EuroHPC-01-2019

IO-SEA

IO - Software for Exascale Architectures

Grant Agreement Number: 955811

D5.4

Demonstrations of data-centric workflows using DASI for in-situ processing and visualisation

Final

Version: 1.0

Author(s): J. Wong (ECMWF), M. Cakircali (ECMWF) and J. Hawkes (ECMWF)

Contributor(s): R. Furmanek (TU Ostrava)

Date: February 26, 2024

Project and Deliverable Information Sheet

<table>
<tr><td rowspan="7">IO-SEA Project</td><td colspan="2">Project Ref. №: 955811</td></tr>
<tr><td colspan="2">Project Title: IO - Software for Exascale Architectures</td></tr>
<tr><td colspan="2">Project Web Site: http://www.iosea-project.eu</td></tr>
<tr><td colspan="2">Deliverable ID: D5.4</td></tr>
<tr><td colspan="2">Deliverable Nature: Report</td></tr>
<tr><td rowspan="2">Deliverable Level:
PU</td><td>Contractual Date of Delivery:
29 / 02 / 2024</td></tr>
<tr><td>Actual Date of Delivery:
29 / 02 / 2024</td></tr>
<tr><td colspan="2">EC Project Officer: René Chatwill</td></tr>
</table>

* - The dissemination levels are indicated as follows: PU = Public, fully open, e.g. web; CO = Confidential, restricted under conditions set out in Model Grant Agreement; CI = Classified, information as referred to in Commission Decision 2001/844/EC.

Document Control Sheet

<table>
<tr><td rowspan="5">Document</td><td colspan="2">Title: Demonstrations of data-centric workflows using DASI for in-situ processing and visualisation</td></tr>
<tr><td colspan="2">ID: D5.4</td></tr>
<tr><td>Version: 1.0</td><td>Status: Final</td></tr>
<tr><td colspan="2">Available at: http://www.iosea-project.eu</td></tr>
<tr><td colspan="2">Software Tool: Microsoft Word</td></tr>
<tr><td rowspan="4">Authorship</td><td>Written by:</td><td>J. Wong (ECMWF), M. Cakircali (ECMWF) and J. Hawkes (ECMWF)</td></tr>
<tr><td>Contributors:</td><td>R. Furmanek (TU Ostrava)</td></tr>
<tr><td>Reviewed by:</td><td>S. Happ (ParTec), J. Novacek (CEITEC)</td></tr>
<tr><td>Approved by:</td><td>Executive Board</td></tr>
</table>

Document Status Sheet

Version	Date	Status	Comments
0.1	29/January/2024	Draft	
0.2	09/February/2024	Ready for internal review	
0.3	19/February/2024	Post 1st review edits complete	
0.4	22/February/2024	Additions on CryoEM usecase	
0.5	26/February/2024	Post 2nd review edits complete	
1.0	26/February/2024	Final	

Document Keywords

Keywords:	IO-SEA, HPC, Exascale, Software

Table of Contents

List of Figures

List of Code Boxes

List of Tables

Executive Summary

In this report we present a description of the integration of DASI into the workflow manager through the DASI ephemeral service and present the final workflow adaptions for the weather forecasting benchmark to use DASI for data retrieval in the post-processing. Adaptions to the workflow definition file to use the DASI ephemeral service and a demonstration run with the service are also presented for the weather forecasting and cryo-electron microscopy usecases.

1 Introduction

The Data Access and Storage Interface (DASI), described in IO-SEA Deliverable 5.1 [1], aims to simplify the management of data through the use of semantic metadata for data archival and retrieval. The use of metadata to uniquely identify data removes possible dependencies on the structure of the underlying storage backend and allows applications to seamlessly move between different storage technologies. This abstraction removes the burden on the user for managing the storage of the data and enables applications to utilise the most performant backend storage solutions, or explore new technologies, without any modifications to the application.

In addition to application interfaces, one of the key concepts of the IO-SEA project is the use of ephemeral services. Initial results from running the usecases with the smart burst buffer ephemeral service to reduce data retrieval from the backend storage system have shown positive results, particularly in the case of a heavily loaded filesystem [2]. The DASI ephemeral service, described in more detail in IO-SEA Deliverable 2.3 [3], enables workflows integrated with DASI to make use of separate network file system ephemeral services for each root specified in the DASI configuration file.

In this deliverable we detail the final workflow adaption for the weather forecasting benchmark to use DASI for data retrieval during post-processing. The initial work on the adaption of the weather forecasting usecase to use DASI for data archival is detailed in Deliverable 5.3 [4], as is the complete modification of the cryo-electron microscopy workflow to use DASI. The integration of DASI into the remaining usecases, such as the lattice quantum chromodynamics and the terrestrial system modelling platform, will be discussed in IO-SEA Deliverable 5.6 [5].

We provide a description of the DASI command line interface (CLI) tool used by the workflow manager v1.6 in the creation of DASI ephemeral services and present results from running the fully adapted weather forecasting and cryo-electron microscopy workflows with the DASI ephemeral service.

This document is structured as follows, in Section 2 we describe the DASI ephemeral service and the DASI CLI tool it utilises. Section 0 details the final adaptions of the weather forecasting usecase to use DASI throughout the workflow and the use of the DASI ephemeral service. Section **Erreur ! Source du renvoi introuvable.** describes the adaptions to the cryo-electron microscopy usecase for the DASI ephemeral service and, finally, we conclude in Section 0.

2 DASI Ephemeral Service

DASI is a metadata-driven data store that abstracts data and storage by providing C/C++/Python APIs to user applications. DASI indexes and identifies data based on its scientifically meaningful description. DASI databases are constructed based on rules (blueprints) that are defined in a user schema file. DASI is configurable via user defined yaml files. An example configuration file is shown in Code box 2.1, which specifies the DASI schema file path as well as a list of database locations, referred to as spaces/roots. Users can specify multiple roots (storage spaces) and set different behaviour, such as read-only.

```
---
schema: /path/to/schema/file
spaces:
    - roots:
        - path: /path/to/output/data1
          writeable: true
        - path: /path/to/output/data2
```

Code box 2.1 Example DASI configuration file.

2.1 Command Line Tool: dasi-schema

To enable DASI as an ephemeral service as part of the IO-SEA project, we developed the dasi-schema Command Line Interface (CLI) tool, which is mainly used by the Workflow Manager (WFM). The dasi-schema tool parses DASI schema and configuration files and prints information to the standard output. Users can pass various options to the dasi-schema CLI tool to execute different functions. Those options are shown in Code box 2.2. The options `--root` and `--list` are used by the workflow manager. An example usage that prints root paths (defined in a configuration file) is shown in Code box 2.3.

```
Usage: dasi-schema [options]
Options are:
   --config=string (DASI Configuration file (yaml))
   --query (Prints out the query string)
   --scan (Prints out the keys found in the root paths)
   --root (Prints out the root paths defined in the configuration file.)
   --list (Prints out the data and index paths for a given query.)
```

Code box 2.2 Various options of dasi-schema CLI tool.

```
$ dasi-schema --config=dasi.yaml --root
/path/to/output/data1
/path/to/output/data2
```

Code box 2.3 Example usage of dasi-schema CLI tool that prints root paths.

```
$ dasi-schema --config=dasi.yaml --list param=p,level=1,number=2
/user/abc/root1/0001:20230807:1320/2:9.20230807.data
/user/abc/root1/0001:20230807:1320/2:8.20230807.data
/user/abc/root1/0001:20230807:1320/2:7.20230807.data
```

Code box 2.4 Example usage of dasi-schema CLI tool to list data paths for a given query.

2.2 Integration to Workflow Manager

For each DASI service, the workflow manager creates a network filesystem (NFS) ephemeral service such that it provides a storage backend for each DASI root. A more in-datadepth discussion of the workflow definition file (WDF) for a DASI service and the required attributes for a DASI ephemeral service will be discussed in the IO-SEA Deliverable 2.3 [3]. One required attribute of the DASI service is `dasiconfig` which specifies the path to the DASI configuration file. An example WDF with a DASI service is shown in Code box 2.5. While DASI allows specification of multiple roots, the initial DASI service in the WFM v1.6 will only support a single DASI root per DASI service. This is a result of the current WFM only supporting a single ephemeral service per job step, but multiple ephemeral services per step may be supported in future versions. The WFM identifies the user defined root paths using the dasi-schema CLI tool, as described in Section 2.1.

WFM Orchestrator supports datamovers that instantiate its Flash Accelerators services on data nodes. The data movers in a DASI service could be configured via new `dasi_list` attribute. This attribute would accept values in DASI Query format, which could be used to identify the DASI data object paths that needs to be moved. WFM would use the dasi-schema CLI tool with `--list` option to identify those DASI data object paths, as described in Section 2.1. An example WDF is shown in Code box 2.6.

```
workflow:
  name: My_Workflow
services:
  - name: dasi-service
    type: DASI
    attributes:
      dasiconfig: /path/to/dasi/configuration/file
```

Code box 2.5 Example workflow definition file for DASI service.

```
...
datamovers:
  - name: datamover1
    dasi_list: “type=ensemble,param=p,level=1,number=2”
```

Code box 2.6 Example datamover for DASI service.

3 Weather Forecasting Usecase

In this section we present the final adaption of the weather forecasting usecase to use DASI as well as the modifications to the WDF to use the DASI ephemeral service, described in Section 0.

3.1 DASI Integration into Post-Processing

In Deliverable 5.3 [4], the integration of DASI into ECMWF's middleware I/O library, MultIO, was described. This details the use of DASI for archiving the data produced in the forecast. In order for DASI to be used for reading the model outputs in the downstream post-processing ECMWF's product generation library, called pgen, also needs to be modified.

A new source class, `DASISource`, was added to pgen which uses the DASI C++ interface. The code snippet for defining the `DASISource` class and its retrieve method is shown in Code box 3.1. The method involves converting the inputs into a DASI Query object and using the DASI retrieve method to obtain the data associated with the query.

```
DASISource::DASISource(const eckit::option::CmdArgs &args):
    Source(args), dasi_(dasi::Dasi(eckit::LocalConfiguration())) {
}

size_t DASISource::retrieve(const std::map<std::string, std::string>&
retrieve, eckit::Buffer& field) const {

    // Create DASI Query
    dasi::Query query;
    for (auto k = retrieve.begin(); k != retrieve.end(); ++k) {
        query.set((*k).first, {(*k).second});
    }

    // Call DASI retrieve method, which returns dasi::RetrieveResult
    // object
    auto result(const_cast<dasi::Dasi&>(dasi_).retrieve(query));


    std::unique_ptr<eckit::DataHandle> dh(result.dataHandle());
    eckit::MemoryHandle mh(field);
    return size_t(dh->copyTo(mh));
}
```

Code box 3.1: C++ code that adapts the pgen application to use DASI.

In addition to defining a new source class for DASI, we also need to modify the arguments passed to the pgen executable to use the DASI source. The product generation executable has a command line argument `--source` which specifies the library to use for data retrieval, and we change this from "fdb5" to "dasi".

3.2 DASI Ephemeral Service

To adapt the ECMWF weather forecasting workflow to use the DASI ephemeral service in WFM v1.6, we modify the WDF used for an NFS service to include the `dasiconfig` attribute detailed in Section 2. As the mountpoints for the NFS services for each root are derived from the DASI configuration file, the `mountpoint` attribute is no longer required in the WDF. The resulting WDF is shown in Code box 3.2, where the attributes with values surrounded by an @ symbol are replaced by JUBE [6] with the paths relevant for the benchmark run.

The original benchmark uses ECMWF's FDB[1] for archiving and retrieving data. As the format of the configuration file for the FDB is similar to that of DASI, the `dasiconfig` attribute in the WDF is set to the location of the FDB configuration file. For the `storagesize` we request 50GiB, compatible with the volume of data archived in the benchmark runs discussed in Deliverable 4.1 [2] using the smart burst buffer service.

```
workflow:
   name: ecmwf-dasi-workflow

services:
   - name: ecmwf-dasi
     type: DASI
     attributes:
        namespace: @NAMESPACE@
        dasiconfig: @FDB_CONFIG@
        storagesize: 50GiB
        datanodes: 1
        location: dp-cn

steps:
   - name: multiohammer
     command: "sbatch {{ job_script_path }}"
     services:
        - name: ecmwf-dasi
   - name: pgen
     command: "sbatch {{ job_script_path }}"
     services:
        - name: ecmwf-dasi
```

Code box 3.2: Weather forecasting WDF using DASI ephemeral service.

We can confirm that the DASI service is running by executing `iosea-wf status -A`, the result of which is shown in Code box 3.3 and shows the service has been allocated and is ready for job steps to be run.

```
$ iosea-wf status -A
SERVICE                                          TYPE       STATUS
wong6-ecmwf_hestia_dasi-ecmwf-dasi               DASI       allocated
```

Code box 3.3: Status of DASI ephemeral service.

[1] https://github.com/ecmwf/fdb

The timeline of the workflow with the DASI ephemeral service from the IO-instrumentation dashboard is shown in Figure 3.1, where time is shown on the x-axis and the y-axis labels the different jobs executed in the workflow. The jobs at the start and end of the workflow, top and bottom of the y-axis, respectively, are the workflow session creation and destruction jobs and the jobs in between are the five I/O simulator and seven product generation jobs.

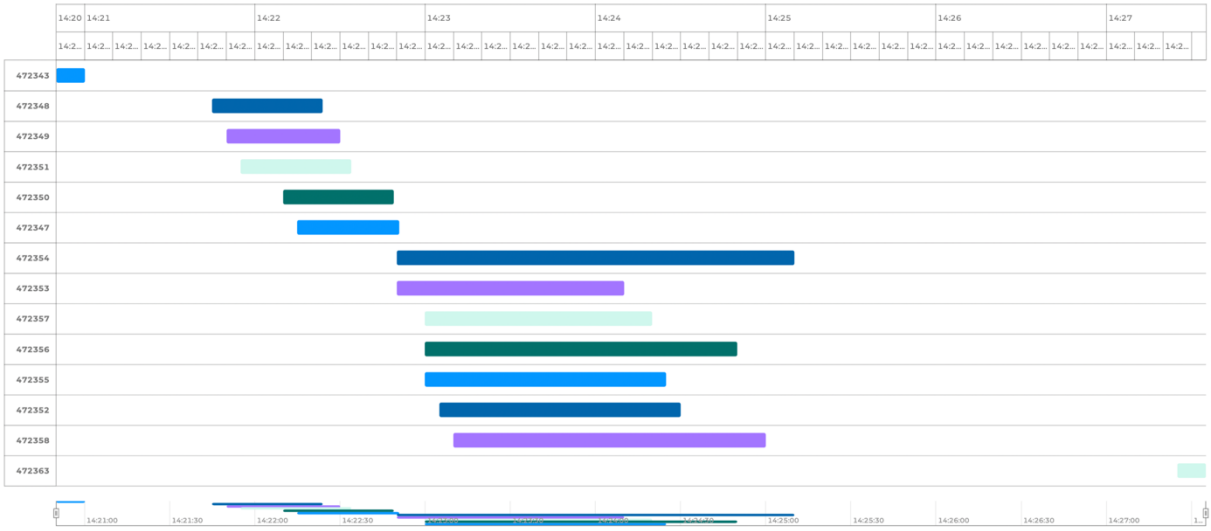

Figure 3.1: Timeline of weather forecasting benchmark workflow with five I/O simulators and seven staggered product generation jobs.

In these initial results using WFM v1.6, the resulting read and write times from a single iteration of the benchmark are presented in Table 3.1, alongside a vanilla run with no ephemeral services, the smart burst buffer (SBB) and Ganesha NFS ephemeral services. The write time for the DASI service is similar to the run with no ephemeral services and NFS service, while the SBB service has a significantly higher write time. The longer write times with the SBB service was discussed in [2] as likely being due to old data nodes on the DEEP system and the fast /afsm filesystem. The read time is lowest using the SBB service as a result of the SBB cache, which means data is no longer being read from the filesystem for the product generation steps. The read time is longer for the NFS and DASI ephemeral services, though this is variable due to varying load on the filesystem during the experiments. For the vanilla and the SBB service runs a reservation was used on the DEEP system so that the experiments were not affected by those of other users. However, this was not the case for the NFS and DASI service runs and therefore the higher read time could be a result of higher loads on the filesystem. In addition, these latter two services both rely on the Ganesha NFS server and accessing the data through this could also be the reason behind the longer read times.

Run	Write Time (s)	Read Time (s)
No ephemeral services	22.88	30.50
SBB service	150.05	17.23
NFS service	24.84	52.73
DASI service	22.49	45.68

Table 3.1: Weather forecasting benchmarking results with ephemeral services.

In the current version of the WFM, only a single ephemeral service can be used by a job step at a time. This means each step can either use the SBB service or the DASI service. From the results in Table 3.1, the weather forecasting benchmark would benefit from the use of the SBB ephemeral service for more efficient access to the model outputs in the product generation jobs in addition to the DASI service for data management. It would therefore be interesting to experiment with using multiple ephemeral services in each step.

4 Cryo Electron Microscopy Usecase

In this section, we unveil the latest adaptation of the Cryo Electron Microscopy usecase, integrating DASI, along with the adjustments to the WDF to incorporate the DASI ephemeral service, as detailed in Section 2.

4.1 DASI Ephemeral Service

In order to integrate the DASI ephemeral service into the Cryo Electron Microscopy workflow within WFM v1.6, we make adjustments to the WDF initially configured for an NFS service in the WFM v1.4.2, which did not support DASI integration. This entails incorporating the `dasiconfig` attribute, as specified in Section 2. Since the mountpoints for the NFS services are now obtained from the DASI configuration file, the `mountpoint` attribute becomes unnecessary in the WDF. The modified WDF is depicted in Code box 4.1, with attributes containing values enclosed by an § symbol being substituted by JUBE with the pertinent paths for the benchmark run.

The original benchmark uses Python scripts for archiving and retrieving data. As the format of the configuration file for the Python scripts is the same to that of DASI, the `dasiconfig` attribute in the WDF is set to the location of the original configuration file for Python. For the `storagesize` we request different sizes, depends on size of input dataset.

```
workflow:
   name: Workflow_CRYOEM

services:
   - name: cryoem-dasi1
     type: DASI
     attributes:
        namespace: §NAMESPACE§
        dasiconfig: §DASI_CONFIG§
        storagesize: §STORAGE_SIZE§
        datanodes: 1
        location: §SLURM_QUEUE§
steps:
   - name: step_1
     command: "§CRYOEM_SUBMITCMD§"
     services:
        - name: cryoem-dasi1
```

Code box 4.1: Cryo-EM WDF using DASI ephemeral service.

Compared to the original deployment of DASI in the Cryo-EM workflow, which used Python scripts for archiving and retrieving data, the current initial operation is very similar except the fact, now are used the implemented commands for DASI. It required an additional step in the workflow, which creates a tarball for the entire dataset, which is archived and then must be extracted when retrieving the data. The data are archived according to the size of the dataset required for processing. The operation for archiving data in DASI is performed only once compared to the retrieve data operation, when it is necessary to extract the files each time. This operation does not have a significant effect on the workflow. This operation can have a

significant effect on the workflow though there is significant scope for improvement which should be investigated.

The behaviour of the process, when the data is to be archived or retrieved is governed by the JUBE manager [6], which is responsible for obtaining additional information. If the dataset exists and has archived data available, it will retrieve the data and start the subsequent processing. If the dataset is not available, it creates a new one with the required data are stored through DASI service.

The timeline depicting the workflow with the DASI ephemeral service from the IO-instrumentation dashboard is illustrated in Figure 4.1. The x-axis represents time, while the y-axis labels the various jobs performed within the workflow. The jobs marked at the beginning and end of the workflow, located at the top and bottom of the y-axis respectively, correspond to the workflow session creation and destruction tasks. The job positioned between them is the Cryo-EM software pipeline job.

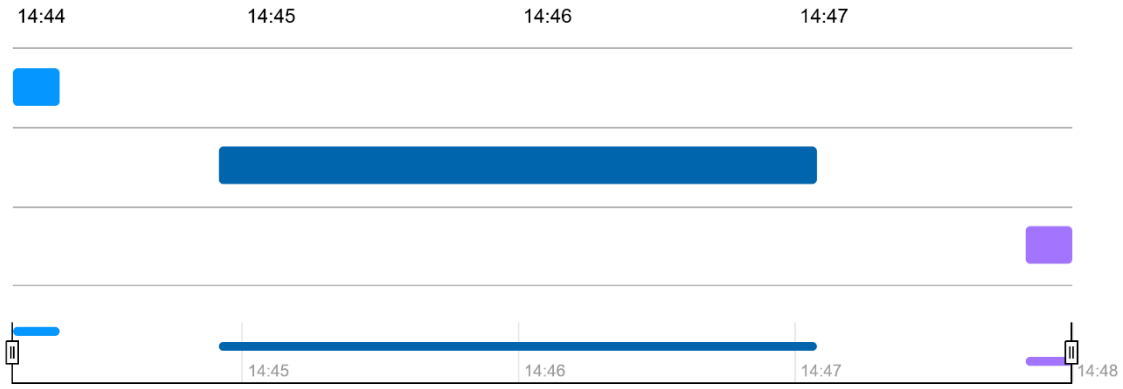

Figure 4.1: Timeline of Cryo-EM benchmark workflow.

In these preliminary findings, Table 4.1 displays the times obtained from a single iteration of the benchmark, both with and without ephemeral services. The SBB service was excluded from the benchmark due to its incompatibility with the results, as previously described in the IOSEA Deliverable 1.4 report [2].

Type	No ephemeral services AVG [STD] (s)	DASI service AVG [STD] (s)
Transfer Data	3.3 [7.0]	0.1 [0.1]
SW MotionCor	5.6 [0.9]	5.0 [0.5]
SW GCTF	2.3 [0.1]	2.3 [0.1]
Entire Pipeline	12.4 [7.7]	8.1 [0.5]

Table 4.1: Cryo-EM workflow benchmarking results with DASI ephemeral service and no ephemeral services.

The results show processing steps within workflow. In the case of using DASI service as NFS, data transfer is 33 times faster than without using ephemeral services and also much more constant which suggests standard deviation 0.1. In contrast, the running of the two software is comparatively the same in both cases. A single image processing run, when using the

ephemeral service, is on average 35% faster and more consistent over time, as indicated by a standard deviation of 0.5.

5 Conclusion

The DASI ephemeral service allows workflows which have been adapted to use DASI to make use of NFS ephemeral services for each root specified in the DASI configuration file. In this report we have detailed the dasi-schema CLI tool developed for the workflow manager to parse the DASI configuration file, which is necessary for the creation of the NFS services. First results from running the DASI ephemeral service in workflow manager v1.6, which only supports a single root in the DASI configuration file, have been presented here for the weather forecasting usecase incorporating the final adaptions of the workflow to use DASI for in-situ processing, and the cryo-electron microscopy usecase for image processing. The integration of DASI into other usecases will be presented in IO-SEA Deliverable 5.6 [5].

List of Acronyms and Abbreviations

A

API An Application Programming Interfaces (API) allows software to communicate with other software which support the same API.

C

CLI Command Line Interface

D

DASI Data Access and Storage Interface developed in Work Package 5.

E

ECMWF European Centre for Medium Range Weather Forecasts

F

FDB ECMWF's semantic data store

I

IO Input/Output is either a noun referring to the action of doing either input and/or output, generally either reading or writing memory, or is an adjective or adverb that describes that the following operation does input and/or output.

J

JUBE The JÜlich Benchmarking Environment is a script based framework to easily create benchmark sets, run those sets on different computer systems and to evaluate the results.

N

NFS Network Filesystem

S

SBB Smart Burst Buffer

U

UUID A universally unique identifier (UUID) is a 128-bit label used for information in computer systems.

W

WDF Workflow Definition File, configuration file for workflow manager specifying ephemeral services and their attributes to the used in a workflow session

WFM The IO-SEA Workflow Manager is responsible for starting ephemeral storage services and running workflow steps in the IO-SEA storage environment.

6 Bibliography

[1] J. Hawkes and O. Iffrig, “First version of the Data Access and Storage Interface (DASI) Implementation,” Technical Report, IO-Software for Exascale Architectures, 2021.

[2] E. Gregory and P. Couvee, “IO-SEA D1.4: First application port and evaluation,” Technical report, IO-Software for Exascale Architectures, 2023.

[3] P. Couvee, S. Happ and M. Rauh, IO-SEA D2.3: Description of the final Ephemeral data Access Environment, Technical report, IO-Software for Exascale Architectures, 2024.

[4] D. Medeiros and S. Markidis, “IO-SEA D5.3: Adaptations of Application Workflows to the DASI,” Technical report, IO-Software for Exascale Architectures, 2023.

[5] D. Medeiros and S. Markidis, IO-SEA D5.6: Final Application workflow adaptations to the DASI, Technical report, IO-Software for Exascale Architectures, 2024.

[6] Julich Supercomputing Centre, “JUBE - Julich Benchmarking Environment,” 2008. [Online]. Available: http://www.fz-juelich.de/jsc/jube.

[7] E. Gregory, IO-SEA D1.5: Final report on applications experience, Technical Report, IO-Software for Exascale Architectures, 2024.