, the latency of transferring the data and metadata close to the compute elements becomes prohibitively expensive. In this context, the attendees noted the need to exploit compute-in-storage and compute-in-network/edge to move compute tasks closer to the data, both for data preprocessing and for metadata query processing. Especially important are aspects such as multitenancy and fairness and how to encapsulate compute tasks, deploy compute tasks uniformly, and handle different degrees of compute capabilities close to the data.

**Data management and streaming for workflows.** Data is increasingly in motion, serving the producer-consumer patterns of workflows with complex task dependencies. In this case, traditional storage repositories (parallel file systems, object stores) are often the common denominator across systems. While convenient to use, they do not meet the requirements of workflows (performance, scalability, resilience). The attendees noted the need for specialized profiling/tuning, specialized data services that can handle streaming data efficiently and can adapt to both local and global task patterns, synergies between workflow-specific and generic data services, and coupled vs. separate metadata management.

**Convergence with high-level data management (databases and beyond).** Despite rapidly evolving data management requirements, most HPC storage efforts are still centered on POSIX and other low-level I/O-oriented data access models (object stores, key-value pairs). Traditional databases (e.g., SQL) are heavyweight and not intended for deployment at HPC scales. Cloud technologies including noSQL databases can scale to meet HPC needs, but they were developed for a different environment (e.g., TCP/IP) and workloads (e.g., Internet commerce). Convergent data services are needed that combine needed database capabilities for DOE science applications, that embody the scalability of cloud solutions, and that can leverage the emerging technologies present in exascale platforms and beyond. In particular, the attendees noted a need to rethink the current storage abstractions such that they naturally couple a fine-grained data distribution across heterogeneous low-level building blocks with a scalable metadata infrastructure that allows efficient indexing and query support.

**Disaggregated storage.** With increasing heterogeneity of convergent HPC, big data, and AI workloads, it is becoming increasingly harder to design a balanced storage stack that serves all types of I/O access patterns efficiently while remaining affordable. In this regard, disaggregated storage is a promising solution to improve resource utilization and reduce costs. At the same time, this introduces an entirely new set of challenges: need for stricter data protection and security (in particular, fine-grained access control to individual data structures and objects), trustless design without compromising I/O performance, efficient remote access that complements near-data processing capabilities, fairness and interference mitigation under multitenancy, and adaptability to I/O access patterns.

**Monitoring, performance analysis, and adaptability.** The complexity of the storage stack (an entire hierarchy of node-local memories and storage devices, remote repositories) makes it infeasible for applications to keep relying on trial-and-error I/O performance tuning. This problem is amplified by changing I/O patterns during runtime. The attendees noted a need for better monitoring and performance analysis tools that provide insight into both the individual behavior of storage tiers and the complex interactions between them. Starting from these tools, a new generation of flexible storage services is needed that is composable (providing the basic building blocks to construct a customized stack) and capable of reconfiguring itself on the fly to efficiently address changing I/O patterns and data requirements.

**Unified cloud/HPC storage stack.** Cloud computing has the potential to allow for rapid development and deployment of HPC applications by making virtual clusters available on-demand. To leverage this platform, however, we need to close the semantic gap between cloud and HPC storage architectures. While cloud resources are virtually abundant and allow for composing reconfigurable storage systems, cloud APIs are not designed for HPC programmers. Therefore, the attendees noted a need to develop backend interfaces that allow for portability between cloud block and object storage and structured HPC data formats; a need for understanding the cost-benefit of moving resource-intensive applications from on-premise environments to public cloud platforms; and a need to identify whether new data abstractions are required for improved performance, compatibility, and cost.

### Summary of Priority Research Direction 4

Reinventing data services for new applications, devices, and architectures

Key Question:

1. Using a co-design approach, how do we create specialized data services leveraging emerging device technologies to enable revolutionary breakthroughs across the breadth of DOE science?

New science endeavors and approaches require specialization of how data is accessed, organized, and retained. New networking and storage devices, including ones with computational capabilities, merit revisiting data service design to maximally exploit these technologies. New architectures, including scenarios in which data lives across sites or across administrative domains or is generated at the edge, similarly place new requirements on data services. Co-design of these services with scientists, hardware architects, and facility operators is needed to unlock the potential of data in these unique environments and ease porting to new ones

## 3.5. Crosscutting themes

### AI for data management and data management for AI.

The use of AI and ML technologies to extract insights from massive scientific datasets has been steadily increasing over the past years. This integration between data management and AI is bi-directional (e.g., in workflows [da Silva, 2021]). The first direction is to apply AI and ML techniques for the optimization of existing data processing pipelines and workflows. Examples include the design of data partitioning and placement strategies, indexes, and statistical cardinality estimators using ML techniques such as regression and autoregressive models, embeddings, and deep neural networks. Although these models can be trained exclusively from the data, they require the acquisition/generation of extensive query workloads when the model features include query clauses. For example, a cardinality estimation model that uses selection predicates as features requires selection queries for training. While this is relatively simple to do for certain querying tasks, an entire logging framework has to be developed in the case of complex workflows. Reinforcement learning methods are another class of AI techniques applied to data storage decisions and for computing the join/correlation among different datasets. The generation of the training dataset is even more complicated in this case because learning the reward function requires both a large number of examples and diverse examples.

The second integration direction consists in the design of efficient data processing techniques to support the training and prediction of massive ML models having billions of parameters and hyperparameters. Examples of such large models include convolutional deep neural nets for image classification and embedding models for text synthesis. Although ML processing extensively uses linear algebra operations, which are a staple of scientific computing, their integration in complex workflows poses new challenges. These can be addressed by extending the database query processing techniques specific to out-of-memory and distributed settings. These techniques are integrated into declarative languages that handle query optimization transparently through relational-linear algebras.

Overall, R&D efforts are needed to support both integration directions, AI for data management and data management for AI. These efforts are crosscutting across the four identified priority research directions. The I/O APIs that enable complex AI workflows on modern HPC and edge systems with complex memory hierarchies and heterogeneous accelerators have to be redesigned. More effective schedulers and data movement tools are needed for using computation, memory, and storage resources efficiently to perform training and inference. Novel AI analysis methods are useful for understanding the behavior of data movement and for optimizing data management services and architectures. Research is also needed to use AI methods to analyze massive metadata and provenance that can lead to identifying or recommending relevant datasets and information to scientists. Achieving these goals also requires the development of representative benchmarks for complex AI workflows [da Silva, 2021].

### Co-design.

Co-design is defined as the process of jointly designing interoperating components of a computer system—in particular, applications, algorithms, programming models, and system software, as well as the hardware on which they run and the facilities they run in [DOE, 2022]. Data has a preeminent role within any HPC contribution to discovery in the modern scientific environment: if data analysis is not considered as critical to science as calculation, HPC's contributions to scientific discovery will be severely curtailed. It follows that the management and storage of scientific data touch on an especially broad segment of HPC practitioners. By mutually addressing issues in a cooperative manner, co-design avoids many pitfalls associated with insufficient context. HPC is rapidly evolving, and no single community (e.g., domain science end users, system software developers, facility staff, vendor R&D staff) can offer general advancements without involving the other affected communities. Co-design is needed to span the breadth of data issues and concerns associated with scientific data management.

For instance, the workshop attendees noted that co-design is necessary in the development of high-level data model capabilities. Without sufficient representation from varied communities within HPC, it is difficult for a single HPC group to accurately specify what operations the underlying system should perform. Co-design is needed to explore how high-level patterns can utilize these capabilities on emerging hardware architectures and storage technologies. On the software side, library maintainers must adapt to efficiently use more complex storage hierarchies without unnecessary overheads and programming complexities. On the hardware side, optimizations could be made to support the numerically oriented access patterns common in scientific computing. Both sides must contend with the differences in consistency requirements between scientific and enterprise computing.

A second example may be found in HPC's unprecedented programmability. Emerging computational devices force us to rethink the storage architecture and services holistically. Co-designing the hardware and software stack becomes essential to harness the power. The rapidly evolving HPC landscape has demonstrated promising co-design opportunities at scale. However, understanding the potentials and achieving the anticipated benefits in the HPC context remain an open challenge due to the different system stacks and scientific needs. Unifying these stacks is an important part of the goal of HPC-cloud convergence, which remains on the 10-year horizon.

New science endeavors and approaches require specialization of how distributed data are accessed, organized, and retained. New networking and storage devices, including ones with computational capabilities, merit revisiting data service design to maximally exploit these technologies. New architectures, including scenarios in which data lives across sites and across administrative domains or is generated at the edge, similarly place new requirements on data services. Co-design of these services with scientists, hardware architects, and facility operators is needed to unlock the potential of data in these unique environments and ease porting to new ones. Research is needed to determine how best to disaggregate, view, modify, and manage large, collaborative datasets.

### FAIR.

For scientific data to be reusable efficiently, the FAIR principles provide guidelines for collecting various types of metadata and provenance. These principles crosscut in all four thrusts discussed at the workshop that require interfaces, standardization efforts, and metadata services.

The "High-productivity interfaces" section suggests that advanced metadata capabilities (e.g., search, sharing, reproducibility, annotations) will be critical for data-driven studies. Interfaces for collecting metadata, such as annotations, attributes, descriptions of the data, and intent of the users for producing or accessing data, are required. The Interfaces section also suggests that FAIR is particularly critical for AI consumers of data [Fagnan, 2019] and the AI-for-science processes.

Provenance of the data life cycle that may go beyond the current requirements of the FAIR principles is potentially useful for understanding and optimizing data management systems and AI processes. A few recent studies have provided interfaces and standards for collecting and managing provenance data with the goal of improving I/O performance [Li, 2019][Murugan, 2022][Han, 2022]. IBM has been recently working on the ProvLake [Souza, 2019a][Souza, 2019b] effort that proposes a standard for collecting AI parameters that can be used for optimizing AI model development. Further efforts are needed to standardize these interfaces for collection and management of provenance for optimizing data management systems.

Overall, the increased size and complexity of metadata and provenance require efficient management, search, and access of FAIR-compliant data. In addition to plain compliance with the FAIR principles, metrics and methods are required for finding data that a scientist desires among data repositories. Scientists may need interfaces for accessing specific data objects or parts of large data variables that match given conditions. These features require standardization of FAIR metadata interfaces, storage, and benchmarks for evaluation of FAIR compliance.

# 4. Summary/Conclusion

Future scientific activities will encompass an increasingly broad range of domains and span both HPC resources and advanced scientific instruments. Significant new directions in hardware architectures are leading to more complex and heterogeneous environments. Design decisions made long ago for a much different environment and a decidedly different workload are no longer appropriate for the full scope of today's requirements. This report finds that key advances are needed in the following areas:

- High-productivity interfaces for accessing scientific data efficiently
- Understanding of the behavior of complex data management systems in DOE science
- Rich metadata and provenance collection, management, search, and access
- Reinventing of data services for new applications, devices, and architectures

The report findings recommend R&D efforts to support advances in each of these topics. Scientists and facility operators working together to co-design data management architectures will ensure that we have the most capable and robust tools for managing these troves of valuable scientific results. By improving how we describe and structure and access this data, we will enable greater sharing of data than ever before and facilitate automation of science with artificial intelligence.

# 5. Acronym glossary

| | |
|---|---|
| **ADIOS** | Adaptable IO System. ADIOS provides a simple, flexible way for scientists to describe the data in their code that may need to be written, read, or processed outside of the running simulation. |
| **AI** | Artificial Intelligence. Artificial intelligence is intelligence demonstrated by machines, as opposed to the natural intelligence displayed by animals including humans. |
| **API** | Application programming interface. Syntax and semantics for invoking services from within an executing application. |
| **ASCR** | The Advanced Scientific Computing Research (ASCR) Program within the Department of Energy Office of Science is a program with the mission to discover, develop, and deploy computational and networking capability to analyze, model, simulate and predict complex phenomena important to the Department of Energy and the advancement of science. |
| **BB** | See Burst Buffer. |
| **Burst-Buffer** | The Burst Buffer is an intermediate, high-speed layer of storage that is positioned between the application and the parallel file system (PFS), absorbing the bulk data produced by the application at a rate a hundred times higher than the PFS, while seamlessly draining the data to the PFS in the background. |
| **Checkpoint** | A snapshot of the state of a process that is sufficient to allow the process to resume execution from the point the checkpoint was recorded. |
| **Co-design** | Co-design refers to a computer system design process where scientific problem requirements influence architecture design and technology and constraints inform formulation and design of algorithms and software. |
| **Co-locate** | Co-locate refers to the placement of multiple services which exist in different enclaves, on a single node. One reason for co-location is to minimize data movement. |
| **Consistent** | Guarantees that data accesses within a multi-layered memory hierarchy provide a compatible view of the data free of contradictions. |
| **COW** | Copy-on-Write. An optimization for consistency. Copy-on-write is the name given to the policy that whenever a task attempts to make a change to the shared information, it should first create a separate (private) copy of that information to prevent its changes from becoming visible to all the other tasks. If this policy is enforced by the operating system kernel, then the fact of being given a reference to shared information rather than a private copy can be transparent to all tasks, whether they need to modify the information or not. |
| **CSV** | A comma-separated values file is a delimited text file that uses a comma to separate values. Each line of the file is a data record. Each record consists of one or more fields, separated by commas. |
| **DAX** | Direct Access. The facility of retrieving data immediately from any part of a computer file, without having to read the file from the beginning. |

| | |
|---|---|
| **Digital twins** | A digital twin is a virtual representation that serves as the real-time digital counterpart of a physical object or process. |
| **DOE** | The United States Department of Energy. |
| **DOI** | A digital object identifier is a persistent identifier or handle used to uniquely identify various objects. |
| **DRAM** | Dynamic Random Access Memory. A high performance RAM which requires a periodic refresh. Generally, DRAM has favorable latency and bandwidth characteristics but unfavorable power consumption characteristics. DRAM is volatile. |
| **FAIR** | Findable, Accessible, Interoperable, and Reusable. A set of data management principles defined in [Wilkinson, 2016]. For data to be Findable, it should satisfy: (F1) (meta)data are assigned a globally unique and persistent identifier; (F2) data are described with rich metadata (defined by R1 below); (F3). metadata clearly and explicitly include the identifier of the data it describes; (F4). (meta)data are registered or indexed in a searchable resource For data to be Accessible, it should satisfy: (A1). (meta)data are retrievable by their identifier using a standardized communications protocol. The protocol is open, free, and universally implementable; The protocol allows for an authentication and authorization procedure, where necessary; (A2) metadata are accessible, even when the data are no longer available. For data to be Interoperable, it should satisfy: (I1) (meta)data use a formal, accessible, shared, and broadly applicable language for knowledge representation; (I2) (meta)data use vocabularies that follow FAIR principles; (I3) (meta)data include qualified references to other (meta)data. For data to be Reusable, it should satisfy: (R1) meta(data) are richly described with a plurality of accurate and relevant attributes; (meta) data are released with a clear and accessible data usage license; (meta)data are associated with detailed provenance; (meta)data meet domain-relevant community standards. |
| **GPFS** | General Parallel File System. A proprietary PFS developed by IBM. Recently, IBM has rebranded GPFS as an element of their Spectrum Scale suite of software. |
| **GPU** | Graphics Processing Unit. A GPU may be used together with a CPU to accelerate scientific and analytical workloads. |
| **HDD** | Hard Disk Drive. A storage device designed around a rotating media platter coated with magnetic material. The platters are paired with magnetic heads on a moving actuator arm. (see also SSD). |
| **HDF** | Hierarchical Data Format (HDF) is a set of file formats (HDF4, HDF5) designed to store and organize large amounts of data. |
| **HPC** | High Performance Computing. |
| **In situ** | A phrase that translates roughly to "in it's original place" or "in position". An ASCR Funding Opportunity Announcement targeted at Scientific Data Management (Lab_14_1043), defined in situ to include: "the reduction, analysis and visualization, occurring in parallel with the simulation, either on the same nodes or on specially designated nodes. A key aspect of in situ processing is that data are intelligently reduced, analyzed, transformed and indexed while they are still in memory and before being written to disk or transferred over networks." |
| **Inter-process** | Involving two or more processes where each process is an executing program. |

| | |
|---|---|
| **Interface** | Syntax and semantics for invoking services from within an executing application. |
| **IO** | Input/Output. Data movement up and down the Memory Hierarchy Layers (MHL). |
| **iRODS** | Integrated Rule-Oriented Data System (iRODS) is open source data management software with the goals of data virtualization, data discovery, data workflows, and secure collaboration. |
| **Isolation** | In distributed systems, (isolation) is the property that defines how/when the changes made by one operation (or entity) become visible to other concurrent operations (or entities). Isolation is a key consideration in the security of federated systems. |
| **Job** | A job comprises a collection of related, potentially interacting enclaves executing on a partition of a machine. A job may also interact with enclaves that are not considered part of the job, such as service enclaves. |
| **KVS** | Key Value Store. A system designed for storing, retrieving, and managing associative arrays, a data structure more commonly known today as a dictionary or hash. Dictionaries contain a collection of objects, or records, which in turn have many different fields within them, each containing data. These records are stored and retrieved using a key that uniquely identifies the record, and is used to quickly find the data within the database. (Contrast with relational database.) |
| **Map-reduce** | Map-Reduce is a programming model and an associated implementation for processing and generating big data sets with a parallel, distributed algorithm on a cluster. A Map-Reduce program is composed of a map procedure, which performs filtering and sorting, and a reduce method, which performs a summary operation. |
| **Metadata** | Data providing information about one or more aspects of the data. |
| **ML** | Machine Learning. Machine Learning is a field of science devoted to understanding and building methods that 'learn', that is, methods that leverage data to improve performance on some set of tasks. It is seen as a part of artificial intelligence. |
| **MPI-IO** | MPI-IO is a portable interface defined by the Message Passing Interface (MPI) Forum in order to perform parallel I/O operations within distributed memory programs, leveraging MPI key concepts such as communicators, datatypes, and collective operations. |
| **MSSD** | Management and Storage of Scientific Data. |
| **NetCDF** | NetCDF is a set of software libraries and self-describing, machine-independent data formats that support the creation, access, and sharing of array-oriented scientific data. |
| **Node** | From the hardware perspective, a node is the building block in a parallel machine; it usually consists of a processor or multiprocessor, memory, an interface to the interconnect and, optionally, a local disk. In cases where a single node contains multiple processing units (e.g., multiple cores), the node may be divided into multiple virtual nodes to permit co-location. |
| **noSQL** | Not Only SQL. NoSQL is an approach to database management that can accommodate a wide variety of data models that are non-relational (e.g., key-value, document, columnar and graph formats) and generally do not use SQL. |

| | |
|---|---|
| **NVM** | Non-volatile Memory. (See also NVRAM.) |
| **NVMe** | Non-Volatile Memory Express. A standard hardware interface for solid state drives (SSDs) that uses the PCI Express (PCIe) bus. |
| **NVRAM** | Non-volatile Random Access Memory. A type of non-volatile memory that allows for data to be accessed quickly in any random order. |
| **Open source** | Software that is available to users in source form and can be used and modified freely. |
| **OS** | Operating system. |
| **OSR** | Operating system and Runtime system. |
| **P2P** | Peer-to-Peer communications. Peers are equally privileged, equipotent participants in the application. Peers are both suppliers and consumers of resources, in contrast to the traditional client-server model in which the consumption and supply of resources is divided. |
| **PCIe** | Peripheral Component Interconnect Express. PCIe is an interface standard for connecting high-speed components. |
| **Persistence** | Any method or apparatus for efficiently storing data structures such that they can continue to be caccessed using memory instructions or memory APIs even after the end of the process that created or last modified them. (Note: persistence does not imply consistency.) |
| **PFS** | Parallel File System. A high performance file system utilizing block-based devices and POSIX file semantics. Examples include Lustre, GPFS, PVFS, and so on. |
| **POSIX** | Portable Operating System Interface. POSIX is a family of standards specified by the IEEE Computer Society for maintaining compatibility between operating systems. POSIX was introduced in 1988. |
| **Provenance** | Provenance is the chronology of the ownership, custody or location of a data object. |
| **Publish/ Subscribe** | Publish/Subscribe is a messaging pattern where senders of messages, called publishers, do not send the messages directly to specific receivers, called subscribers. Instead, published messages are characterized into classes, without knowledge of what, if any, subscribers there may be. Similarly, subscribers express interest in one or more classes, and only receive messages that are of interest, without knowledge of what, if any, publishers there are. |
| **RAM** | Random Access Memory. A type of memory that allows for data to be accessed quickly in any random order. |
| **RDF** | Resource Description Framewoark. a model for encoding semantic relationships between items of data so that these relationships can be interpreted computationally. |
| **RDMA** | Remote Direct Memory Access (RMA) is a direct memory access from the memory of one computer into that of another without involving either one's operating system. This permits high-throughput, low-latency networking, which is especially useful in massively parallel computers. |

| | |
|---|---|
| **SAS** | Serial Attached SCSI. A high-performance interface for block-based devices. SAS implementations typically outperform SCSI and SATA. (See also SCSI, SATA). |
| **SATA** | Serial ATA. A mid-level performance interface for block-based devices. (See also SAS, SCSI). |
| **SCSI** | Small Computer System Interface. A low-level performance parallel interface standard used by personal computers. (See also SAS, SATA). |
| **SQL** | Structured Query Language. SQL is a standardized programming language that is used to manage relational databases and perform various operations on the data in them. |
| **SSD** | Solid State Disk. A storage device compatible with traditional disks comprised of moving parts, but built with integrated circuits instead. SSDs typically offer improved latency and access time performance. However, they are roughly six to ten times more expensive per unit of storage than traditional hard-disk drives (see also HDD). |
| **SmartNIC** | Smart Network Interface Card. A SmartNIC is a programmable accelerator that makes data center networking, security and storage efficient and flexible. SmartNICs offload from server CPUs an expanding array of jobs required to manage modern distributed applications. |
| **URL** | Uniform Resource Locator. A URL, colloquially termed a web address, is a reference to a web resource that specifies its location on a computer network and a mechanism for retrieving it. |
| **Zero copy** | Operations in which the CPU does not perform the task of copying data from one memory area to another. This is most often used to save on processing power and memory use when sending files over a network. |

# 6. Attendees

## 6.1 Workshop Organizers

| First Name | Last Name | Affiliation | Role |
|---|---|---|---|
| Suren | Byna | Lawrence Berkeley National Laboratory | Organizer |
| Hal | Finkel | DOE/ASCR | Sponsor |
| Stratos | Idreos | Harvard | Organizer |
| Terry | Jones | Oak Ridge National Laboratory | Organizer |
| Margaret | Lentz | DOE/ASCR | Sponsor |
| Kathryn | Mohror | Lawrence Livermore National Laboratory | Organizer |
| Rob | Ross | Argonne National Laboratory | Organizer |
| Florin | Rusu | University of California, Merced | Organizer |

## 6.2 Writing Leads

| First Name | Last Name | Affiliation |
|---|---|---|
| George | Amvrosiadis | 3.4. Architectures and Services |
| Swen | Boehm | 3.1 High Productivity Interfaces |
| Suren | Byna | 3.3 Metadata and Provenance |
| Dong | Dai | 3.3 Metadata and Provenance |
| Terry | Jones | 3.1 High Productivity Interfaces |
| Gerald | Lofstead II | 3.1 High Productivity Interfaces |
| Kshitij | Mehta | 3.1 High Productivity Interfaces |
| Kathryn | Mohror | 3.2 Understanding the behavior |
| Bogdan | Nicolae | 3.4. Architectures and Services |
| Line | Pouchard | 3.3 Metadata and Provenance |
| Rob | Ross | 3.4. Architectures and Services |
| Florin | Rusu | 3.5. Cross-cutting themes |
| Nik | Sultana | 3.4. Architectures and Services |
| Patrick | Widener | 3.3 Metadata and Provenance |
| Matthew | Wolf | 3.1 High Productivity Interfaces |
| Justin | Wozniak | 3.1 High Productivity Interfaces |
| Mai | Zheng | 3.3 Metadata and Provenance |

## 6.3 Workshop Attendees

| First Name | Last Name | Affiliation |
|---|---|---|
| Deb | Agarwal | Lawrence Berkeley National Laboratory |
| George | Amvrosiadis | Carnegie Mellon University |
| Valentine | Anantharaj | Oak Ridge National Laboratory |
| James | Ang | Pacific Northwest National Laboratory |
| Molham | Aref | RelationalAI |
| Kazi | Asifuzzaman | Oak Ridge National Laboratory |
| David | Bader | New Jersey Institute of Technology |
| Deborah | Bard | Lawrence Berkeley National Laboratory |
| Oceane | Bel | Pacific Northwest National Laboratory |
| Douglas | Benjamin | Brookhaven National Laboratory |
| Wes | Bethel | Lawrence Berkeley National Laboratory |
| Jean Luca | Bez | Lawrence Berkeley National Laboratory |
| Suparna | Bhattacharya | Hewlett Packard Enterprise |
| Wahid | Bhimji | Lawrence Berkeley National Laboratory |
| Laura | Biven | NIH |
| Spyros | Blanas | The Ohio State University |
| Johannes | Blaschke | Lawrence Berkeley National Laboratory |
| Johannes | Blaschke | Lawrence Berkeley National Laboratory |
| Swen | Boehm | Oak Ridge National Laboratory |
| Philippe | Bonnet | IT University of Copenhagen |
| Aaron | Brewster | Lawrence Berkeley National Laboratory |
| Ron | Brightwell | Sandia National Laboratories |
| Michael | Brim | Oak Ridge National Laboratory |
| Christopher | Brislawn | Los Alamos National Laboratory |
| Joshua | Brown | Oak Ridge National Laboratory |
| David | Brown | Brookhaven National Laboratory |
| Ali | Butt | Virginia Tech |
| Suren | Byna | Lawrence Berkeley National Laboratory |
| Franck | Cappello | Argonne National laboratory |
| Richard | Carlson | US Department of Energy |
| Philip | Carns | Argonne National Laboratory |
| David | Castle | ISC World Data System |
| Stuart | Chalk | University of North Florida |

| | | |
|---|---|---|
| Iris | Chang | SLAC National Accelerator Laboratory |
| Aashish | Chaudhary | Kitware Inc. |
| Jieyang | Chen | Oak Ridge National Laboratory |
| Yong | Chen | Texas Tech University |
| Ganesh | Chennimalai Sankaran | University of Southern California |
| Alvin | Cheung | UC Berkeley |
| Shreyas | Cholia | Lawrence Berkeley National Laboratory |
| Joaquin | Chung | Argonne National Laboratory |
| Eric | Church | US Department of Energy |
| Heather | Coates | Indiana University-Purdue University Indianapolis |
| Guojing | Cong | Oak Ridge National Laboratory |
| Matthew | Curry | Sandia National Laboratories |
| Serena | Curzel | Pacific Northwest National Laboratory |
| Dong | Dai | University of North Carolina at Charlotte |
| Philip | Davis | University of Utah |
| Ewa | Deelman | University of Southern California |
| Alin | Deutsch | University of California San Diego |
| Hariharan | Devarajan | Lawrence Livermore National Laboratory |
| Sheng | Di | Argonne National Laboratory |
| Andreas | Dilger | Whamcloud |
| Bin | Dong | Lawrence Berkeley National Laboratory |
| Eugene | Duah | US Department of Energy |
| Benjamin | Dudson | Lawrence Livermore National Laboratory |
| Greg | Eisenhauer | Georgia Institute of Technology |
| Wael | Elwasif | Oak Ridge National Laboratory |
| Murali | Emani | Argonne National Laboratory |
| Christian | Engelmann | Oak Ridge National Laboratory |
| Christopher | Erdmann | American Geophysical Union (AGU) |
| Luke | Erikson | NNSA |
| David | Etim | NNSA |
| Lance | Evans | Hewlett Packard Enterprise |
| Kjiersten | Fagnan | Lawrence Berkeley National Laboratory |
| Paolo | Faraboschi | Hewlett Packard Enterprise |
| Michael | Feldman | Stanford University |
| Rafael | Ferreira da Silva | Oak Ridge National Laboratory |
| Hal | Finkel | US Department of Energy |

| | | |
|---|---|---|
| Martin | Foltin | Hewlett Packard Enterprise |
| Jon | Fortney | Oak Ridge National Laboratory |
| Ana | Gainaru | Oak Ridge National Laboratory |
| Carlos | Gamboa | Brookhaven National Laboratory |
| Vincent | Garonne | Brookhaven National Laboratory |
| Ada | Gavrilovska | Georgia Institute of Technology |
| Lisa | Gerhardt | Lawrence Berkeley National Laboratory |
| Devarshi | Ghoshal | Lawrence Berkeley National Laboratory |
| Meredith | Goins | World Data System International Programme Office |
| Matthias | Graf | US Department of Energy |
| Annette | Greiner | Lawrence Berkeley National Laboratory |
| Lipi | Gupta | Lawrence Berkeley National Laboratory |
| Hannah | Hamalainen | Los Alamos National Laboratory |
| Kevin | Harms | Argonne National Laboratory |
| Gerd | Heber | The HDF Group |
| Bryan | Hess | Thomas Jefferson National Accelerator Facility |
| Graham | Heyes | Thomas Jefferson National Accelerator Facility |
| Dean | Hildebrand | Google |
| Thuc | Hoang | NNSA |
| Christopher | Hogan | The HDF Group |
| Aaron | Holder | US Department of Energy |
| Stratos | Idreos | Harvard |
| Keith | Jankowski | US Department of Energy |
| Henry | JIN | NASA |
| Dylan | Johnson | University of North Florida |
| Terry | Jones | Oak Ridge National Laboratory |
| Sudarsun | Kannan | Rutgers University |
| Ahmad Maroof | Karimi | Oak Ridge National Laboratory |
| Bryan | Kim | Syracuse University |
| Scott | Klasky | Oak Ridge National Laboratory |
| Martin | Klein | Los Alamos National Laboratory |
| Kathryn | Knight | Oak Ridge National Laboratory |
| Annmary | Koomthanam | Hewlett Packard Enterprise |
| Anthony | Kougkas | Illinois Institute of Technology |
| Quincey | Koziol | Amazon Web Services |
| Olga | Kuchar | Oak Ridge National Laboratory |

| | | |
|---|---|---|
| Resham | Kulkarni | US Department of Energy |
| Kuldeep | Kurte | Oak Ridge National Laboratory |
| Eric | Lancon | Brookhaven National Laboratory |
| Matthew | Lanctot | US Department of Energy |
| Daniel | Laney | Lawrence Livermore National Laboratory |
| Rob | Latham | Argonne National Laboratory |
| Amber | Lauer-Coles | Brookhaven National Laboratory |
| Jerome | Lauret | Brookhaven National Laboratory |
| Steven | Lee | US Department of Energy |
| Margaret | Lentz | US Department of Energy |
| Dong | Li | University of California, Merced |
| Xin | Liang | Missouri S&T |
| Chunhua | Liao | Lawrence Livermore National Laboratory |
| Ankur | Limaye | Pacific Northwest National Laboratory |
| Zhengchun | Liu | Argonne National Laboratory |
| Xu | Liu | North Carolina State University |
| Qing | Liu | New Jersey Institute of Technology |
| Glenn | Lockwood | Lawrence Berkeley National Laboratory |
| Gerald | Lofstead II | Sandia National Laboratories |
| Johann | Lombardi | Intel |
| Lawrence | London | Balex Technologies |
| Burlen | Loring | Lawrence Berkeley National Laboratory |
| Jeremy | Love | US Department of Energy |
| Matt | Macduff | Pacific Northwest National Laboratory |
| Sandeep | Madireddy | Argonne National Laboratory |
| Ramana | Madupu | US Department of Energy |
| Ketan | Maheshwari | Oak Ridge National Laboratory |
| Kirill | Malkin | Hewlett Packard Enterprise |
| Carlos | Maltzahn | University of California, Santa Cruz |
| Edmund J | Mansky II | NASA |
| Andres | Marquez | Pacific Northwest National Laboratory |
| Donnie | Mason | Brookhaven National Laboratory |
| Alex | May | Oak Ridge National Laboratory |
| Elizabeth | McCutchan | Brookhaven National Laboratory |
| Marshall | McDonnell | Oak Ridge National Laboratory |
| Kshitij | Mehta | Oak Ridge National Laboratory |

| | | |
|---|---|---|
| Claudia | Mewes | US Department of Energy |
| Raul | Miranda | US Department of Energy |
| Shigeki | Misawa | Brookhaven National Laboratory |
| Kathryn | Mohror | Lawrence Livermore National Laboratory |
| Thomas | Naughton | Oak Ridge National Laboratory |
| Bogdan | Nicolae | Argonne National Laboratory |
| Angela | Norbeck | Pacific Northwest National Laboratory |
| Mark | Nossokoff | Hyperion Research |
| Sarp | Oral | Oak Ridge National Laboratory |
| Ippokratis | Pandis | Amazon Web Services |
| Manish | Parashar | University of Utah |
| Sean | Peisert | Lawrence Berkeley National Laboratory |
| Ivy | Peng | Lawrence Livermore National Laboratory |
| Talita | Perciano | Lawrence Berkeley National Laboratory |
| Lori | Perkins | NASA |
| Martin | Perlmutter | Balex Technologies |
| Jordan | Perr-Sauer | National Renewable Energy Laboratory |
| Tom | Peterka | Argonne National Laboratory |
| Bei | Phillips | University of Utah |
| Robinson | Pino | US Department of Energy |
| Norbert | Podhorszki | Oak Ridge National Laboratory |
| Line | Pouchard | Brookhaven National Laboratory |
| Elena | Pourmal | The HDF Group |
| Giri | Prakash | Oak Ridge National Laboratory |
| Branislav | Radovanovic | Balex Technologies |
| Ioan | Raicu | Illinois Institute of Technology |
| Lavanya | Ramakrishnan | Lawrence Berkeley National Laboratory |
| Tejas | Rao | Brookhaven National Laboratory |
| Silvio | Rizzi | Argonne National Laboratory |
| Ivan | Rodero | University of Utah |
| David | Rogers | Oak Ridge National Laboratory |
| Brandon | Rohnke | US Department of Energy |
| Rob | Ross | Argonne National Laboratory |
| Thomas | Russell | US Department of Energy |
| James | Rustad | US Department of Energy |
| Florin | Rusu | University of California, Merced |

| | | |
|---|---|---|
| Rajesh | Sankaran | Argonne National Laboratory |
| Ada | Sedova | Oak Ridge National Laboratory |
| Saba | Sehrish | Fermi National Accelerator Laboratory |
| Tom | Settersten | US Department of Energy |
| Bradley | Settlemyer | Los Alamos National Laboratory |
| John | Shalf | Lawrence Berkeley National Laboratory |
| Galen | Shipman | Los Alamos National Laboratory |
| Horst D. | Simon | Lawrence Berkeley National Laboratory |
| Chitra | Sivaraman | Pacific Northwest National Laboratory |
| Marc | Snir | University of Illinois at Urbana-Champaign |
| Jerome | Soumagne | The HDF Group |
| Rohit | Srivastava | Oak Ridge National Laboratory |
| Dale | Stansberry | Oak Ridge National Laboratory |
| David | Stevens | Lawrence Livermore National Laboratory |
| Nik | Sultana | Illinois Institute of Technology |
| Ceren | Suset-Bennett | US Department of Energy |
| Nathan | Tallent | Pacific Northwest National Laboratory |
| Houjun | Tang | Lawrence Berkeley National Laboratory |
| Aditya | Tanikanti | Argonne National Laboratory |
| Dingwen | Tao | Washington State University |
| Nicholas | Taylor | Los Alamos National Laboratory |
| Rajeev | Thakur | Argonne National Laboratory |
| Devesh | Tiwari | Northwestern University |
| Aalap | Tripathy | Hewlett Packard Enterprise |
| Antonino | Tumeo | Pacific Northwest National Laboratory |
| Nicholas | Tyler | Lawrence Berkeley National Laboratory |
| Jeffrey | Ullman | Stanford University |
| Peter | van Gemmeren | Argonne National Laboratory |
| Kaushik | Velusamy | Argonne National Laboratory |
| Carlos | Verdoza | Pacific Northwest National Laboratory |
| Lucas | Villa Real | IBM Research |
| Venkatram | Vishwanath | Argonne National Laboratory |
| Matt | Wallis | Dell Technologies |
| Amy | Walton | National Science Foundation |
| Lipeng | Wan | Oak Ridge National Laboratory |
| Zhe | Wang | Rutgers University |

| | | |
|---|---|---|
| Chen | Wang | University of Illinois at Urbana-Champaign |
| Lee | Ward | Sandia National Laboratories |
| Patrick | Widener | Sandia National Laboratories |
| Sean | Wilkinson | Oak Ridge National Laboratory |
| Felix | Wittwer | Lawrence Berkeley National Laboratory |
| Matthew | Wolf | Oak Ridge National Laboratory |
| Justin | Wozniak | Argonne National Laboratory |
| John | Wu | Lawrence Berkeley National Laboratory |
| Meng | Xu | Apple |
| Guohui | Yuan | US Department of Energy |
| Wei | Zhang | Texas Tech University |
| Junbo | Zhao | University of Connecticut |
| Huihuo | Zheng | Argonne National Laboratory |
| Mai | Zheng | Iowa State University |
| Christopher | Zimmer | Oak Ridge National Laboratory |

Back to TOC

# 7. Workshop Agenda

## Day 1 (January 24)

| Time | Topic |
|---|---|
| **12:00 - 12:15** | Opening Remarks (Hal Finkel, ASCR) |
| **12:15 - 12:45** | Introduction and Logistics (Organizing Committee) - Terry Jones |
| **12:45 - 1:30** | Keynote - Giri Prakash |
| **1:30 - 3:00** | Panel: Workflows (Moderator: Kathryn Mohror)<br>• Lavanya Ramakrishnan<br>• Dan Laney<br>• Rafael Ferreira da Silva<br>• Philip Davis<br>• Tom Peterka<br>• Ewa Deelman |
| **3:00 - 3:15** | Break |
| **3:15 - 4:15** | Breakouts<br>1. Understanding the overlap between traditional storage systems and I/O (SSIO) efforts and data management<br>a. Session leads: Carlos Maltzahn, Lance Evans<br>b. Note takers: Burlen Loring<br>2. Data management support for AI and complex workflows<br>a. Session leads: Rafael Ferreira da Silva, Lavanya Ramakrishnan<br>b. Note takers: Hariharan Devarajan, Ana Gainaru<br>3. Novel architectures for scientific data (Data warehouses, lakes, cloud, and reconfigurable storage)<br>a. Session leads: Sarp Oral, John Shalf<br>b. Note takers: Kevin Harms, Glenn Lockwood |
| **4:15 - 5:00** | Readouts / Summary |

## Day 2 (January 25)

| Time | Topic |
|---|---|
| 12:00 - 12:05 | Opening Remarks and Logistics |
| 12:05 - 12:50 | Keynote - Pandis Ippokratis |
| 12:50 - 2:00 | Panel: Database technologies for scientific applications (Moderator: Rob Ross)<br><br>● Ioan Raicu<br>● Jay Lofstead<br>● Lee Ward<br>● John Wu<br>● Spyros Blanas |
| 2:00 - 2:05 | Break |
| 2:05 - 3:00 | Breakouts, First Session<br>1. Storage-system architecture design<br>a. Session leads: Rob Ross, Sudarsun Kannan<br>b. Note takers: Galen Shipman, Nik Sultana<br>2. Capturing and using provenance information in data life cycle<br>a. Session leads: Aaron Brewster Katie Knight<br>b. Note takers: Phil Carns, Justin Wozniak<br>3. AI for data management<br>a. Session leads: Devesh Tiwari, Wahid Bhimji<br>b. Note takers: Sandeep Madireddy, Murali Emani |
| 3:00 - 3:15 | Break |
| 3:15 - 4:15 | Breakouts, Second Session<br>1. Interfaces for accessing scientific data<br>a. Session leads: Suren Byna, Johann Lambardi<br>b. Note taker: Gerd Heber, John Shalf<br>2. Data management and storage needs of scientific applications<br>a. Session leads: Jay Lofstead, Antonino Tumeo<br>b. Note taker: Marshall McDonnell<br>3. Understanding application data movement and management<br>a. Session leads: Phil Carns, Glenn Lockwood<br>b. Note taker: Dong Dai, Kathryn Mohror, Rob Ross |
| 4:15 - 5:00 | Readouts / Summary (Breakouts, First Session) |

## Day 3 (January 27)

<table>
<tr><th>Time</th><th>Topic</th></tr>
<tr><td>12:00 - 12:05</td><td>Opening Remarks and Logistics</td></tr>
<tr><td>12:05 - 12:50</td><td>Keynote - Deb Agarwal</td></tr>
<tr><td>12:50 - 2:00</td><td>Panel: Metadata and provenance management (Moderator: Suren Byna)<ul><li>Katie Knight</li><li>Yong Chen</li><li>Patrick Widener</li><li>Kjiersten Fagnan</li><li>Galen Shipman</li></ul></td></tr>
<tr><td>2:00 - 2:05</td><td>Break</td></tr>
<tr><td>2:05 - 2:50</td><td>Readouts / Summary (Breakouts, Day 2, Second Session)</td></tr>
<tr><td>2:50 - 3:45</td><td>Breakouts<ol><li>Data management co-design for edge and HPC applications<ol type="a"><li>Session leads: John Wu, Terry Jones</li><li>Note takers: Glenn Lockwood</li></ol></li><li>Data management support for AI and complex workflows<ol type="a"><li>Session leads: Scott Klasky, George Amvrosiadis</li><li>Note takers: Huihuo Zheng, Burlen Loring</li></ol></li><li>Metadata management infrastructure to support FAIR principles<ol type="a"><li>Session leads: Kjiersten Fagnan, Line Pouchard</li><li>Note takers: Bogdan Nicole</li></ol></li><li>Interfaces for accessing scientific data<ol type="a"><li>Session leads: Rob Ross, Justin Wozniak</li><li>Note takers: Hariharan Devarajan</li></ol></li></ol></td></tr>
<tr><td>3:45 - 4:00</td><td>Break</td></tr>
<tr><td>4:00 - 5:00</td><td>Readouts / Summary</td></tr>
</table>

# 8. References


[Adhianto, 2010] Adhianto, L., S. Banerjee, M. Fagan, M. Krentel, G. Marin, J. Mellor-Crummey, and N.R. Tallent. HPCTOOLKIT: Tools for Performance Analysis of Optimized Parallel Programs. Concurrency Computat.: Pract. Exper., 22: 685-701. https://doi.org/10.1002/cpe.1553.

[Agarwal, 2019] Megha Agarwal, Divyansh Singhvi, Preeti Malakar, and Suren Byna. "Active Learning-Based Automatic Tuning and Prediction of Parallel I/O Performance", PDSW 2019, in conjunction with SC19.

[Agelastos, 2014] Agelastos, A., et al. "The Lightweight Distributed Metric Service: A Scalable Infrastructure for Continuous Monitoring of Large Scale Computing Systems and Applications," SC '14: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 154-165, doi: 10.1109/SC.2014.18.

[Albrecht, 2012] Albrecht, Michael, Patrick Donnelly, Peter Bui, and Douglas Thain. "Makeflow: A Portable Abstraction for Data Intensive Computing on Clusters, Clouds, and Grids." In Proceedings of the 1st ACM SIGMOD Workshop on Scalable Workflow Execution Engines and Technologies, pp. 1-13.

[ALCF, 2013] Argonne Leadership Computing Facility. "ALCF I/O Data Repository." Technical Memorandum ANL/ALCF/TM-13/1, Argonne National Laboratory, February 2013.

[Alvaro, 2015] Alvaro, Peter, Joshua Rosen, and Joseph M. Hellerstein. "Lineage-driven fault injection." In ACM SIGMOD.

[Amazon, 2006] Amazon Web Services. 2019. Amazon Simple Storage Service is a service offered by Amazon Web Services (AWS) that provides object storage through a web service interface. https://en.wikipedia.org/wiki/Amazon_S3

[Ames, 2005] Ames, Alexander, Nikhil Bobb, Scott A. Brandt, Adam Hiatt, Carlos Maltzahn, Ethan L. Miller, Alisa Neeman, and Deepa Tuteja. "Richer file system metadata using links and attributes." In 22nd IEEE/13th NASA Goddard Conference on Mass Storage Systems and Technologies (MSST'05), pp. 49-60. IEEE.

[Ames, 2013] Ames, Sasha, Maya Gokhale, and Carlos Maltzahn. "QMDS: a file system metadata management service supporting a graph data model-based query language." International Journal of Parallel, Emergent and Distributed Systems 28, no. 2, pp. 159-183.

[Apache Arrow] Apache Software Foundation, Arrow, A cross-language development platform for in-memory data. https://arrow.apache.org. Accessed June 2022.

[Apache Hadoop] Apache Hadoop. https://hadoop.apache.org. Accessed June 2022.

[Apache Iceberg] Apache Iceberg. https://iceberg.apache.org. Accessed June 2022.

[Apache Kafka] Apache Kafka. https://kafka.apache.org. Accessed June 2022.

[Apache OpenWhisk] Apache OpenWhisk. https://openwhisk.apache.org. Accessed June 2022.

[Apptainer] https://apptainer.org. Accessed June 2022.

[Armbrust, 2020] Armbrust, Michael, Tathagata Das, Liwen Sun, Burak Yavuz, Shixiong Zhu, Mukul Murthy, Joseph

Torres, et al. "Delta Lake: High-Performance ACID Table Storage over Cloud Object Stores." In Proceedings of the VLDB Endowment 13, no. 12, pp. 3411-3424.

[Amstutz, 2022] Amstutz, Peter,Maxim Mikheev, Michael R. Crusoe, Nebojša Tijanić, Samuel Lampa, et al. Existing Workflow Systems. *Common Workflow Language wiki*, GitHub. https://s.apache.org/existing-workflow-systems updated June 20, 2022, accessed June 20, 2022.

[AWS] Amazon Web Services: High Performance Computing: https://aws.amazon.com/hpc/. Accessed June 2022.

[Awtrey, 2021] Nikolas Awtrey, heatmap figure support contributed to Darshan source code, https://github.com/darshan-hpc/darshan/pull/396

[Azevedo, 2020] Guerreiro Azevedo, Leonardo, Renan Souza, Raphael Melo Thiago, Elton Soares, and Marcio Moreno. "Experiencing ProvLake to Manage the Data Lineage of AI Workflows," IBM Research, http://ibm.biz/provlake, 2020.

[Azure] Azure high-performance computing: https://azure.microsoft.com/en-us/solutions/high-performance-computing. Accessed June 2022.

[Barkes, 1998] Barkes, J. M. R. F. Cougard, P. G. Crumley, D. Marin, H. Reddy, and T. Thitayanun, T. GPFS: A Parallel File System. IBM International Technical Support Organization. available online.

[Bates, 2015] Bates, Adam, Dave Jing Tian, Kevin R. B. Butler, and Thomas Moyer. "Trustworthy Whole-System Provenance for the Linux Kernel." In 24th USENIX Security Symposium (USENIX Security 15), pp. 319-334. 2015.

[Behzad, 2013] Behzad, B., H. Luu, J. Huchette, S. Byna, Prabhat, R. Aydt, Q. Koziol, and M. Snir, "Taming Parallel I/O Complexity with Auto-Tuning." ACM/IEEE Supercomputing 2013 (SC13).

[Bez, 2021] Bez, J. L., et al. "I/O Bottleneck Detection and Tuning: Connecting the Dots using Interactive Log Analysis." 2021 IEEE/ACM Sixth International Parallel Data Systems Workshop (PDSW), pp. 15-22, doi: 10.1109/PDSW54622.2021.00008.

[binder] https://mybinder.org. Accessed June 2022.

[Biton, 2007] Biton, Olivier, Sarah Cohen Boulakia, and Susan B. Davidson. "Zoom* UserViews: Querying Relevant Provenance in Workflow Systems." In VLDB, vol. 7, pp. 1366-1369. 2007.

[Bjørling, 2021] Bjørling, Matias, Abutalib Aghayev, Hans Holmberg, Aravind Ramesh, Damien Le Moal, Gregory R. Ganger, and George Amvrosiadis. "ZNS: Avoiding the Block Interface Tax for Flash-based SSDs." In Proceedings of 2021 USENIX Annual Technical Conference, pp. 689-703.

[Bonnie, 2011] Bonnie, M. M. D., Ligon, B., Marshall, M., Ligon, W., Mills, N., Sampson, E. Q. S., ... and Wilson, B. OrangeFS: Advancing PVFS. In USENIX Conference on File and Storage Technologies (FAST).

[Braam, 1999] Braam, P. J., and Callahan, M. J. "Lustre: A SAN File System for Linux. "white paper.

[Buneman, 2001] Buneman, Peter, Sanjeev Khanna, and Wang-Chiew Tan. "Why and Where: A Characterization of Data Provenance." In Proceedings of the 8th International Conference on Database Theory (ICDT).

[Byna, 2018] Byna, Suren, Quincey Koziol, Venkatram Vishwanath, Jerome Soumagne, Houjun Tang, Kimmy Mu, Richard Warren, François Tessier, Bin Dong, Teng Wang, and Jialin Liu. "Proactive Data Containers (PDC): An object-centric data store for large-scale computing systems." AGU Fall Meeting.

[Byna, 2020] Byna, Suren, M. Scot Breitenfeld, Bin Dong, Quincey Koziol, Elena Pourmal, Dana Robinson, Jerome Soumagne, Houjun Tang, Venkatram Vishwanath, and Richard Warren. "ExaHDF5: Delivering Efficient Parallel I/O on Exascale Computing Systems." Journal of Computer Science and Technology35(1): 145-160. DOI: 10.1007/s11390-020-9822-9.

[Callahan, 2006] Callahan, Steven P., Juliana Freire, Emanuele Santos, Carlos E. Scheidegger, Cláudio T. Silva, and Huy T. Vo. "VisTrails: visualization meets data management." In Proceedings of the 2006 ACM SIGMOD international conference on Management of Data, pp. 745-747.

[Cao, 2020] Cao, Wei, et al. "POLARDB Meets Computational Storage: Efficiently Support Analytical Workloads in Cloud-Native Relational Database." 18th USENIX Conference on File and Storage *Technologies* (*FAST 20*).

[Carata, 2013] Carata, Lucian, Ripduman Sohan, Andrew Rice, and Andy Hopper. "IPAPI: Designing an Improved Provenance API." In 5th USENIX Workshop on the Theory and Practice of Provenance (TaPP 13).

[Carns, 2000] Carns, Philip H., Walter B. Ligon III, Robert B. Ross, and Rajeev Thakur. "PVFS: A Parallel File System for Linux Clusters." In 4th Annual Linux Showcase & Conference (ALS 2000.

[Carns, 2011] Carns, Philip, Kevin Harms, William Allcock, Charles Bacon, Samuel Lang, Robert Latham, and Robert Ross. "Understanding and Improving Computational Science Storage Access through Continuous Characterization", in ACM Transactions on Storage 7, pp. 8:1-8:26.

[Carragáin, 2019] Carragáin, Eoghan Ó., Carole A. Goble, Peter Sefton, and Stian Soiland-Reyes. "RO-Crate, a Lightweight Approach to Research Object Data Packaging." In RO. 2019.

[Casacore] https://github.com/casacore/casacore. Accessed June 2022.

[Castro, 2019] Castro, Paul, Vatche Ishakian, Vinod Muthusamy, and Aleksander Slominski. "The Rise of Serverless Computing." Communications of the ACM
62, no. 12, pp. 44-54.

[Chamberlin, 1974] Chambcrlin, D. D., & Boyce, R. F. (1974). SEQUEL: A Structured English Query Language: Proc. In 1974 ACIvl SIGFIDET Workshop, Ann Arbor, Michigan, pp. 249-264. DOI: 10.1145/800296.811515

[Chowdhury, 2020] Chowdhury, F., Y. Zhu, F. Di Natale, A. Moody, E. Gonsiorowski, K. Mohror, and W. Yu. "Emulating I/O Behavior in Scientific Workflows on High Performance Computing Systems." 5th International Parallel Data Systems Workshop. November 2020, Virtual Workshop.

[Chou, 2011] Chou, J., M. Howison, B. Austin, K. Wu, J. Qiang, E. W. Bethel, A. Shoshani, O. Rübel, and R. D. Ryne. "Parallel Index and Query for Large Scale Data Analysis." Proc. of 2011 International Conference for High Performance Computing, Networking, Storage, and Analysis (SC 2011).

[Ciccarese, 2013] Ciccarese, Paolo, Stian Soiland-Reyes, Khalid Belhajjame, Alasdair JG Gray, Carole Goble, and Tim Clark. "PAV Ontology: Provenance, Authoring and Versioning." Journal of Biomedical Semantics 4, no. 1, pp. 1-22.
[Computational Storage, 2022] https://www.xilinx.com/applications/data-center/computational-storage/smartssd.html

[Costa, 2021] Costa, Emily, Tirthak Patel, Benjamin Schwaller, Jim M. Brandt, and Devesh Tiwari. "Systematically

Inferring I/O Performance Variability by Examining Repetitive Job Behavior." In Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 1-15.

[Coughlin, 2013] Coughlin, T. (2013). Evolving Storage Technology in Consumer Electronic Products [The Art of Storage]. *IEEE Consumer Electronics Magazine*, *2*(2), pp. 59-63.

[CRAM] CRAM. "Computational RAM." https://www.eecg.utoronto.ca/~dunc/cram/. Accessed June 2022.

[Crusoe, 2021] Crusoe, Michael R., Sanne Abeln, Alexandru Iosup, Peter Amstutz, John Chilton, Nebojša Tijanić, Hervé Ménager, Stian Soiland-Reyes, Carole Goble, The CWL Community: Methods Included: Standardizing Computational Reuse and Portability with the Common Workflow Language. arXiv 2105.07028 [cs.DC] https://arxiv.org/abs/2105.07028

[CXL] CXL. "Computer Express Link". https://www.computeexpresslink.org/about-cxl. Accessed June 2022.

[da Silva, 2021] da Silva, R. F>, et al. "A Community Roadmap for Scientific Workflows Research and Development," *2021 IEEE Workshop on Workflows in Support of Large-Scale Science* (*WORKS*), pp. 81-90, doi: 10.1109/WORKS54523.2021.00016.

[Dai, 2014] Dai, Dong, Yong Chen, Dries Kimpe, and Rob Ross. "Provenance-Based Object Storage Prediction Scheme for Scientific Big Data Applications." In 2014 IEEE International Conference on Big Data, pp. 271-280.

[Dai, 2016] DI, Dong, Yong Chen, Philip Carns, John Jenkins, Wei Zhang, and Robert Ross. "GraphMeta: A Graph-Based Engine for Managing Large-Scale HPC Rich Metadata." In Proceedings of the IEEE International Conference on Cluster Computing, 2016.

[Dai, 2017] Dai, Dong, Yong Chen, Philip Carns, John Jenkins, and Robert Ross. "Lightweight Provenance Service for High-Performance Computing." In 2017 26th International Conference on Parallel Architectures and Compilation Techniques (PACT), pp. 117-129. IEEE.

[Dai, 2018] Dai, Dong, Yong Chen, Philip Carns, John Jenkins, Wei Zhang, and Robert Ross. "Managing Rich Metadata in High-Performance Computing Systems Using a Graph Model." IEEE Transactions on Parallel and Distributed Systems 30, no. 7, pp. 1613-1627.

[Deelman, 2015] Deelman, Ewa, Karan Vahi, Gideon Juve, Mats Rynge, Scott Callaghan, Philip J. Maechling, Rajiv Mayani, et al. "Pegasus, a Workflow Management System for Science Automation." Future Generation Computer Systems 46, pp. 17-35.

[Del Rosario, 2020] Del Rosario, Eliakin, Mikaela Currier, Mihailo Isakov, Sandeep Madireddy, Prasanna Balaprakash, Philip Carns, Robert B. Ross, Kevin Harms, Shane Snyder, and Michel A. Kinsy. "Gauge: An Interactive Data-Driven Visualization Tool for HPC Application I/O Performance Analysis." In 2020 IEEE/ACM Fifth International Parallel Data Systems Workshop (PDSW), pp. 15-21.

[Devarajan, 2021] Devarajan, H., H. Zheng, A. Kougkas, X. -H. Sun and V. Vishwanath, "DLIO: A Data-Centric Benchmark for Scientific Deep Learning Applications," *2021* IEEE/ACM 21st International Symposium on Cluster, Cloud and Internet Computing (CCGrid), pp. 81-91, doi: 10.1109/CCGrid51090.2021.00018.

[Dickson, 2016] Dickson, J., S. Wright, S. Maheswaran, A. Herdman, M. C. Miller, and S. Jarvis, "Replicating HPC I/O Workloads with Proxy Applications." 2016 1st Joint International Workshop on Parallel Data Storage and Data Intensive Scalable Computing Systems (PDSW-DISCS), pp. 13-18, doi: 10.1109/PDSW-DISCS.2016.007.

[Dividino, 2009] Dividino, Renata, Sergej Sizov, Steffen Staab, and Bernhard Schueler. "Querying for Provenance, Trust, Uncertainty and Other Meta Knowledge in RDF." Journal of Web Semantics 7, no. 3, pp. 204-219.

[DOE, 2022] DOE Office of Advanced Scientific Computing Research. https://science.osti.gov/ascr. Accessed June 2022.

[Dong, 2016] Dong, Bin, Suren Byna, Kesheng Wu, Prabhat, Hans Johansen, Jeffrey N. Johnson, and Noel Keen. "Data Elevator: Low-contention Data Movement in Hierarchical Storage System." The 23rd annual IEEE International Conference on High Performance Computing, Data, and Analytics (HiPC).

[Dreherand, 2017] Dreherand, M., and T. Peterka, "Decaf: Decoupled Data Flows for In-Situ High-Performance Workflows." Technical report ANL/MCS-TM-371.

[Embley, 1980] Embley, D. W. "Programming with Data Frames for Everyday Data Items. In Proceedings of the May 19-22, 1980, AFIPS national computer conference, pp. 301-305. DOI: 10.1145/1500518.1500564

[Ethernet] Ethernet. "IEEE P802.3df 200Gb/s, 400Gb/s, 800Gb/s, and 1.6Tb/s Ethernet Task Force." https://www.ieee802.org/3/df/index.html

[Fagnan, 2019] Fagnan, K., Y. Nashed, G. Perdue, D. Ratner, A. Shankar, and S. Yoo. "Data and Models: A Framework for Advancing AI in Science." USDOE Office of Science (SC)(United States). available online.

[FAIR4HEP] Findable, Accessible, Interoperable, and Reusable Frameworks for Physics-Inspired Artificial Intelligence in High Energy Physics, https://fair4hep.github.io. Accessed June 2022.

[fio, 2022] fio - Flexible I/O tester rev. 3.27, https://fio.readthedocs.io/en/latest/fio_doc.html, Accessed April 2022.

[Firestone, e2018] Firestone, Daniel, et al. "Azure Accelerated Networking: SmartNICs in the Public Cloud." 15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18).

[Folk, 2011] Folk, M., G. Heber, Q. Koziol, E. Pourmal, and D. Robinson, D. "An Overview of the HDF5 Technology Suite and Its Applications. In Proceedings of the EDBT/ICDT 2011 Workshop on Array Databases, pp. 36-47.

[Freire, 2008] Freire, Juliana, David Koop, Emanuele Santos, and Cláudio T. Silva. "Provenance for Computational Tasks: A Survey." Computing in Science & Engineering 10, no. 3, pp. 11-21.

[Gehani, 2012] Gehani, Ashish, and Dawood Tariq. "SPADE: Support for Provenance Auditing in Distributed Environments," In Narasimhan, P., Triantafillou, P. (eds) Middleware 2012. Middleware 2012. Lecture Notes in Computer Science, vol 7662. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-35170-9_6

[Gen-Z] Gen-Z. "Gen-Z: An Open Systems Interconnect Designed to Provide Memory-Semantic Access to Data and Devices via Direct-Attached, Switched, or Fabric Topologies." https://genzconsortium.org

[Ghose, 2019] Ghose, S., A. Boroumand, J. S. Kim, J. Gómez-Luna, and O. Mutlu, "Processing-in-Memory: A Workload-Driven Perspective." IBM Journal of Research and Development 63, no. 6, pp. 3:1-3:19, doi: 10.1147/JRD.2019.2934048.

[Ghoshal, 2017] Ghoshal, Devarshi, and Lavanya Ramakrishnan. "MaDaTS: Managing Data on Tiered Storage for Scientific Workflows. In Proceedings of the 26th International Symposium on High-Performance Parallel and

Distributed Computing (HPDC '17). Association for Computing Machinery, New York, pp. 41–52. DOI:https://doi.org/10.1145/3078597.3078611

[Ghoshal, 2021] Ghoshal, Devarshi, Ludovico Bianchi, Abdelilah Essiari, Michael Beach, Drew Paine, and Lavanya Ramakrishnan. "Science Capsule-Capturing the Data Life Cycle." Journal of Open Source Software 6, no. 62 (2021).

[Goble, 2020] Goble, Carole, Sarah Cohen-Boulakia, Stian Soiland-Reyes, Daniel Garijo, Yolanda Gil, Michael R. Crusoe, Kristian Peters, and Daniel Schober. "FAIR Computational Workflows." Data Intelligence 2 (1-2), pp. 108–121. doi: https://doi.org/10.1162/dint_a_00033

[GoogleCloud] Google Cloud: High performance computing: https://cloud.google.com/solutions/hpc. Accessed June 2022.

[Greenburg, 2015] Greenberg, H., J. Bent, and G. Grider, G. "MDHIM: A Parallel Key/Value Framework for HPC." In *7th USENIX Workshop on Hot Topics in Storage and File Systems* (*HotStorage 15*).

[Grimsrud, 2003] Grimsrud, K., and H. Smith. Serial ATA Storage Architecture and Applications: Designing High-Performance, Cost-Effective I/O Solutions. Intel Press.

[Guo, 2011] Guo, Philip J., and Dawson Engler. "CDE: Using System Call Interposition to Automatically Create Portable Software Packages." In 2011 USENIX Annual Technical Conference (USENIX ATC 11).

[Han, 2022] Runzhou Han, Suren Byna, Houjun Tang, Bin Dong, and Mai Zheng. "PROV-IO: An I/O-Centric Provenance Framework for Scientific Data on HPC Systems." The 31st International Symposium on High-Performance Parallel and Distributed Computing (HPDC).

[HDFGroup, 2012] HDF Group. 2012. Virtual Object Layer in HDF5. available online.

[Hedges, 2005] Hedges, R., B. Loewe, T. McLarty, and C. Morrone. "Parallel File System Testing for the Lunatic Fringe: The Care and Feeding of Restless I/O Power Users." 22nd IEEE / 13th NASA Goddard Conference on Mass Storage Systems and Technologies (MSST'05), pp. 3-17, doi: 10.1109/MSST.2005.22.

[Heichler, 2014] Heichler, J. "An introduction to BeeGFS." available online.

[Hewitt, 2010] Hewitt, E. "Cassandra: The Definitive Gguide." O'Reilly Media, Inc.".

[Hills, 2015] Hills, Denise, Robert R. Downs, Ruth Duerr, Justin C. Goldstein, Mark A. Parsons, and Hampapuram K. Ramapriyan. "The Importance of Data Set Provenance for Science." Eos 96, no. 10.1029. https://eos.org/opinions/the-importance-of-data-set-provenance-for-science

[Huawei, 2018] Huawei. "Oceanstor 9000: A Cloud Storage System Designed for High Performance and fFexible Scalability. https://support.huawei.com/enterprise/en/cloud-storage/oceanstor-9000-pid-8291244

[HXHIM] "The Hexadimensional Hashing Indexing Middleware (HXHIM). https://github.com/hpc/hxhim
"
[IBMCloud] High-Performance Computing on IBM Cloud." https://www.ibm.com/cloud/hpc. Accessed June 2022.

[Intel, 2022] Intel Platform Analysis Technology, https://www.intel.com/content/www/us/en/develop/documentation/vtune-help/top.html, accessed April 2022.

[IOBAT, 2022] IOBAT Documentation, https://iobat.readthedocs.io/en/latest/, Accessed April 2022.

[Isakov, 2020] lo Isakov, Mihai, Eliakin Del Rosario, Sandeep Madireddy, Prasanna Balaprakash, Philip Carns, Robert B. Ross, and Michel A. Kinsy. "HPC I/O Throughput Bottleneck Analysis with Explainable Local Models." In SC20: International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 1-13. IEEE.

[Jones, 2001] Jones, Matthew B., Chad Berkley, Jivka Bojilova, and Mark Schildhauer. "Managing Scientific Metadata." IEEE Internet Computing 5, no. 5, pp. 59-68.

[Jones, 2017] Jones, T., Brim, M. J., Vallee, G., Mayer, B., Welch, A., Li, T., ... and Fernando, P. Unity: Unified Memory and File Space. In Proceedings of the 7th International Workshop on Runtime and Operating Systems for Supercomputers ROSS 2017, pp. 1-8. DOI: 10.1145/3095770.3095776

[Kakola, 1996] Kakola, T. K., and Koota, K. I." Redesigning Work with Dual Information Systems: The Work Process Benchmarking Service." In Proceedings of HICSS-29: 29th Hawaii International Conference on System Sciences, Vol. 3, pp. 461-471. IEEE. DOI: 10.1109/HICSS.1996.493242

[Kang, 2013] Kang, Yangwook, Yang-suk Kee, Ethan L. Miller, and Chanik Park. "Enabling Cost-Effective Data Processing with Smart SSD." In 2013 IEEE 29th symposium on Mass Storage Systems and Technologies (MSST), pp. 1-12.

[Katsifodimos, 2016] Katsifodimos and S. Schelter. "Apache Flink: Stream Analytics at Scale." 2016 IEEE International Conference on Cloud Engineering Workshop (IC2EW), pp. 193-193, doi: 10.1109/IC2EW.2016.56.

[Kelly, 2020] Kelly, Christopher, Sungsoo Ha, Kevin Huck, Hubertus Van Dam, Line Pouchard, Gyorgy Matyasfalvi, Li Tang et al. "Chimbuko: A Workflow-Level Scalable Performance Trace Analysis Tool." In ISAV'20 In Situ Infrastructures for Enabling Extreme-Scale Analysis and Visualization, pp. 15-19.

[*Konnecke*, 2015] Konnecke, M., et al. The NeXus data format. (2015). J. Appl. Cryst. 48, 301-305.

[Knüpfer, 2012] Knüpfer, A., et al. "Score-P: A Joint Performance Measurement Run-Time Infrastructure for Periscope, Scalasca, TAU, and Vampir." In Brunst, H., Müller, M., Nagel, W., Resch, M. (eds) Tools for High Performance Computing 2011. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-31476-6_7

[Kougkas, 2018] Kougkas, A., H. Devarajan, and X. H. Sun. "Hermes: A Heterogeneous-Aware Multi-tiered Distributed I/O Buffering System." In Proceedings of the 27th International Symposium on High-Performance Parallel and Distributed Computing, pp. 219-230. DOI: 10.1145/3208040.3208059

[Kougkas, 2019] Kougkas, A., H. Devarajan, J. Lofstead, and X. H. Sun. "LABIOS: A Distributed Label-Based I/O System." In Proceedings of the 28th International Symposium on High-Performance Parallel and Distributed Computing, pp. 13-24. DOI: 10.1145/3307681.3325405

[Koziol, 2014] Koziol, Quincey. "Damsel: A Data Model Storage Library for Exascale Science." No. DOE-HDFGROUP-04948. The HDF Group, Champaign, IL.

[Kreps, 2011] Kreps, J., N. Narkhede, and J. Rao, J." Kafka: A Distributed Messaging System for Log Processing." In Proceedings of the NetDB, Vol. 11, pp. 1-7. available online.

[Kube, 2021] Kube, R., R, M. Churchill, C. S. Chang, J. Choi, R. Wang, S., Klasky, et al. "Near Real-Time Streaming Analysis of Big Fusion Data." Plasma Physics and Controlled Fusion.

[Kurtzer, 2017] Kurtzer, G. M., V. Sochat, and M. W. Bauer. "Singularity: Scientific Containers for Mobility of Compute." *PLOS One 12*(5), e0177459. DOI: 10.1371/journal.pone.0177459

[Latham, 2004] Latham, R., N. Miller, R. Ross, and P. Carns. "A Next-Generation Parallel File System for Linux Cluster. LinuxWorld Mag., *2* (ANL/MCS/JA-48544).

[Lebo, 2013] Lebo, Timothy, Satya Sahoo, and Deborah McGuinness, eds. "PROV-O: The PROV Ontology." W3C Recommendation. http://www.w3.org/TR/2013/REC-prov-o-20130430/
[Li, 2003] Li, Jianwei, Wei-keng Liao, Alok Choudhary, Robert Ross, Rajeev Thakur, William Gropp, Rob Latham, Andrew Siegel, Brad Gallagher, and Michael Zingale. "Parallel netCDF: A Scientific High-Performance I/O Interface." In Proceedings of ACM/IEEE conference on Supercomputing, pp. 39.

[Li, 2019] Li, Tonglin, Quincey Koziol, Houjun Tang, Jialin Liu, and Suren Byna. "I/O Performance Analysis of Science Applications Using HDF5 File-level Provenance." Cray User Group (CUG) 2019.

[Liran, 2018] Joakim, Liran zvibel. "On Using D to Create the World's Fastest f=File System. available online.

[Liu, 2014] Liu, Qing, Jeremy Logan, Yuan Tian, Hasan Abbasi, Norbert Podhorszki, Jong Youl Choi, Scott Klasky, et al. "Hello ADIOS: The Challenges and Lessons of Developing Leadership Class I/O Frameworks." Concurrency and Computation: Practice and Experience 26, no. 7, pp. 1453-1473.

[LKF, 2014] Linux Kernel Foundation. Documentation, L. K. Direct Access for files. available online.

[LMT, 2022] Lustre Monitoring Tool, https://github.com/llnl/lmt/wiki, accessed April 2022.

[Lockwood, 2018] Lockwood, Glenn K., Shane Snyder, Teng Wang, Suren Byna, Philip Carns, and Nicholas J. Wright. "A Year in the Life of a Parallel File System." In Proceedings of the International Conference for High Performance Computing, Networking, Storage, and Analysis (SC '18).

[Lofstead, 2008] Lofstead, J. F., S. Klasky, K. Schwan, N. Podhorszki, and C. Jin. "Flexible IO and Integration for Scientific Codes through the Adaptable IO System (ADIOS)." In Proceedings of the 6th international workshop on Challenges of Large Applications in Distributed Environments, pp. 15-24.

[Lofstead, 2016] Lofstead, J., I. Jimenez, C. Maltzahn, Q. Koziol, J. Bent, and E. Barton. "DAOS and Friends: A Proposal for an Exascale Storage System." In SC'16: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 585-596. IEEE. DOI: 10.1109/SC.2016.49

[Logan, 2012] Logan, J., et al. "Understanding I/O Performance Using I/O Skeletal Applications." In Kaklamanis, C., Papatheodorou, T., Spirakis, P.G. (eds), Euro-Par 2012 Parallel Processing. Lecture Notes in Computer Science, vol. 7484. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-32820-6_10.

[Logan, 2021] Logan, L., J. Lofstead, S. Levy, P. Widener, X. H. Sun, and . Kougkas, A. "pMEMCPY:A Simple, Lightweight, and Portable I/O Library for Storing Data in Persistent Memory." In 2021 IEEE International Conference on Cluster Computing (CLUSTER), pp. 664-670. DOI: 10.1109/Cluster48925.2021.00098

[Lu, 2009] Lu, J., Du, B., Zhu, Y., & Li, D. (2009, May). MADFS: the mobile agent-based distributed network file system. In *2009 WRI Global Congress on Intelligent Systems* (Vol. 1, pp. 68-74). IEEE. DOI: 10.1109/GCIS.2009.208

[Lu, 2020] Lu, Sidi, Bing Luo, Tirthak Patel, Yongtao Yao, Devesh Tiwari, and Weisong Shi. "Making Disk Failure

Predictions SMARTer!." In 18th USENIX Conference on File and Storage Technologies (FAST 20), pp. 151-167. 2020.

[Luk, 2005] Luk, Chi-Keung, Robert Cohn, Robert Muth, Harish Patil, Artur Klauser, Geoff Lowney, Steven Wallace, Vijay Janapa Reddi, and Kim Hazelwood. "Pin: Building Customized Program Analysis Tools with Dynamic Instrumentation." SIGPLAN Not. 40, 6, pp. 190–200. DOI: https://doi.org/10.1145/1064978.1065034

[Ma, 2016] Ma, Shiqing, Xiangyu Zhang, and Dongyan Xu. "Protracer: Towards Practical Provenance Tracing by Alternating Between Logging and Tainting." In NDSS, vol. 2, p. 4.

[Mackey, 2009] Mackey, G., S. Sehrish, and J. Wang. "Improving Metadata Management for Small Files in HDFS." In 2009 IEEE international conference on cluster computing and workshops, pp. 1-4. DOI: 10.1109/CLUSTR.2009.5289133

[Macko, 2011] Macko, Peter, and Margo I. Seltzer. "Provenance Map oObiter: Interactive Exploration of Large Provenance Graphs." TaPP 2011, pp. 1-6.

[Macko, 2012] Macko, Peter, and Margo I. Seltzer. "A General-Purpose Provenance Library." In TaPP.

[Madireddy, 2018] Madireddy, Sandeep, Prasanna Balaprakash, Philip Carns, Robert Latham, Robert Ross, Shane Snyder, and Stefan M Wild. "Machine Learning Based Parallel I/O Predictive Modeling: A Case Study on Lustre File Systems." In Proceedings of the International Conference on High Performance Computing, 2018.

[Mahgoub, 2020] Mahgoub, Ashraf, Alexander M. Medoff, Rakesh Kumar, Subrata Mitra, Ana Klimovic, Somali Chaterji, and Saurabh Bagchi. "OPTIMUSCLOUD: Heterogeneous Configuration Optimization for Distributed Databases in the Cloud." In Proceedings of 2020 USENIX Annual Technical Conference, pp. 189-203.

[MASplus] https://www.mdsplus.org/index.php/Introduction. Accessed June 2022.

[Mathur, 2007] Mathur, A., M. Cao, S. Bhattacharya, A., Dilger, A. Tomas, and L. Vivier, L. "The New ext4 Filesystem: Current Status and Future Plans." In Proceedings of the Linux symposium, Vol. 2, pp. 21-33.

[Mayhew, 2003] Mayhew, D., and V. Krishnan. "PCI Express and Advanced Switching: Evolutionary Path to Building Next Generation Interconnects." In Proceedings of the 11th Symposium on High Performance Interconnects, pp. 21-29. IEEE. DOI: 10.1109/CONECT.2003.1231473

[McKinney, 2011] McKinney, W. "pandas: A Foundational Python Library for Data Analysis and Statistics." Python for High Performance and Scientific Computing, 14(9), pp. 1-9. available online.

[Mcluckie, 2014] Mcluckie, C. "Containers, VMs, Kubernetes and Vmware." Blog, available online.

[Miles, 2007] Miles, S., P. Groth, M. Branco, and L. Moreau. "The Requirements of Using pPovenance in e-Science Experiments." Journal of Grid Computing 5(1), pp. 1-25.

[Miryeong, 2022] Kwon, Miryeong, et al. "Hardware/Software Co-Programmable Framework for Computational SSDs to Accelerate Deep Learning Service on Large-Scale Graphs." 20th USENIX Conference on File and Storage Technologies (FAST 22).

[Missier, 2013a] Missier, Paolo, Khalid Belhajjame, and James Cheney. "The W3C PROV Family of Specifications for Modeling Provenance Metadata." In Proceedings of the 16th International Conference on Extending Database Technology, pp. 773-776.

[Missier, 2013b] Missier, Paolo, Saumen Dey, Khalid Belhajjame, Víctor Cuevas-Vicenttín, and Bertram Ludäscher. "D-PROV: Extending the PROV Provenance Model with Workflow Structure." In 5th USENIX Workshop on the Theory and Practice of Provenance (TaPP 13).

[MongoDB, 2009] Apache Software Foundation. "Arrow. A Cross-Language Development Platform for In-Memory Data." https://arrow.apache.org

[Moreau, 2008] Moreau, Luc, Juliana Freire, Joe Futrelle, Robert E. McGrath, Jim Myers, and Patrick Paulson. "The Open Provenance Model: An Overview." In International Provenance and Annotation workshop, pp. 323-326. Springer, Berlin, Heidelberg, 2008.

[Moreau, 2013] Moreau, Luc, and Paolo Missier, eds. "PROV-DM: The PROV Data Model". 30 April 2013, W3C Recommendation. http://www.w3.org/TR/2013/REC-prov-dm-20130430/

[Morozov, 2016] Morozov, D., and Z. Lukic, "Master of Puppets: Cooperative Multitasking for In Situ Processing," ser. HPDC '16, pp. 285–288.

[Mu, 2020] Mu, Jingqing, Jerome Soumagne, Suren Byna, Quincey Koziol, Houjun Tang, and Richard Warren, "Interfacing HDF5 with a Scalable Object-centric Storage System on Hierarchical Storage." Journal of Concurrency and Computation: Practice and Experience.

[Murugan, 2022] Murugan, M., S. Bhattacharya, D., et al. "ProSPECT: Proactive Storage Using Provenance for Efficient Compute and Tiering." Trans Indian Natl. Acad. Eng. 7, 219–234. https://doi.org/10.1007/s41403-021-00261-8

[Neisse, 2017] Neisse, Ricardo, Gary Steri, and Igor Nai-Fovino. "A Blockchain-Based Approach for Data Accountability and Provenance Tracking." In Proceedings of the 12th international conference on Availability, Reliability and Security, pp. 1-10 .

[NetCompute, 2018] *NetCompute'18: Proceedings of the 2018 Morning Workshop on In-Network Computing.* Association for Computing Machinery, New York.

[Netto, 2019] Netto, Marco A. S., Rodrigo N. Calheiros, Eduardo R. Rodrigues, Renato L. F. Cunha, and Rajkumar Buyya. "HPC Cloud for Scientific and Business Applications: Taxonomy, Vision, and Research Challenges." ACM Computing Surveys 51, no. 1, no. 8, pp. 1-29.

[Niazi, 2017] Niazi, Salman, Mahmoud Ismail, Seif Haridi, Jim Dowling, Steffen Grohsschmiedt, and Mikael Ronström. "HopsFS: Scaling Hierarchical File System Metadata Using NewSQL Databases." In 15th USENIX Conference on File and Storage Technologies (FAST 17), pp. 89-104.

[Nicolae, 2020] Nicolae, Bogdan. "DataStates: Towards Lightweight Data Models for Deep Learning.." SMC 2020, pp. 117-129.

[NVIDIA, 2022] "NVIDIA Profiler User's Guide." https://docs.nvidia.com/cuda/profiler-users-guide/index.html, accessed April 2022.

[OpenFaaS] OpenFaaS. https://github.com/openfaas/faas. Accessed June 2022.

[Parquet] https://parquet.apache.org. Accessed June 2022.

[Pasquier, 2018] Pasquier, Thomas, Xueyuan Han, Thomas Moyer, Adam Bates, Olivier Hermant, David Eyers, Jean Bacon, and Margo Seltzer. “Runtime Analysis of Whole-System Provenance.” In Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, pp. 1601-1616.

[Patel, 2019] Patel, Tirthak, Suren Byna, Glenn K. Lockwood, and Devesh Tiwari. “Revisiting I/O Behavior in Large-Scale Storage Systems: The Expected and the Unexpected.” In Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 1-13.

[Patel, 2020] Patel, Tirthak, Suren Byna, Glenn K. Lockwood, Nicholas J. Wright, Philip Carns, Robert Ross, and Devesh Tiwari. “Uncovering Access, Reuse, and Sharing Characteristics of I/O-Intensive Files on Large-Scale Production HPC Systems.” In 18th USENIX Conference on File and Storage Technologies (FAST 20), pp. 91-101. 2020.

[Patil, 2011] Patil, Swapnil, and Garth Gibson. “Scale and Concurrency of GIGA+: File System Directories with Millions of Files.” In 9th USENIX Conference on File and Storage Technologies (FAST 11).

[Peng, 2014] Peng, I. B., S. Markidis, R. Gioiosa, G. Kestor, and E. Laure. “MPI Streams for HPC Applications,” *CoRR*, vol. abs/1708.01306, 2017.

[Perez, 2007] Pérez, F., and B. E. Granger. “IPython: A System for Interactive Scientific Computing.” Comput. Sci. Eng. DOI: 10.1109/MCSE.2007.53

[Pitchumani, 2020] Pitchumani Rekha, and Yang-Suk Kee. “Hybrid Data Reliability for Emerging Key-Value Storage Devices.” Proc. of 18th USENIX Conference on File and Storage Technologies (FAST ‘20).

[Pouchard, 2020] Pouchard, Line, Pavol Juhas, Gilchan Park, Huub Van Dam, Stuart I. Campbell, Eli Stavitski, Simon Billinge, and Christopher J. Wright. “Provenance Infrastructure for Multi-modal X-ray Experiments and Reproducible Analysis.” In Handbook On Big Data And Machine Learning In the Physical Sciences: Volume 2. Advanced Analysis Solutions for Leading Experimental Techniques, pp. 307-331.

[Prowell, 2021] Prowell, Stacy, David Manz, Candace Culhane, Sheikh Ghafoor, Martine Kalke, Kate Keahey, Celeste Matarazzo, Chris Oehmen, Sean Peisert, and Ali Pinar. Position Papers for the ASCR Workshop on Cybersecurity and Privacy for Scientific Computing Ecosystems. USDOE Office of Science (SC)(United States), 2021.

[Ravi, 2020] Ravi, J., S. Byna, and Q. Koziol. “GPU Direct I/O with HDF5.” In *2020* IEEE/ACM Fifth International Parallel Data Systems Workshop (PDSW), pp. 28-33). DOI: 10.1109/PDSW51947.2020.00010

[Reddi, 2020] Reddi, V. J.*, et al.,* “MLPerf Inference Benchmark,” 2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA), pp. 446-459, doi: 10.1109/ISCA45697.2020.00045.

[Reddy, 2006] Muniswamy-Reddy, Kiran-Kumar, David A. Holland, Uri Braun, and Margo Seltzer. “Provenance-Aware Storage Systems.” In Proceedings of USENIX Annual Technical Conference (USENIX ATC).

[Reddy, 2009] Muniswamy-Reddy, Kiran-Kumar, Uri Braun, David A. Holland, Peter Macko, Diana Maclean, Daniel Margo, Margo Seltzer, and Robin Smogor. “Layering in Provenance Systems.” In Proceedings of USENIX Annual Technical Conference (USENIX ATC).

[Ren, 2014] Ren, Kai, Qing Zheng, Swapnil Patil, and Garth Gibson. “IndexFS: Scaling File System Metadata Performance with Stateless Caching and Bulk Insertion.” In SC’14: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 237-248. IEEE, 2014.

[Rew, 1990] Rew, R., and G. Davis. "NetCDF: An Interface for Scientific Data Access." In Computer Graphics and Applications *10(4)*, 10-1109. DOI: 10.1109/38.56302

[Rew, 2006] Rew, R., E. Hartnett, and J. Caron, J. "NetCDF-4: Software Implementing an Enhanced Data Model for the Geosciences." In 22nd International Conference on Interactive Information Processing Systems for Meteorology, Oceanograph, and Hydrology, Vol. 6. available online.

[Rio, 2007] Rio, Nicholas Del, and Paulo Pinheiro da Silva. "Probe-it! Visualization Support for Provenance." In International Symposium on Visual Computing, pp. 732-741. Springer, Berlin, Heidelberg.

[Rodeh, 2003] Rodeh, O., and A. Teperman, A. "zFS -- a Scalable Distributed File System Using Object Disks." In Proceedings of the 20th IEEE/11th NASA Goddard Conference on Mass Storage Systems and Technologies (MSST 2003), pp. 207-218. DOI: 10.1109/MASS.2003.1194858

[ROOT] "ROOT: Analyzing Petabytes of Data, Scientifically." https://root.cern. Accessed June 2022.

[Ross, 2020] Ross, Robert B., George Amvrosiadis, Philip Carns, Charles D. Cranor, Matthieu Dorier, Kevin Harms, Greg Ganger, et al. "Mochi: Composing Data Services for High-Performance Computing Environments." Journal of Computer Science and Technology 35, no. 1, pp. 121-144.

[Roy, 2021] Roy, Rohan Basu, Tirthak Patel, and Devesh Tiwari. "Characterizing and Mitigating the I/O Scalability Challenges for Serverless Applications." In 2021 IEEE International Symposium on Workload Characterization (IISWC), pp. 74-86.

[Roy, 2022] Roy, Rohan Basu, Tirthak Patel, Vijay Gadepally, and Devesh Tiwari. "Mashup: Making Serverless Computing Useful for HPC Workflows via Hybrid Execution." In Proceedings of the 27th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPoPP '22). Association for Computing Machinery, New York, pp. 46–60. DOI: https://doi.org/10.1145/3503221.3508407

[Ruan, 2019] Ruan, Zhenyuan, Tong He, and Jason Cong. "INSIDER: Designing In-Storage Computing System for Emerging High-Performance Drive." In Proceedings of the 2019 USENIX Conference on Usenix Annual Technical Conference (USENIX ATC '19). USENIX Association, USA, pp. 379–394.

[Sanner, 1999] Sanner, M. F. "Python: A Programming Language for Software Integration and Development." J Mol Graph Model, 17(1), pp. 57-61.

[sciunit] https://sciunit.run. Accessed June 2022.

[Shende, 2006] Shende, Sameer S., and Allen D. Malony. "The Tau Parallel Performance System." The International Journal of High Performance Computing Applications, 20, no. 2, pp. 287–311, doi: 10.1177/1094342006064482.

[Shin, 2019] Shin, Woong, Christopher D. Brumgard, Bing Xie, Sudharshan S. Vazhkudai, Devarshi Ghoshal, Sarp Oral, and Lavanya Ramakrishnan. "Data Jockey: Automatic Data Management for HPC Multi-Tiered Storage Systems." In 2019 IEEE International Parallel and Distributed Processing Symposium (IPDPS), pp. 511-522.

[Simmhan, 2006] Simmhan, Yogesh L., Beth Plale, Dennis Gannon, and Suresh Marru. "A Framework for Collecting Provenance in Data-Centric Scientific Workflows." In 2006 IEEE International Conference on Web Services (ICWS).

[SNIA, 2021] SNIA. "Computational Storage Architecture and Programming Model, Version 0.8 Revision 0."

Working Draft, June.

[SNIA, 2022] Storage Networking Industry Association (SNIA) "IOTTA Repository" http://iotta.snia.org, retrieved April 2022.

[Souza, 2019a] Souza, Renan, et al. "Provenance Data in the Machine Learning Lifecycle in Computational Science and Engineering." Workflows in Support of Large-Scale Science (WORKS) workshop co-located with SC19.

[Souza, 2019b] Souza, R., L. Azevedo, R. Thiago, et al., "Efficient Runtime Capture of Multiworkflow Data Using Provenance." In IEEE eScience.

[Stansberry, 2019] Stansberry, Dale, Suhas Somnath, Jessica Breet, Gregory Shutt, and Mallikarjun Shankar. "DataFed: Towards Reproducible Research via Federated Data Management." In 2019 International Conference on Computational Science and Computational Intelligence (CSCI), pp. 1312-1317. IEEE.

[Suchyta, 2022] Suchyta, Eric, Scott Klasky, Norbert Podhorszki, Matthew Wolf, Abolaji Adesoji, C. S. Chang, Jong Choi, et al. "The Exascale Framework for High Fidelity coupled Simulations (EFFIS): Enabling Whole Device Modeling in Fusion Science." The International Journal of High Performance Computing Applications 36, no. 1, pp. 106-128.

[Sultana, 2021] Sultana, Nik, John Sonchack, Hans Giesen, Isaac Pedisich, Zhaoyang Han, Nishanth Shyamkumar, Shivani Burad, André DeHon, and Boon Thau Loo. "Flightplan: Dataplane Disaggregation and Placement for P4 Programs." In 18th USENIX Symposium on Networked Systems Design and Implementation (NSDI 21), pp. 571-592.

[Suriarachchi, 2015] Suriarachchi, Isuru, Quan Zhou, and Beth Plale. "Komadu: A Capture and Visualization System for Scientific Data Provenance." Journal of Open Research Software 3, no. 1.

[Tang, 2017] Tang,  Kun, Ping Huang, Xubin He, Tao Lu, Sudharshan S. Vazhkudai, and Devesh Tiwari. "Toward Managing HPC Burst Buffers Effectively: Draining Strategy to Regulate Bursty I/O Behavior." In 2017 IEEE 25th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS), pp. 87-98.

[Tang, 2018] Houjun Tang, Suren Byna, Francois Tessier, Teng Wang, Bin Dong, Jingqing Mu, Quincey Koziol, Jerome Soumagne, Venkatram Vishwanath, Jialin Liu, and Richard Warren, "Toward Scalable and Asynchronous Object-centric Data Management for HPC", 18th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid).

[Tensorboard, 2022] TensorBoard: TensorFlow's visualization toolkit, https://www.tensorflow.org/tensorboard, accessed April 2022.

[Thakur, 1999] Thakur, Rajeev, William Gropp, and Ewing Lusk. "On Implementing MPI-I/O Portably and with High Performance." In Proceedings of the sixth workshop on I/O in Parallel and Distributed Systems.

[Thavasimani, 2019] Thavasimani, Priyaa, Jacek Cała, and Paolo Missier. "Why-diff: Exploiting Provenance to Understand Outcome Differences from Non-identical Reproduced Workflows." IEEE Access 7, pp. 34973-34990.

[Tiwari, 2013] Tiwari, Devesh, Simona Boboila, Sudharshan Vazhkudai, Youngjae Kim, Xiaosong Ma, Peter Desnoyers, and Yan Solihin. "Active Flash: Towards Energy-Efficient, In-Situ Data Analytics on Extreme-Scale Machines." In 11th USENIX Conference on File and Storage Technologies (FAST 13), pp. 119-132.

[Tonglin, 2021] Tonglin, Li, Suren Byna, Quincey Koziol, Houjun Tang, Jean Luca Bez, and Qiao Kang. "h5bench: HDF5 I/O Kernel Suite for Exercising HPC I/O Patterns." Proceedings of Cray User Group Meeting, CUG.

[Tseng, 2019] Tseng,S. M., B. Nicolae, G. Bosilca, E. Jeannot, A. Chandramowlishwaran, and F. Cappello, Towards Portable Online Prediction of Network Utilization using MPI-level Monitoring, EuroPar.- ?Cited?

[Tseng, 2021Tseng, S.-M., B. Nicolae, F. Cappello, A. Chandramowlishwaran, Demystifying Asynchronous I/O Interference in HPC Applications, accepted in International Journal on High Performance Computing Application, JHPCA, SAGE, 2021.?Cited?

[Turnbull, 2014] Turnbull, J. The Docker Book: Containerization Is the New Virtualization. James Turnbull.

[UnifyFS, 2017] UnifyFS: A Distributed Burst Buffer File System, https://github.com/llnl/unifyfs

[Vef, 2018] Vef, M. A., N. Moti, T. Süß, T. Tocci, R. Nou, A. Miranda, ... and A. Brinkmann. "Gekkofs -- a Temporary Distributed File System for HPC Applications." In 2018 IEEE International Conference on Cluster Computing (CLUSTER), pp. 319-324. DOI: 10.1109/CLUSTER.2018.00049

[Venabels, 2000] Venables, W., Venables, W. N., and B. D. Ripley. S Programming. Springer Science & Business Media.

[Wang, 1993] Wang, R. Y., and T. E. Anderson. "xFS: A Wide Area Mass Storage File System." In Proceedings of IEEE 4th Workshop on Workstation Operating Systems. WWOS-III, pp. 71-78. DOI: 10.1109/WWOS.1993.348169

[T. Wang, 2018] Wang, Teng, Suren Byna, Bin Dong, and Houjun Tang, "UniviStor: Integrated Hierarchical and Distributed Storage for HPC." IEEE Cluster 2018, Belfast.

[W. Wang, 2018] Wang, W., and S. Diestelhorst. "Quantify the Performance Overheads of PMDK." In Proceedings of the International Symposium on Memory Systems, pp. 50-52. DOI: 10.1145/3240302.3240423

[C. Wang, 2020] Wang, C.,J. Sun, M. Snir, K. Mohror, and E. Gonsiorowski, "Recorder 2.0: Efficient Parallel I/O Tracing and Analysis." 2020 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), pp. 1-8, doi: 10.1109/IPDPSW50202.2020.00176.

[J. Wang, 2020] Wang, J., K. U. O. Tzu-Yang, L. Li, and A. Zeller. "Assessing and Restoring Reproducibility of Jupyter Notebooks." In 2020 35th IEEE/ACM international conference on Automated Software Engineering (ASE), pp. 138-149. available online.

[R. Wang, 2020] Wang, R., et al. "Processing Full-Scale Square Kilometre Array Data on the Summit Supercomputer," SC20: International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 1-12, doi: 10.1109/SC41405.2020.00006.

[Weil, 2006] Weil, S. A., S. A. Brandt, E. L. Miller, D. D. Long, and C. Maltzahn. "Ceph: A Scalable, High-Performance Distributed File System." In Proceedings of the 7th symposium on Operating Systems Design and Implementation, pp. 307-320. available online.

[wholetale] https://wholetale.org. Accessed June 2022.

[Wilkinson, 2016] Wilkinson, M. D., M. Dumontier, I. J. Aalbersberg, G. Appleton, M. Axton, A. Baak,., ... and B. Mons. "The FAIR Guiding Principles for Scientific Data Management and Stewardship." Scientific Data, 3(1), pp. 1-9. DOI:

10.1038/sdata.2016.18

[Wilkinson, 2019] Wilkinson, M. D., et al., "Addendum: The FAIR Guiding Principles for Scientific Data Management and Stewardship." Scientific Data, 6, no. 1, Art. no. 1, doi: 10.1038/s41597-019-0009-6.

[Wozniak, 2021] Wozniak, Justin M., Zhengchun Liu, Rafael Vescovi, Ryan Chard, Bogdan Nicolae, and Ian T. Foster. "Braid-DB: Toward AI-Driven Science with Machine Learning Provenance." SMC 2021: 247-261.

[Wu, 2009] Wu, H., Z. Shang, and K. Wolter. "Trak: A Testing Tool for Studying the Reliability of Data Delivery in Apache Kafka. In 2019 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW), pp. 394-397. DOI: 10.1109/ISSREW.2019.00101

[Xie, 2021] Xie, B., et al., "Interpreting Write Performance of Supercomputer I/O Systems with Regression Models." 2021 IEEE International Parallel and Distributed Processing Symposium (IPDPS), 2021, pp. 557-566, doi: 10.1109/IPDPS49936.2021.00064.

[Yang, 2017] Yang, Z., Harris, J. R., Walker, B., Verkamp, D., Liu, C., Chang, C., ... & Paul, L. E. (2017, December). SPDK: A development kit to build high performance storage applications. In *2017 IEEE International Conference on Cloud Computing Technology and Science* (*CloudCom*) (pp. 154-161). IEEE. DOI: 10.1109/CloudCom.2017.14. ?Cited?

[Zarr, 2021] Zarr Storage Specification 2.0 Community Standard. Open Geospatial Consortium. Submitted June 21, 2021.

[Zheng, 2015] Zheng, Qing, Kai Ren, Garth Gibson, Bradley W. Settlemyer, and Gary Grider. "DeltaFS: Exascale File Systems Scale Better without Dedicated Servers." In Proceedings of the 10th Parallel Data Storage Workshop, pp. 1-6.

[Zheng, 2018] Zheng, Qubg, Charles D. Cranor, Danhao Guo, Gregory R. Ganger, George Amvrosiadis, Garth Gibson, Bradley W. Settlemyer, Gary Grider, and Fan Guo. "Scaling Embedded In-Situ Indexing with DeltaFS." In Proceedings of International Conference for High Performance Computing, Networking, Storage, and Analysis (SC 2018), 2018.

# 9. Acknowledgments

The organizers wish to thank a number of people for their assistance with the workshop. The organizers thank Hal Finkel and Margaret Lentz for facilitating the meeting. Additionally, the organizers thank Oak Ridge Institute for Science and Education and Deneise Terry for managing the registration and logistics of the workshop series, and Gail Pieper and Mary Gines for their editing and layout assistance.