EuroHPC-01-2019

IO-SEA

IO – Software for Exascale Architectures

Grant Agreement Number: 955811

D1.1

Application and co-design input

Final

Version:	1.0
Author(s):	E. B. Gregory (FZJ), P. Couvée (ATOS), M. Golasowski (IT4I)
Contributor(s):	D. Caviedes Voullième (FZJ), J. Hawkes (ECMWF), O. Iffrig (ECMWF), T. Leibovici (CEA), A. Lopez (ATOS), L. Strafella (CEA)
Date:	July 29, 2021

Project and Deliverable Information Sheet

IO-SEA Project	**Project ref. No.:**	955811
	Project Title:	IO – Software for Exascale Architectures
	Project Web Site:	https://www.iosea-project.eu/
	Deliverable ID:	D1.1
	Deliverable Nature:	Report
	Deliverable Level: PU *	**Contractual Date of Delivery:** 31 / July / 2021
		Actual Date of Delivery: 29 / July / 2021
	EC Project Officer:	Daniel Opalka

*– The dissemination levels are indicated as follows: **PU** - Public, **PP** - Restricted to other participants (including the Commissions Services), **RE** - Restricted to a group specified by the consortium (including the Commission Services), **CO** - Confidential, only for members of the consortium (including the Commission Services).

Document Control Sheet

Document	**Title:** Application and co-design input	
	ID: D1.1	
	Version: 1.0	**Status:** Final
	Available at: https://www.iosea-project.eu/	
	Software Tool: LaTeX	
	File(s): IO-SEA_D1.1-report.pdf	
Authorship	**Written by:**	E. B. Gregory (FZJ), P. Couvée (ATOS), M. Golasowski (IT4I)
	Contributors:	D. Caviedes Voullième (FZJ), J. Hawkes (ECMWF), O. Iffrig (ECMWF), T. Leibovici (CEA), A. Lopez (ATOS), L. Strafella (CEA)
	Reviewed by:	C. Clauss (ParTec) S. Mimouni (ATOS)
	Approved by:	Exec Board/WP7 Core Group

Document Keywords

Keywords:	IO-SEA, HPC, Exascale, Software

Contents

Project and Deliverable Information Sheet 1

Document Control Sheet 1

List of Figures 5

List of Tables 6

Executive Summary 7

1. Introduction 8

I. Description of IO-SEA Solutions 9

2. IO-SEA Solutions Overview 10

3. Ephemeral Data Access Environment 13

4. Instrumentation and Monitoring 14

5. Hierarchical Storage Management 15

6. Application Interfaces 16

II. IO-SEA Use-case Inputs 18

7. Overview of the Use Cases 19

8. RAMSES : Astrophysical Plasma Flows 20
8.1. Overview . . . 20
8.2. RAMSES workflow . . . 21
8.3. Upcoming challenges of the field . . . 21
8.4. Applying features of the IO-SEA solution to RAMSES . . . 21

9. Analysis of Electron Microscopy Images 23
9.1. Identified challenges . . . 23
9.2. Data description . . . 23
9.3. Image processing pipeline . . . 24
9.4. Applying features of the IO-SEA solution to cryo-electron microscopy . . . 25

10. ECMWF Weather Forecasting Workflow 26
10.1. Overview . . . 26
10.2. ECMWF workflow . . . 26
10.3. Upcoming challenges of the field . . . 26
10.4. Applying features of the IO-SEA solution to the ECMWF workflow . . . 27

11. TSMP: Multi-physics Regional Earth System Model **29**
11.1. Overview 29
11.2. TSMP workflow 29
11.3. Upcoming challenges of the field 31
11.4. Applying features of the IO-SEA solution to TSMP 32

12. Lattice QCD **34**
12.1. Overview 34
12.2. LQCD workflow 34
12.3. Upcoming challenges of the field 35
12.4. Applying features of the IO-SEA solution to LQCD 36

13. Summary **38**

List of Acronyms and Abbreviations **39**

List of Figures

1. Modular Supercomputer Architecture example 10
2. Data nodes in the Modular Supercomputer Architecture 11
3. IO-SEA Stack 11
4. DASI will connect IO-SEA use-cases and other I/O middlewares to IO-SEA storage infrastructure. The interface will be generic enough to adapt to any scientific domain and support different types of storage backend. 16
5. Each scientific dataset in DASI will have a defined schema, via configuration. DASI will support read/write/policy operations where a full dictionary of scientific metadata is used as the unique index for the data. DASI will also support setting policies, retrieving or listing data using a query-style subset of this metadata. For example, you can retrieve all data belonging to experiment 42. 17
6. RAMSES dataflow with the Hercule parallel I/O library: checkpoint/restart and post-processing snaphots. 20
7. Cryo-electron microscopy image processing pipeline 24
8. Weather forecasting data flow. One instance of the IFS model is run at high resolution and 51 instances are run as an ensemble at coarser resolution. Each step of the high-resolution run and each common step of the ensemble trigger one product generation run. 27
9. Typical TSMP workflow for a single simulation including all three component models. Approximate I/O volumes for the EUR-11 CORDEX case, per unit model time are included. 31
10. Lattice QCD workflow with four workflow steps. A is gauge configuration generation, and $B - D$ are steps of the measurement phase. Purple ovals represent gauge configurations, cyan ovals are quark propagators, and the green oval represents hadron correlators. Quantitative information is given in Table 1. 34

List of Tables

1. Quantitative description of LQCD data flow. A project or campaign may consist of 10 to 100 ensembles of $\sim 10^5$ gauge configurations each. A subset (10%) are re-read for measurement. A sparse system solver produces propagator solutions (*B*), which are re-read in relevant combinations to be contracted (*C*) to generate hadron correlators which are analysed offline (*D*). 36

Executive Summary

This first IO-SEA deliverable presents the early results of the workshops and co-design sessions held since the project beginning on April 1, 2021.

Part I gives a description of the IO-SEA technical solution, with a focus on the user interaction with the planned technical features. Part II describes the five IO-SEA use cases, from a workflow point of view, and how they will benefit from the specific features developed in IO-SEA. We present for each use-case its I/O activity in terms of produced and consumed data, as well as their requirements for data retention. The upcoming challenges of the field are also identified and the IO-SEA features that will be integrated during the project lifetime are listed.

Both parts describe the evolved understanding resulting from the discussion process, described above, in the first four months of the IO-SEA project. This work is on-going, and will be refined in the upcoming months and the upcoming deliverables.

This deliverable is followed by D1.2 due in M9 which will describe integration of the use-cases described here with the JUBE benchmarking suite. The next deliverable D1.3 due in M12 will then detail how the IO-SEA technologies developed within WP3, 4 and 5 are applied to each use-case.

1. Introduction

The IO-SEA project aims to produce a new data management and storage platform for exascale computing in a Modular Supercomputing Architectures (MSA) environment.

The resulting solution will draw upon novel technologies being developed in the four technical Work Packages. Work Package 2 will develop an ephemeral data access environment, running on data nodes in the modular supercomputing environment. Work package 3 will develop instrumentation and monitoring tools that will analyse I/O performance and benefit both users and administrators. Work Package 4 will provide technology to manage data and integrate the various layers of a hierarchical storage system, such as NVMe, Solid State Drives, hard drives, and tape systems. Finally Work Package 5 will provide application interfaces to the storage system, with efforts centring on building a Data Access and Storage Interface (DASI).

These technologies will be tested by five scientific use cases in Tasks 1.3 – 1.7 in Work Package 1. These are, respectively, the RAMSES Astrophysical plasma flows simulation code, the analysis of cryo-electron microscopy imagery, the ECMWF weather forecasting workflow, the TSMP regional Earth system model, and lattice quantum-chromodynamics simulations. These five use cases have extremely varied workflows and I/O demands. They will provide a broad test suite for the IO-SEA solution.

This report, due at the end of the fourth month of the project, describes the results of the very first phase of the IO-SEA project, in which the use case personnel

The process leading to this report involved instructive presentations of both the use-cases to the audience of the technical Work Packages, and vice versa. A collaboration-wide cross-work package (virtual) workshop allowed broad discussion of both scientific use case demands and also technical features of the IO-SEA solution and how they would be exposed to users. Finally, there occurred a large number of bi-lateral discussions between technical Work Package leaders and use case task leaders. In these the debate focused on ways that specific workflow challenges could take advantage of the novel features to be developed in this project.

We divide this report in two parts. Part I is a description of the IO-SEA technical solution, with a chapter devoted to each technical work package (Work Package 2 through Work Package 5). The focus of these chapters is specifically on the user interaction with the technical features. The chapters in Part II describe the respective use cases, their workflows and how they will benefit from the specific features in Part I.

Both parts describe the evolved understanding resulting from the discussion process, described above, in the first four months of the IO-SEA project. In this sense, this deliverable is the product not just of the use-cases in Work Package 1, but also the efforts and contribution of the four technical work packages.

This is the first major step of the IO-SEA project, though not the final word on the co-design process. As we continue to work together we fully expect to encounter some un-expected challenges and which will inspire further innovative improvements to the IO-SEA solution.

Part I.

Description of IO-SEA Solutions

2. IO-SEA Solutions Overview

The IO-SEA project will build a novel I/O Stack targeted to exascale systems following the *Modular Supercomputer Architecture* (MSA) principles [1]. Those exascale systems will be composed of different *compute modules* providing different computing technologies (such as CPUs, GPU booster modules, ...), on which computational workflows will be executed. Workflows are the execution units considered in IO-SEA. A workflow is typically composed of many *steps* that will be executed either sequentially or in parallel. A step is typically a regular HPC job that reads its input data from files and produces its results in files. IO-SEA targets workflows where steps communicate with each other through files. The different steps of a workflow will be scheduled on the modules that best suit their computing requirements (cf. Figure 1).

In addition to the compute modules, a *long-term data storage* module (multi-tier storage system) provides a hierarchical storage infrastructure to record data on disks and tapes.

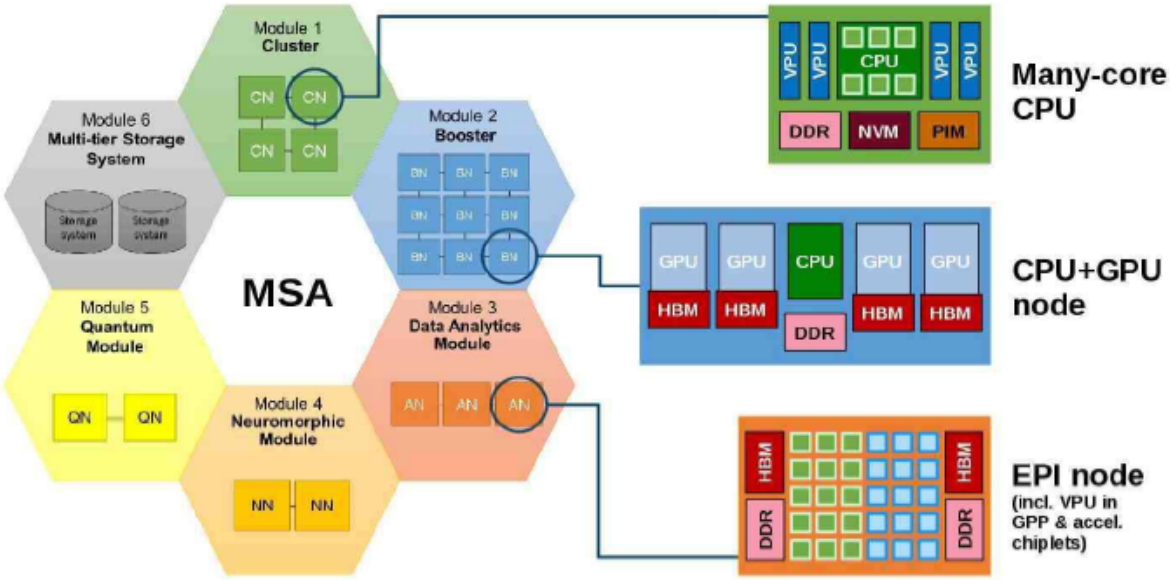

Figure 1.: Modular Supercomputer Architecture example

If modules are not closely interconnected through high-speed/low-latency interconnects, I/O operations from compute modules to the long-term storage module could slow down workflows and prevent scalability. For this reason the IO-SEA stack includes dedicated *data nodes* equipped with the latest storage technologies (NVMe, NVRAM). Data nodes are not parts of compute modules, but they will be connected on high speed/low latency interconnects to one or more of them and will act as gateways to access the long-term storage module, as illustrated on Figure 2.

When a workflow will be launched, a set of data nodes attached to the target compute modules will be allocated and software services called *data accessors* and *ephemeral I/O services* will be set up. These software services will be designed to leverage the high speed storage resources of the data nodes and hide as much as possible the higher latency and the lower throughput to the long-term storage module. They enable a fast and local communication through files between workflows steps,

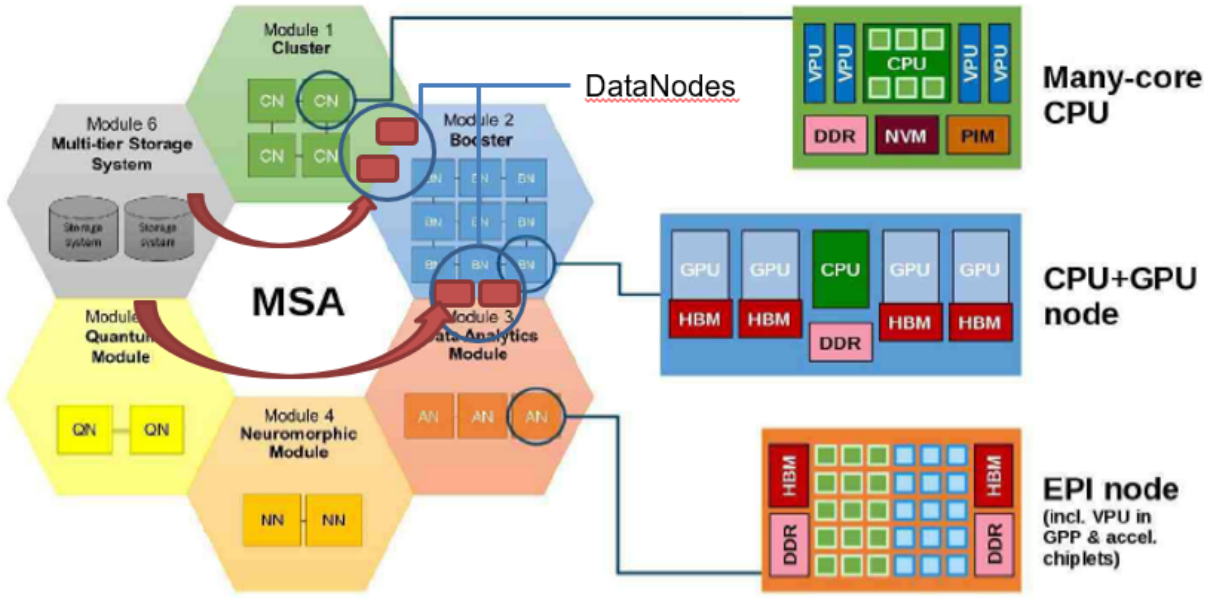

Figure 2.: Data nodes in the Modular Supercomputer Architecture

even when the steps are not executed on the same compute module. Data nodes act as the highest tier of the IO-SEA storage infrastructure with the particularity of being specifically allocated and configured for a workflow upon its I/O requirements.

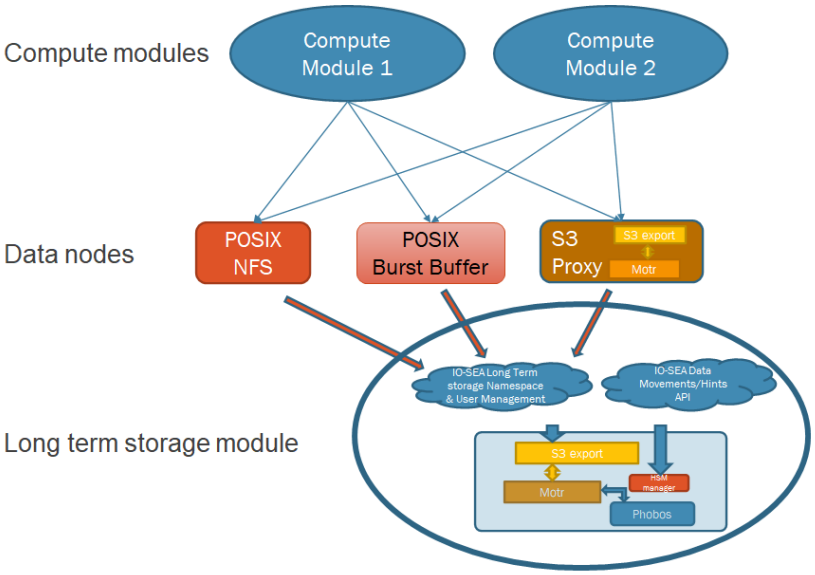

Figure 3.: IO-SEA Stack

The long-term storage module integrates different technologies from the fastest (NVMe disks) to the cheapest (tapes) into a unified tiered solution. It relies on object storage technologies to overcome POSIX parallel file systems limitations and ensure exascale scalability. Object storage proposes a simple organisation of data in just two levels: *objects* are containers to store any kind of raw data,

and *buckets* are groups of objects that share metadata (ownership, ...) and can be manipulated as a whole (moved, archived, ...).

IO-SEA users will be invited to import and organize their data as *data sets* in the long-term storage module. A data set is a collection of objects whose contents share affinities from a user/application point of view. Data sets will be stored as buckets.

When submitting a workflow for execution, IO-SEA users will have to specify the data sets they will need access to, as well as the *namespaces* used to expose them to applications on compute nodes. Indeed, IO-SEA separates the data sets themselves from the way they are presented to applications, enabling the same data sets to be exposed either as POSIX directories and files (thanks to an ephemeral I/O service) or as objects (thanks to a data accessor) at different times. Workflows will be able as well to create new data sets and extend/modify existing data sets with new data.

At workflow launch time (cf. Figure 3), ephemeral I/O services and data accessors will be set up, all or parts of the data sets will be transferred from the long-term storage module to the allocated data nodes, and exposed to compute nodes as specified by the namespaces, allowing applications to access them either through POSIX APIs, Object Storage APIs, or the new DASI API implemented in WP5. IO-SEA users will be given tools to manage their data sets in the storage hierarchy (move data sets from fast to slower storage tiers, give hints to automate data movements...), and tools to instrument and analyze the I/O behaviour of their workflows.

Upon completion of the workflow steps, ephemeral I/O services and data accessors no longer needed will be stopped. Newly produced data that needs to be persisted will be transferred back to the long-term storage module in the target tier (fast disk, tapes, ...) upon users' requirements.

Finally, IO-SEA will develop for workflows a new data access API with scientific-data oriented semantics (DASI). This API will hide from IO-SEA users most of the complexity of the stack by handling the details of the data storage organisation and proposing abstracted *hints* to manage the data placement in the storage hierarchy. The API will also propose means to run some steps of a workflow directly on data nodes, enabling "in-transit" processing of newly produced data that will minimize data movements.

The development of the IO-SEA stack has been organized in four work packages:

- Work Package 2 will develop the data accessors, the ephemeral I/O services and the data nodes run time environment.
- Work Package 3 focuses on workflow instrumentation and infrastructure monitoring to create a recommendation system that will help to allocate the right data nodes resources at workflow launch time.
- Work Package 4 will build the long-term storage solution, including the tiered storage architecture and the data movements utilities.
- Work Package 5 is in charge of the data access and “in-transit” APIs for workflows.

The next four chapters introduce more in depth their contributions.

3. Ephemeral Data Access Environment

The main focus of this work package is the deployment and use of data accessors and ephemeral I/O services, running on data nodes, in order for users to access their data using a known interface (POSIX files and directories, S3 objects and buckets), while accelerating data access for workflows.

Data nodes are cluster nodes dedicated to providing I/O services. They are equipped with multiple NVMe and/or NVRAM devices used for fast, local storage. The fact that the data nodes are located, from a network point of view, near the compute nodes that will use them is an extra factor of acceleration. Each compute module is possibly connected to multiple data nodes.

Data accessors and *ephemeral I/O services* run on the data nodes, allowing them to take advantage of the fast, local storage and network-closeness. Each instance of a data accessor or an ephemeral I/O service is associated with a specific workflow. They are called *ephemeral* because they share the associated workflow's life cycle: they are created right before the workflow starts running, they are only used by the processes participating in the workflow and they are finally destroyed right after the workflow ends. This will be achieved by using Slurm and OpenStack to allocate the data nodes and take care of the ephemeral data services' life cycle, using existing techniques such as plug-ins, prologues and epilogues, etc. No modification to Slurm or OpenStack is planned.

Processes running on the compute nodes will access data through the ephemeral I/O services running on the data nodes close to them. Provided ephemeral I/O services and data accessors will be based on, but not limited to, Smart Burst Buffer, Smart Bunch of Flash and a NFS or S3 interface accessing the long term storage.

The ephemeral I/O services are taken from this non-exhaustive list:

- ATOS Smart Burst Buffer (SBB) is an intelligent burst buffer on top of an existing parallel filesystem. It can also be used to expose S3 buckets via the POSIX API for application portability. It provides two levels of local cache: RAM and flash storage. A third level will be added to handle NVRAM.
- With ATOS Smart Bunch of Flash (SBF) each compute node running a step of a workflow has a dedicated NVMe storage present on the data nodes exported to the compute nodes through NVMe-oF. It can be used as a fast temporary storage.
- The CEA's NFS exporter is used to expose long-term storage objects into a POSIX namespace.
- Seagate's CortX provides an S3 interface with access management for which we will have to decide if it will also be launched on the data nodes.

Users will be able to express their requirements in terms of data sets, data accessors and ephemeral I/O services when launching their workflows. They can select the input, intermediate and output data sets, which ephemeral I/O services will be launched and their sizing. These requirements will be completed with recommendations from the tools implemented in Work Package 3.

4. Instrumentation and Monitoring

As described in the previous section, the IO-SEA stack relies on data nodes running ephemeral I/O services to provide workflows with a dedicated, fast and scalable access to data. Depending on the access patterns and the volumes of data, it may be needed to move it into the data nodes' fast storage prior to launching a workflow to benefit from a lower access time from compute nodes.

Monitoring I/O activity of workflows is then essential to learn their behaviour and deduce the data nodes resource requirements to ensure optimum performance.

IO-SEA workflows will be instrumented and many I/O related metrics will be collected and saved in databases for further analysis. Metrics will cover the general I/O activity of the workflows (volumes of data accessed for reads and writes, file systems, etc.) as well as more IO-SEA specific metrics such as the amount of data movements between the long term storage module and the data nodes, the latencies of I/O operations as seen from applications. IO-SEA users will be able to analyse the behaviour of their workflows through different graphical user interfaces and reports generated by the tools.

Selected instrumentation tools minimize generated overhead so that the impact on applications' performance is very low. They are activated automatically and every workflow will be instrumented in order to build a "knowledge base" of the workflow runs over time.

In addition to workflow instrumentation, Work Package 3 will set up an infrastructure health monitoring solution covering all MSA supercomputer equipment. It will be based upon the ParTec's ParaStation HealthChecker [2], enriched with new data sources such as data nodes, and with further metrics from other tools (LLView [3], Motr Telemetry framework [4], etc). IO-SEA users will not be exposed to the infrastructure monitoring component.

The last component developed in Work Package 3 is the recommendation system. Leveraging workflow metrics together with infrastructure health data, a recommendation system will be transparently triggered upon workflow submission by IO-SEA users. It will define from previous workflow observations, and from the current health state of the infrastructure, the optimum ephemeral environment to be set up, in terms of number and configuration of data nodes, configuration of ephemeral I/O services and data placement.

Users will likely be able to request a *specific* or *minimum* amount of resources for their workflows at submission time and the recommendation system will adapt it to fit the current available resources and system state. This point is still being designed and more details will be provided later in the deliverable D3.1.

5. Hierarchical Storage Management

Hierarchical Storage Management (HSM) enables the design of storage systems that provide both high performance and large storage capacity at an affordable price and with a reduced total cost of ownership (TCO).

The IO-SEA project will implement a hierarchical storage management mechanism that allows integration of various storage technologies, including NVMe, Solid State Drives (SSD), Hard Drive Disks (HDD) and tapes. In this system, the application data will be located in the technology that best matches application needs in terms of I/O pattern, workload, access frequency, data lifetime, etc. As the needs of the computations can evolve over time, and in order to adapt in real time to the overall load of the system, data can be migrated dynamically between the storage levels.

From the user and application point of view, the heterogeneity of storage resources is made transparent, so that an application does not have to care where the data is actually stored to use the storage system. Thus, the system must be able to reload automatically data from the archival storage when an application needs it. It must also be able to make room on the fast storage levels to make space available for newly accessed data, by migrating older data to slower storage technologies.

Nonetheless, this automated hierarchical storage management can be made even more efficient if the user or application provides information about its future access intentions. For example, if an application knows it will no longer use a specific data — like an output file that is completed and will not be modified again — the application can tell the system that this file could be moved to archival storage and that the associated resources in fast storage resources could be released. To achieve this, IO-SEA plans to implement a way to pass this kind of information across the software layers (workflows, data sets, DASI, ...) down to the HSM, so it can optimise its data placement strategies accordingly: this mechanism is called “hints”.

This hierarchical storage system constitutes the “multi-tiered storage system” described in Chapter 2. It will provide interfaces to access data in the form of objects grouped into buckets. These objects and buckets can be accessed in various ways:

- they can be accessed directly by the application, using an HTTP-based protocol like Swift or S3;
- they can be loaded into a temporary location managed by the Ephemeral I/O services;
- they can be accessed through the new DASI API implemented in WP5.

Additionally, WP4 will implement a way to import data from an existing file system, thus allowing to migrate from a legacy storage to the IO-SEA environment.

The design of this hierarchical storage service and its components is in progress, and will be detailed in later deliverable D4.1.

6. Application Interfaces

Work Package 5 (Application Interfaces) is primarily focused on building a Data Access and Storage Interface (DASI), which abstracts storage systems from the scientists and their applications. This interface will connect the use-cases and other I/O middlewares to the storage infrastructure developed in IO-SEA, and will focus on providing an API for managed, curated, scientific datasets. Work Package 5 will also build a POSIX interface to the storage infrastructure, complementing the data access methods described in Section 3 (Work Package 2) and Section 5 (Work Package 4).

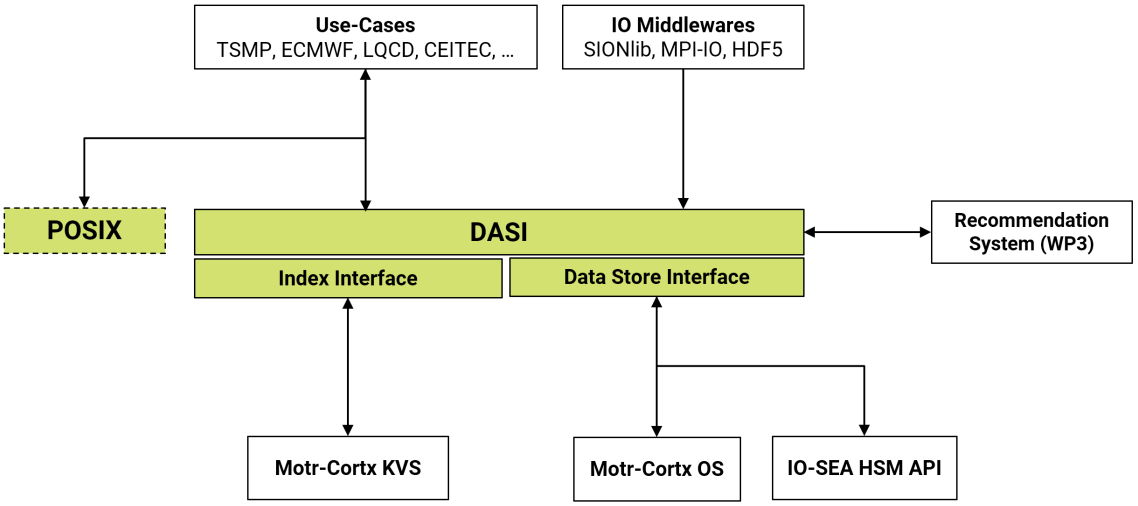

Figure 4.: DASI will connect IO-SEA use-cases and other I/O middlewares to IO-SEA storage infrastructure. The interface will be generic enough to adapt to any scientific domain and support different types of storage backend.

DASI will use a semantic description of data, configured to speak the language of the scientific domains it serves. For example, an application which writes or reads COVID modelling data would address the data, not by filename or by object IDs, but by a set of scientifically meaningful keys such as experiment type, simulation date, timestep and parameter name. Data can be queried using subsets of these keys, and it will also be possible to set policies for data lifetime, data hotness and discovery based on query-matching.

DASI will be provided primarily as an API, but standalone utility tools will also be built which will allow data control outside of the application – for example, to set policies or hints on data long after the application has terminated. This will also allow expression of data movement for efficient workflow management.

Behind the scenes, the implementation of DASI will sit on top of two main abstractions – one for the index and one for the data storage. These will be implemented concretely in the IO-SEA project using some of the variety of technologies available, including the lower-level APIs developed in WP4 and the Motr-Cortx software stack.

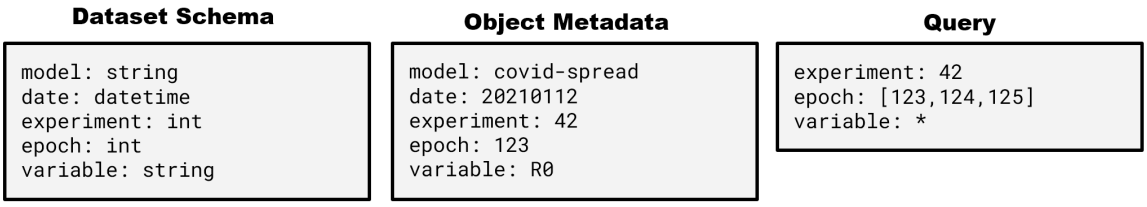

Figure 5.: Each scientific dataset in DASI will have a defined schema, via configuration. DASI will support read/write/policy operations where a full dictionary of scientific metadata is used as the unique index for the data. DASI will also support setting policies, retrieving or listing data using a query-style subset of this metadata. For example, you can retrieve all data belonging to experiment 42.

Each use-case will attempt to use DASI for at least one scientific dataset, as described in Part II. Where possible, I/O middleware libraries such as SIONlib will also be adapted to use DASI; such that applications beyond IO-SEA could leverage DASI too.

Another facet of Work Package 5 is the challenge of enabling data-centric workflows, where scientific applications run in tandem with in-situ processing or visualization. For example, in the ECMWF weather forecasting workflow, product generation runs in parallel with the weather forecasting system – ideally in-situ, using data nodes (see WP2). In Task 5.4, scheduling facilities will be developed to enable this kind of coupling and may use aspects of DASI (e.g. the dataset schema definitions), or the DASI utility tools directly, to express data coupling or movement. This task starts in month 12, and more details, including co-design with use-cases, will feature in later deliverables.

Part II.

IO-SEA Use-case Inputs

7. Overview of the Use Cases

Work Package 1 of the IO-SEA project contains five use cases to test the IO-SEA technical solutions. The cases are: simulation of astrophysical plasma, analysis of images from an electron microscope, weather forecasting, modelling of terrestrial systems, and the simulation of quark and gluon quantum fields. These are real scientific applications chosen for inclusion in the project for their unique I/O challenges. These applications have diverse workflows and data needs. Two of the use cases encounter the time pressures and demands of external customers. Anticipated algorithmic and hardware advances will allow the increase in size, scope and speed of calculations in all of the use cases, which will naturally increase the pressure on their I/O interfaces and existing storage systems.

For the most part the personnel involved in these diverse use cases are not I/O or hardware specialists, rather they are typical scientists working from the various numerical scientific fields. The dialog begun in this co-design process helps to illuminate how the non-specialist will encounter the IO-SEA technical features in future production systems based on MSA principles.

In this part of the report we introduce each use case and its typical work flow, giving careful quantitative thought to the pattern of data flow into and out of the calculation. By exposing the specifics of the data flows, such as the size and number of data objects, the frequency of transfers, as well as the source, destination and expected lifetime of the data, we help the designers of the IO-SEA technical elements to understand how these features will be used in real-world cases and make sure they are up to the task.

8. RAMSES : Astrophysical Plasma Flows

8.1. Overview

RAMSES is an open-source simulation code for astrophysical compressible plasma flows, featuring self-gravitation, magnetism, and radiative processes. It is based on the Adaptive Mesh Refinement (AMR) technique on a fully-threaded graded octree. It is written in Fortran 90 and makes intensive use of the MPI library. RAMSES computational intensive routines have been highly optimised to take advantage of supercomputer architectures. Nevertheless, the RAMSES legacy I/O engine uses the "n files per process" paradigm with a memory footprint of about 2 to 8 GB per MPI processes, due to the fact that MPI processes are bound to core. This system rapidly leads to I/O bottlenecks starting around 8,000 cores. In order to prepare the exascale era, the Hercule parallel I/O library (IO-SEA WP5) has been successfully integrated in RAMSES. This leads to an upgrade of the I/O system of RAMSES that can now make post-processing specific outputs in addition to checkpoints/restarts, see Fig 6.

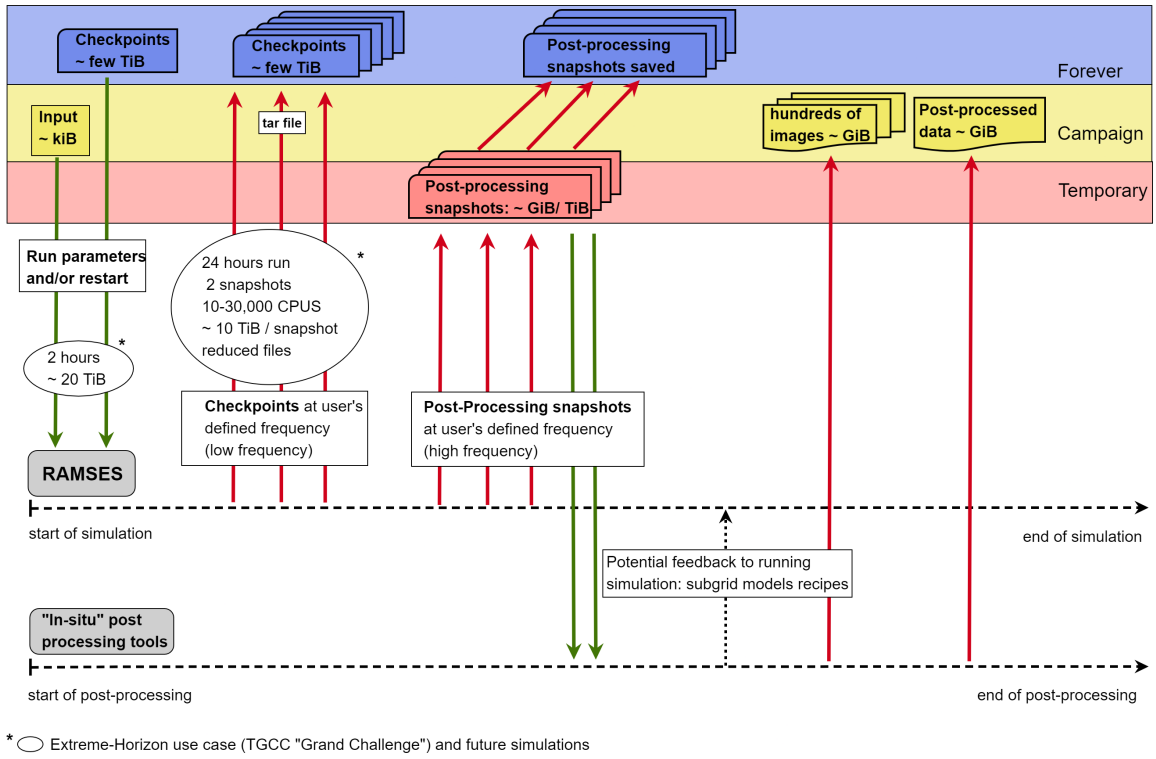

Figure 6.: RAMSES dataflow with the Hercule parallel I/O library: checkpoint/restart and post-processing snaphots.

8.2. RAMSES workflow

The RAMSES workflow is subdivided in two distinct purposes: checkpoint/restart and post-processing snapshots. Both are mono-directional in the sense that data are written to disk and never read during the lifetime of the simulation, prior to IO-SEA project. Because of the use of Adaptive Mesh Refinement (AMR), the amount of data to transfer on each workflow is not static and will evolve with the simulation. Indeed, the AMR technique implies that the mesh of the simulation can vary in time and thus the amount of data managed by MPI processes for both computation and I/O steps also varies. It introduces a lot of variability on I/O volume of each MPI process and it makes the I/O volume request complex. For example, it is common for a RAMSES simulation to have a difference up to 10 times for the amount of data between the most populated MPI process and the least populated one. In addition, the computational time is also directly linked to the amount of data, and therefore, I/O dump time cannot be precisely predicted as a function of CPU time. In addition, the post-processing workflow is dynamic since, within RAMSES, users can select input file and fields of interest that need to be stored during the run of the simulation.

8.3. Upcoming challenges of the field

AMR codes and specifically RAMSES, are a powerful tool for studying multi-scale astrophysical processes. With the advent of the next generation of supercomputer, it will be possible to tackle a new generation of numerical simulation solving large-scale processes with an unparalleled accuracy. One of the key point to tackle is the enormous data volume those simulations will produce for both of RAMSES workflows. In addition to improving I/O performance, "in-transit" and "in-situ" post-processing will become key technologies to post-process results at the same time data are produced in order to reduce final results data storage. In the case of RAMSES, "in-situ" analysis takes advantage of the data directly available in the memory of the nodes to produce images of area of interest within the simulation. On the other hand, "in-transit" post-processing is the equivalent of sending data over a temporary storage from which post-processing tools will be able to consume the data on the fly, for example: halo finding and structure identification. Both of those techniques aim at reducing I/O bandwidth, data volume and post-processing times. The IO-SEA initiative is the first step for the RAMSES code to run those ambitious simulations.

8.4. Applying features of the IO-SEA solution to RAMSES

With the upgrade of RAMSES I/O engine to the Hercule library, high-bandwidth data nodes are an opportunity for the community to significantly improve the scientific analysis process by introducing the possibility of "in-transit" within RAMSES post-processing workflow as shown on figure 6. This can also open the possibility of feedback from the post-processing tools to the simulation itself to dynamically tune some subgrid model parameters on detected structures. Some small-scale physical process could be implemented in unprecedented ways if structure formation is detected on-the-fly during the simulations.

Having tools for monitoring I/O behaviour specifically for RAMSES post-processing dynamic workflow would bring more light on possible optimization and "in-transit" post-processing.

With RAMSES, users manually manage the storage hotness by saving their simulation checkpoint/restart directories as tarball on tapes. The RAMSES workflow is quite linear, and because of the difficulty to manage outputs and storage, "in-transit" data processing is a very difficult task to complete and is therefore generally made "offline". This is a point that we hope, will be improved by the hierarchical storage thus introducing the possibility of "in-transit" data processing to users of RAMSES. In addition, the possibility of direct "on tape" saving of checkpoint/restart will improve data management. Thanks to the integration of the Hercule library, we have already introduced a lot of flexibility in the post-processing data flow of RAMSES by using a data object-oriented approach. This is an important step achieved by RAMSES because the abstraction of the data has a significant impact on the maintainability and development of post-processing tools. It is expected that using DASI interface for in-situ processing and visualization will provide more flexibility, thus the effort of development for new post-processing tools will be more focused on the data analysis.

9. Analysis of Electron Microscopy Images

Cryo-electron microscopy (cryo-EM) is a rapidly evolving technique in the field of Structural and Cellular Biology research. The method is used to derive 3D models of macromolecules from large number of 2D projection images collected by the electron microscopes. Amount of data generated during acquisition of electron micrographs for so called single particle analysis is significant and, what is more, deriving of 3D models requires substantial computational resources.

CEITEC operates several transmission electron microscopes including the state-of-the-art Titan Krios which produces 1-2TB of raw data per day and is operated in 24/7 mode. The raw data are currently stored on the local HDD storage to be further processed to derive the 3D models of the studied macromolecules. The data are processed at a couple of bare-metal machines equipped with GPU also located on premises at CEITEC.

Main objective of this use-case within the IO-SEA context is to use the HPC and storage resources operated by IT4I. The purpose is to improve processing time of the raw data produced by the electron microscope and to facilitate data publication mechanisms.

9.1. Identified challenges

Main challenge of this application is enabling near real time feed-back loop which would allow to control the quality of the produced data during the data acquisition phase. Significant amount of data is generated by the machine, which is operated continuously 24 hours per day. Computing resources needed to enable the near real-time processing of the data are not located on the same premises as the electron microscope, therefore sufficient bandwidth has to be available between the two locations and storage infrastructure capable of handling the amount of data.

During the initial project phase we have identified the following main challenges in the CEITEC EM application:

- Compression and transfer of the raw data from the EM facility to a HPC centre storage
- Orchestration of the image processing workflow using open-source tools
- Publishing and archiving of the RAW data and processing results following FAIR principle[1]

9.2. Data description

The raw data produced by the machine are in the form of TIFF image files. Single file contains a matrix of 40 images with 4k x 4k pixels each. The images are grayscale where each pixel holds intensity values described by a single number between 0-7.

Single dataset is structured in the following way:

[1] FAIR data are Findable, Accessible, Interoperable and Reusable - `https://www.go-fair.org/fair-principles`

- 150 MB per single TIFF file
- Around 7-15 thousand TIFF files in a single dataset
- 2TB of RAW data per dataset, 130 datasets annually from single microscope

9.3. Image processing pipeline

The image processing pipeline shown in Figure 7, which will be deployed on IT4I computing infrastructure, consists of five steps:

1. Transfer of the raw data from the microscope
2. Correction for motion during data acquisition
3. Estimation of the contrast transfer functions parameters
4. Particle selection and extraction
5. Reference-free 2D classification

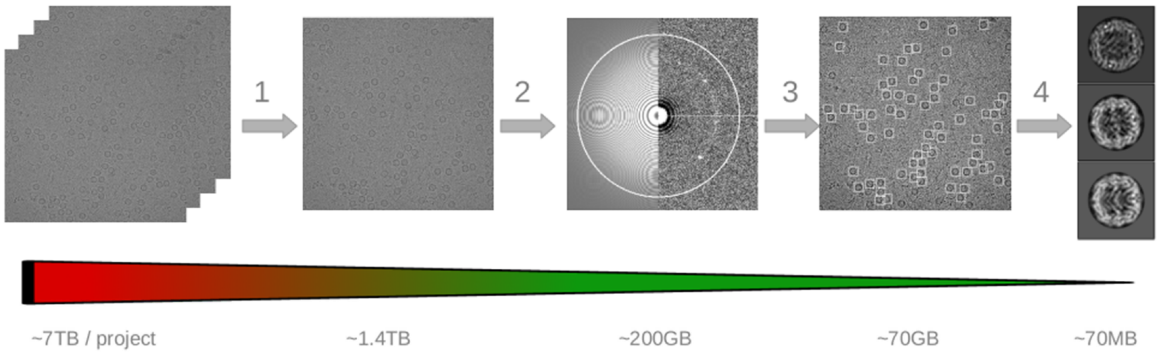

Figure 7.: Cryo-electron microscopy image processing pipeline

Our initial efforts for the deployment of the single particle cryo-EM data pre-processing pipeline will be based on utilization of the existing software packages and their incorporation into a single workflow. The MotionCor2 [5] program will be used to minimize the effect of the sample motion during data acquisition and the gCTF [6] program will be employed to estimate the contrast transfer function parameters. Selection of the particles will be used by either gAutomatch [7] or Cryolo [8], or Topaz [9] programs. Finally, programs from Relion [10] of Sphire [11] software packages will be used to carry out reference-free 2D classification of the extracted particles.

9.4. Applying features of the IO-SEA solution to cryo-electron microscopy

The cryo-EM use case can be described as I/O heavy HPC application, therefore it has a good potential to benefit from the various technologies provided by the IO-SEA platform. The image processing pipeline is by nature an I/O heavy application. Therefore, it would greatly benefit from fast NVMe/NVRAM-based storage available on the compute nodes.

Substantial part of this use case is the acquisition of the data from remote location and their movement closest to the computing resources. Proper monitoring and instrumentation are vital to measure the improvement offered by the IO-SEA platform, which contributes to its successful validation.

Datasets in this use-case have different levels of hotness. Naturally, the hottest data are the newest ones, whose rapid processing can be used to improve the precision of the currently sampled dataset. Then, next level are datasets obtained and published recently, which are usually downloaded by the community to test various processing techniques and tools. The last level (coldest) is the archive of all the datasets produced by the imaging device. HSM is therefore very useful feature of the IO-SEA platform, which can be used to improve the processing time of the cryo-EM data.

Raw cryo-EM data have to be stored with great care since the machinery involved is expensive and the sample preparation process cannot be repeated. There is a lot of effort in the scientific community which goes to development of custom image processing pipelines. Therefore, this use-case puts a strong emphasis on data curation and publication following the FAIR principles in order to enable proper referencing of the original raw data. DASI can be used in this case to address the individual datasets by its metadata for staging, publication and archival of the data. Depending on the level of integration, 3rd party services like B2HANDLE provided by EUDAT [12] can be used for obtaining a unique persistent identifier (PID) and curating its metadata. EMPIAR [13] is a domain-specific public index of cryo-EM data which can be also used to publish data from this use-case.

10. ECMWF Weather Forecasting Workflow

10.1. Overview

ECMWF is an intergovernmental organisation supported by 23 member states and 11 co-operating states. It is a research institute and a 24/7 operational service, producing numerical weather predictions and other data for the weather and climate communities. The Centre has the largest meteorological data archive in the world (over 300 PiB of meteorological data) and also has one of the largest supercomputer facilities. ECMWF's primary forecasts consist of a 9 km, 10-day high-resolution (HRES) deterministic forecast and an 18 km, 15-day, 51-member ensemble (ENS) forecast – these run four times per day and cover the entire globe. Weather prediction data is incredibly valuable to a wide range of industries, but it is only valid for a short time. As such, the forecasts are time-critical and data must be delivered to downstream consumers according to a tight schedule.

10.2. ECMWF workflow

ECMWF's operational workflow is built around the flow of data: observations come in and are assimilated into the Integrated Forecasting System (IFS), whose output is then post-processed and distributed as well as archived. This workflow uses a significant part of ECMWF's computing resources and produces about 30 TiB of model data per run. The time-critical part of the workflow has to run within an hour. It consists of the aforementioned high-resolution forecast and 51 coarser-resolution ENS forecasts, as well as the associated post-processing 'product generation'. As soon as either the high-resolution run or all 51 ensemble runs have produced a snapshot, about 70% of it is processed to generate products to be disseminated to the member states and customers. This creates a significant concurrent read/write contention on the filesystem that requires careful tuning of the I/O pipeline. The data flow and the corresponding data volumes are pictured in Fig. 8.

The datasets used in the pipeline are carefully curated using a domain-specific language. In the case of the model output, the data are aggregated into two-dimensional fields: the values of one parameter all around the Earth (e.g. wind vorticity at a fixed altitude). Each field is uniquely identified by a set of scientifically meaningful key-value pairs (e.g., operational or research dataset, experiment identifier, timestamp of the run, timestep inside the forecast, vertical level, physical parameter, . . .). This allows for efficient access and storage of the fields in a dedicated object store and perpetual tape archive. A variety of auxiliary I/O (configuration, log files, etc.) are handled via standard POSIX files.

10.3. Upcoming challenges of the field

The challenges ECMWF (and weather forecasting in general) is facing arise from both technical and scientific concerns. On the scientific side, ECMWF's forecasts are constantly evolving, leading to:

- Increased model complexity (Earth system physics), with direct impact on the model range in time

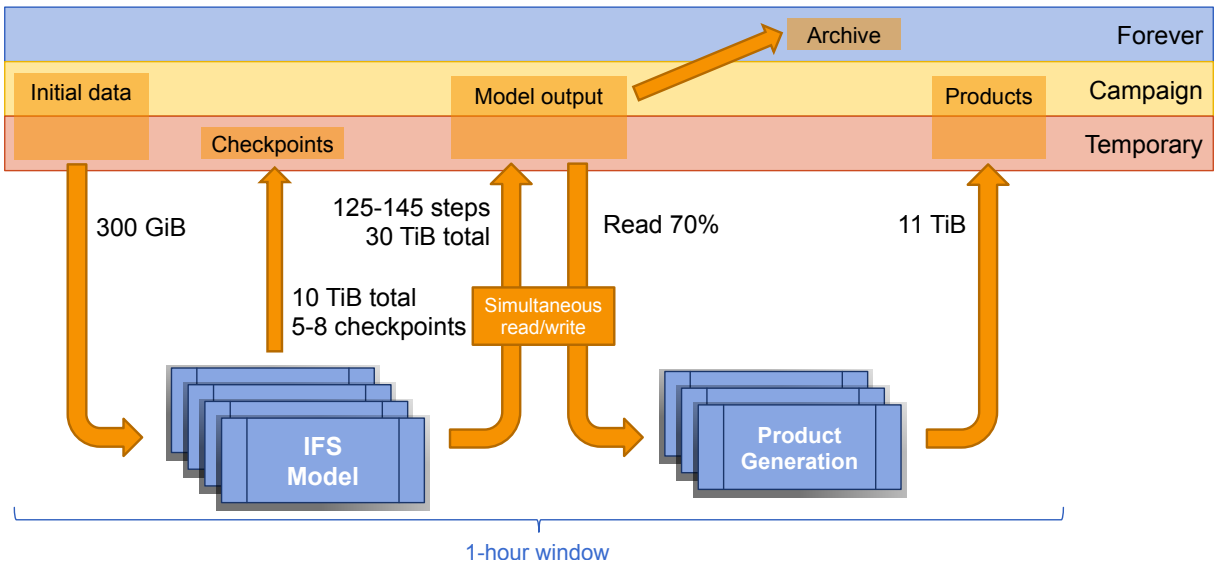

Figure 8.: Weather forecasting data flow. One instance of the IFS model is run at high resolution and 51 instances are run as an ensemble at coarser resolution. Each step of the high-resolution run and each common step of the ensemble trigger one product generation run.

- Increased size of the ensemble, for better reliability
- Higher model resolution

To allow scientific improvements to be implemented, the appropriate technical resources need to be available as well:

- Increase in computational power
- Good use of novel computing architectures (GPU, non-volatile memory, . . .)
- Increase in data storage and transfer capacity

As a consequence of these changes, one can expect a sustained increase in the amount of data that will be processed, which will also require adaptation in the way forecast data is shared and accessed:

- Cloud-based technology to bring the data consumers to the data, instead of moving the data
- Optimisation of data access patterns (e.g. access to subsets of fields, enhanced data-centric feature extraction)
- Interaction between multiple data providers

10.4. Applying features of the IO-SEA solution to the ECMWF workflow

Weather forecasts have a high value that sharply drops when time goes on. Therefore, any improvement that allows for a higher data throughput can make a significant difference given the amount of data transiting through the ECMWF weather forecasting system each day. In this regard, the IO-SEA

solution will provide a set of features allowing for more efficient use of the current and future compute and storage infrastructures.

Having access to data nodes equipped with high-bandwidth storage like NVRAM may be a game changer for the ECMWF workflow. In particular, it has been shown that the read/write contention issue described above can be solved by using NVRAM [14]. The domain-specific object store used at ECMWF will be adapted to make good use of the ephemeral storage solution developed in Work Package 2.

ECMWF manages a hierarchical storage management interface: the MARS [15] archival and retrieval system allows the manipulation of data stored on tapes and disk cache, but also on the parallel filesystems used by the operational forecasting pipeline. ECMWF will benefit from the developments of Work Package 4 by integrating the relevant improvements into MARS. In particular, optimising data placement is crucial to ensure that the storage resources available are used correctly and efficiently. Having an automated solution capable of analysing the I/O behaviour of applications as proposed by Work Package 3 would enable more efficient management of the storage resources available at ECMWF.

The DASI, developed in Work Package 5, is in essence a generalisation of the interface used to manipulate ECMWF data. Therefore, porting the IFS I/O stack to use the DASI should be straightforward. This will enable ECMWF to leverage all the features of the IO-SEA solution developed in Work Packages 2, 3, and 4 transparently. Additionally, the in-situ processing facilities developed in Task 5.4 can be leveraged to run product generation as close to the data as possible, thus reducing the potential for I/O bottlenecks.

11. TSMP: Multi-physics Regional Earth System Model

11.1. Overview

The Terrestrial Systems Modelling Platform (TSMP) is a fully coupled, scale consistent, highly modular and massively parallel regional Earth System Model. TSMP (v1.2.3) is a model interface which couples three core model components: the COSMO (v5.01) model for atmospheric simulations, the CLM (v3.5) land surface model and the ParFlow (v3.7) hydrological model. Coupling is done through the OASIS coupler. TSMP is also enabled for Data Assimilation (DA) through the Parallel Data Assimilation Framework PDAF. TSMP allows to simulate complex interactions and feedbacks between the different compartments of terrestrial systems. Specifically, it enables the simulation of mass, energy and momentum fluxes and exchanges across land surface, subsurface and atmosphere [16]. TSMP is maintained by the Simulation and Data Lab Terrestrial Systems (SDLTS) at JSC, and is an open source software publicly available in GitHub [1].

The coupling design is inherently modular, allowing to build all combinations of component models, or only build one of them. This design also leads to a multiple-program-multiple-data (MPMD) execution model and operational flexibility, allowing the different model components to run at different spatial and temporal resolutions. In fact, within TSMP different versions of the component models are supported for both legacy and experimental purposes.

TSMP is undergoing significant developments. Experimental development branches include the coupling of the ICON atmospheric model and the upgrade from CLM3.5 to CLM5. These implementations are at different stages of development but are expected to become operational within a year.

The component models – developed by third-parties – are written using different programming languages, use different parallelisation and acceleration schemes (can exploit different hardware), and show different scaling behaviour. The diversity in features and responses is partly what motivates the modular design of TSMP.

TSMP is mostly a compute-driven application. The core computational effort comes from solving large sets of partial differential equations. The computational requirements are higher for COSMO/ICON and ParFlow, and considerably lower for CLM. Additionally DA ensemble runs are somewhat data-driven, as observational data needs to be assimilated into the workflows.

11.2. TSMP workflow

The typical, single simulation TSMP workflow is illustrated in Figure 9. At the core of the workflow are computationally-intensive numerical simulations. Within the core simulation pipeline, the core component models are initialised, and all three component models are run concurrently. The different complexities and numerical approximations result in different runtimes to a particular time level.

[1] https://github.com/HPSCTerrSys/TSMP

In particular CLM will typically finish first. Once the two other components run until the desired time level, information is exchanged from COSMO/ICON and ParFlow into CLM, which can start advancing again, and then exchange the information back again, followed by triggering the other two component models to continue marching forward. This process is repeated throughout the simulation. TSMP runs with an MPMD execution model, with the individual model components running on different computational resources (nodes) of the system. It is foreseen that in the upcoming MSA exascale systems, the components will run on different modules. Moreover, component models solve processes at different spatiotemporal resolutions (as dictated by numerical stability requirements), and solve systems with very different complexity. Consequently, both the computational effort and the I/O volumes and throughput required are different for each of them.

Within the simulation, TSMP requires boundary-forcing input data which is obtained from global simulations got from third-parties. Each of the component models writes output files at user prescribed frequency (model time). These frequencies are independent of each other, independent of the computational time step of each model, and independent of the coupling time step used to exchange information between component models. Consequently, output from the component models may or may not be concurrent, and each of these components performs I/O through the computational resources (nodes) allocated to it.

I/O is not controlled by TSMP itself, but rather by its component models. The key dependency for I/O is the netCDF library.

Pre-processing workflows are necessary to set up initial conditions (initialisation), and to prepare (interpolation/projection) boundary forcing data (in particular for the atmospheric component) which will be used throughout the simulation. Post-processing workflows are used for different types of visualization, analytics, etc. Data Assimilation workflows are also frequent with TSMP. DA workflows rely on ensembles of concurrent runs (ensemble members) of the simulation pipeline shown in Figure 9.

Additionally, Figure 9 includes typical I/O volumes for a typical use case of TSMP. This case is commonly referred to as the EUR-11 CORDEX case, which represents Europe at a resolution of 0.11° (approximately 12×12 km). Extensive experience exists [e.g., 17, 18] on this use case at SDLTS (FJZ) and collaborator at the Institute of Bio- and Geosciences: Agrosphere (FZJ). Different durations of this test case, from a few hours of model time to months and decades, have been performed at JSC's HPC systems. Model initialisation requires a set of files for each model component, from namelists (a few MBs) to netCDF files (in the order of 100 MBs). Boundary forcing data is in the form of netCDF files, usually obtained from global atmospheric simulations, downscaled and interpolated into the typical EUR-11 domain. Boundary data volumes depend on the frequency at which forcing is imposed on the model. Approximately 3 GB of input data are required per hour of model time. All component models support the netCDF standard for output. COSMO additionally supports GRIB and ParFlow also supports a native, non-standard binary output format. In practice, mostly netCDF files are used in TSMP jobs. Total output volumes depend on the output frequency. For the typical case, a run would require approximately 15 GB per model day (with hourly COSMO and CLM output, and daily ParFlow output). During runs in JUWELS (at JSC) of the typical EUR-11 case show I/O rates peaking at ~ 850 MB/s, for both read and write to the fastest storage tier available.

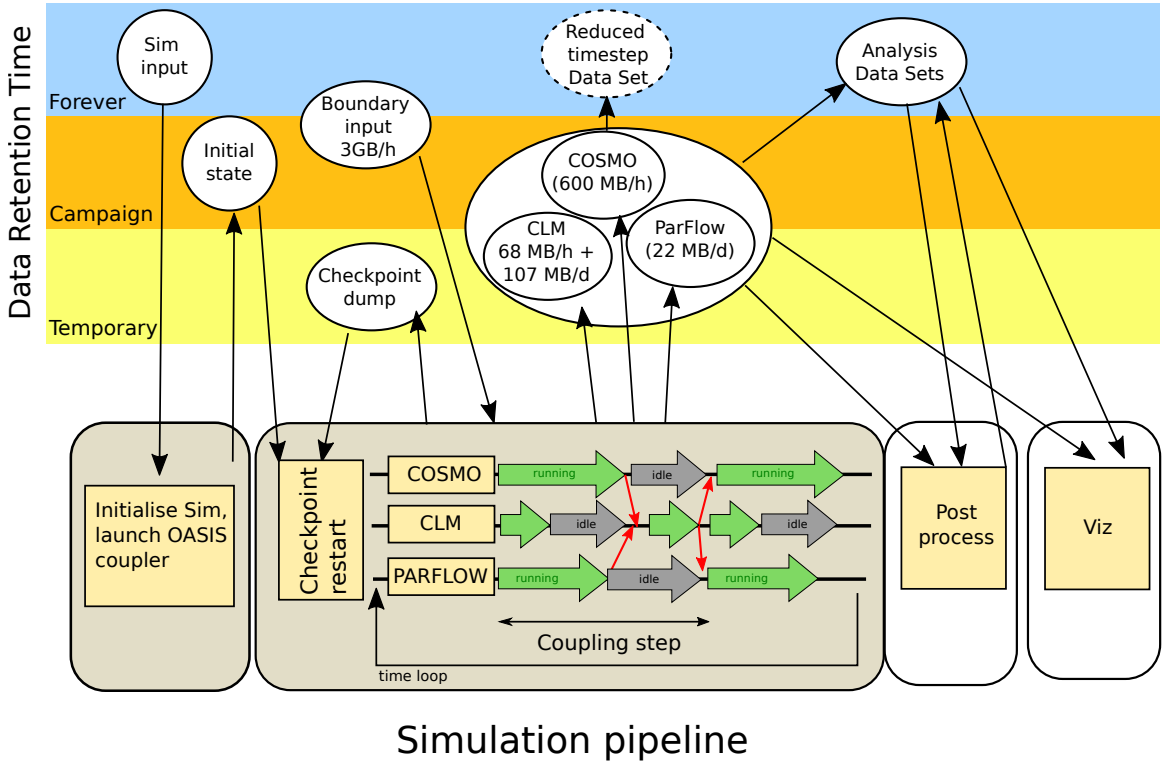

Figure 9.: Typical TSMP workflow for a single simulation including all three component models. Approximate I/O volumes for the EUR-11 CORDEX case, per unit model time are included.

11.3. Upcoming challenges of the field

Current continental-scale models solved with TSMP run at a spatial resolution of approximately 12×12 km. This operational setup is likely to be continued to use for the foreseeable future. Recently, one of TSMP's component (ParFlow v3.7) has been ported to CUDA, enabling its execution on NVidia GPUs. It is foreseen that the atmospheric components of TSMP will also be available for GPUs during the lifetime of IO-SEA. Together, these developments will result in a significant acceleration of the most computationally-intensive kernels, thus reducing computational time. The I/O volumes should remain constant, thus increasing the (real time) frequency of output and warranting a higher throughput. This will also lead to an increased proportion of runtime dedicated to I/O, increasing its relevance in terms of performance.

These accelerator-porting efforts, together with the upcoming exascale computers will allow for an increase in model resolution, which is strongly desirable from the terrestrial systems science point of view. The target resolution for coupled land-atmosphere simulations will be in the order of 3×3km, and possibly higher [19, 20]. The increase in grid points will represent an increase of ~ 13.2 in I/O volumes, going from 15 to 198 GB of output per model day, if the same frequency is kept. However, higher output frequencies may become relevant in such a problem. Managing both I/O and storage of these increased volumes is a key challenge. It is difficult to predict currently if the increased cost due to resolution will result in slower runs despite the increase in computational performance of exascale

machines. Consequently, it is also difficult to assess whether the I/O throughput will be higher or lower than the current one. Additionally, workflows (beyond the core numerical simulations) will face challenges in recovering and processing such higher volumes of data. This will warrant to develop new workflows to harness novel I/O and storage solutions, and/or in-situ post-processing which will avoid writing the simulation output altogether (whenever possible).

Finally, it is envisioned that TSMP will also be used for larger systems, such as the CORDEX Africa domain, roughly three times larger than the European domain, with its consequent increase in computational and I/O requirements.

11.4. Applying features of the IO-SEA solution to TSMP

Following a set of discussions and workshops, a number of clear pathways exist for TSMP to benefit from the technical solutions being developed by the different Work Packages in IO-SEA. In what follows, these are briefly summarised.

The high-speed data nodes planned in the I/O stack will enable asynchronous data staging for TSMP's I/O. NVRAM will likely allow to significantly reduce the time required to (i) stage input data from long term storage tiers and (ii) the compute-idle time waiting for the component models to dump their output. Additionally, dedicated data nodes will likely be the resources required to (i) ingest TSMP's native POSIX file output into DASI, (ii) collect the output from the different component models (running each on different computational nodes) into a centralised resource, (iii) process simulation output in-situ / on-the-fly, (iv) stage cold(er) data for post-processing workflows.

The MPMD execution model of TSMP, together with the different computational requirements of each component imply a large degree of asynchronous processes also including I/O. This feature is currently not exploited to optimise I/O. Instrumentation and monitoring features are necessary to formally and fully characterise TSMP's I/O performance, and to identify possible optimisations such as fine-grained control of I/O timing, optimally-timed data staging for boundary forcing data, and most importantly, data-resource allocation (for I/O and in-situ post-processing). It is relevant to highlight that TSMP components have computational load imbalances, and consequently idle times, due to the different computational requirements of each of them. Although such idle times are an inefficiency in the system, they may also provide a time in which I/O operations can be performed. Exploring this concept will require a detailed instrumentation of I/O processes.

The HSM concept will facilitate managing TSMPs simulation output. Depending on the use that simulations are intended for, different degrees of accessibility and lifetime are required for the output data. Very often, results for reanalysis simulations are stored in full and forever but require a low accessibility. In contrast, specialised studies with interests on a specific subset of variables (e.g., soil water content) may require high accessibility to a small subset, and potentially require storage of the rest of the dataset only during the project lifetime. All these aspects fall well within the capabilities of the HSM. Additionally, TSMP requires input for boundary forcing throughout its runtime. These datasets are likely to be stored in persistent data tiers (perhaps forever), but during the simulation they should be staged for input on fast access data tiers, which follows the design of the HSM.

TSMP would benefit from using both the POSIX and DASI interface to access the HSM. The initial co-design discussions with WP5 suggest that implementing the DASI API directly into TSMP may

suffer from strong constraints. These arise mainly from the fact that TSMP does not manage I/O itself. This is done by the component models, which are not owned by TSMP (that is, not by SDLTS/JSC). Moreover, the implementation of the coupling strategy is based on patching the source codes of the component models, explicitly targeting a minimally-invasive effect on the component's source codes. It is in theory possible therefore to include a patching strategy to directly access the API through TSMP, but it is necessary to gauge how invasive this would be on third-party codes and how sustainable this may be within the software development plans for TSMP. Nonetheless, discussion with WP5 has led into the need for DASI to provide standalone binaries, which would allow the raw (POSIX) TSMP output to be then pushed into DASI, without invoking the DASI API from within TSMP.

12. Lattice QCD

12.1. Overview

The goal of a numerical LQCD calculation is to understand and predict the properties of hadrons, or particles composed of quarks and gluons. The starting point is the QCD action, an equation describing the interactions of quark and gluon quantum fields. Quarks and gluons interact via the strong force and have a charge characterised by a “colour”, hence the name quantum-chromodynamics. Clever mathematical transformations, namely, treating time as an imaginary variable and taking space-time as a discrete and finite 4-D grid (hence the “lattice”), allow the simulation of quark and gluon fields as a statistical physics problem. First, a large statistical ensemble of background gluon field configurations (known as “gauge configurations”) is generated through Monte Carlo simulation. Second, the configurations are re-loaded, and on each configuration operators corresponding to physical observables are calculated. The average over the ensemble of these operators is interpreted as the expectation value of the observable, such as one might measure in a suitable experiment.

In the following sub-sections we describe the LQCD workflow from the viewpoint of data movement, and highlight how IO-SEA features will be implemented and improve scientific output.

12.2. LQCD workflow

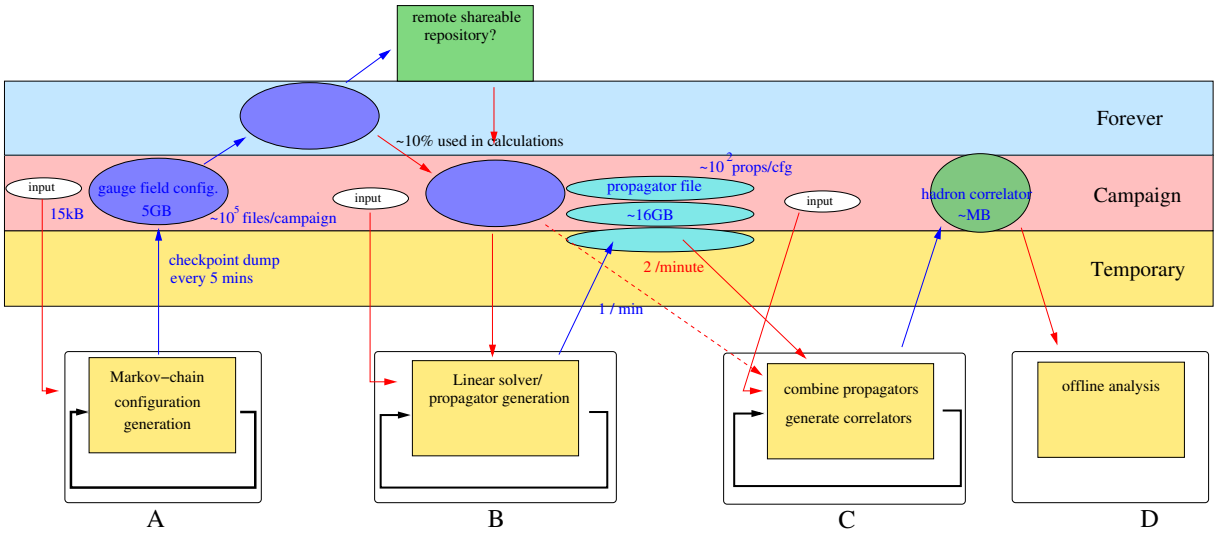

Figure 10.: Lattice QCD workflow with four workflow steps. *A* is gauge configuration generation, and *B* – *D* are steps of the measurement phase. Purple ovals represent gauge configurations, cyan ovals are quark propagators, and the green oval represents hadron correlators. Quantitative information is given in Table 1.

We begin with the caveat that, from the standpoint of data movement, there is no true “typical” LQCD problem. Problem complexity, problem size and computing architecture can affect the data volume

and velocity by several orders of magnitude. However, we show in Fig. 10, a fairly common workflow, divided into four steps, labelled *A* – *D*.

Step *A* is gauge configuration generation, in which new configuration are generated from proposed updates to the previous ones. A Metropolis test determines whether the update is accepted or rejected, but regardless, the configuration is check-pointed to disk. The checkpoint destination may be a temporary scratch disk, but at JSC compute nodes are connected to fast campaign-type storage disks. A single gauge configuration is perhaps 5 GB in size, though may be an order of magnitude larger. A single update may take five minutes between checkpoints.

A collection of configurations generated with the same input parameters is an *ensemble* and may contain $\sim$3000 configurations. At a later time, when the ensemble is complete, it is backed up to tape (forever storage), and may even be made available to a public repository for sharing with other research groups worldwide. A completed ensemble is reusable for a large variety of physics calculations.

Steps *B* – *C* are the “measurement phase”, in which the gauge configurations are loaded and a calculation performed on each. Because the successive checkpoints are highly correlated, it is normal to load only every tenth configuration for measurement. The measurement of each configuration is independent. With enough available compute nodes, all stored configurations could be loaded and measured concurrently.

The precise details of the measurement phase depend on the quantity to be studied. Almost always, however, one must solve a large, sparse linear system multiple ($\sim 10^2$) times on each configuration. The solution vectors are known as quark propagators (shown in cyan in Fig.10) System solvers have traditionally been the most costly part of the calculation, but fast GPU algorithms have reduced solver time to a propagator per minute in some cases. Propagators may be 20 GB each in size.

In Step *C*, groups of a few quark propagator vectors are combined in element-wise products and reductions (contractions) to produce "hadron correlators" (green ovals in Fig.10), arrays of real numbers with total size of a few MB. The contraction may or may not require the gauge configuration to also be in memory. Hadron correlators are analysed offline for physics results.

For simple calculations, *B* and *C* may be a single step with a repeated solve–contract cycle, and propagators are discarded after each contraction. For more involved problems, the same propagators are re-used many times and are saved in temporary- or campaign-level storage between *B* and *C*.

12.3. Upcoming challenges of the field

Several categories of QCD problems require the production of a large number of propagators per configuration, and then the contraction of subsets of these in an even larger number of combinations and/or permutations.

As an example, an atomic nucleus with N nucleons (protons and neutrons) contains $3N$ valence quarks. The correlator for this nucleus would have to contain contractions for all valid permutations of contracting the $3N$ quarks propagators. This is currently an intractable problem for all but the lightest nuclei.

Workflow step	Type	size	freq.	mode	number
A: Config. generation	Parameter input	200 kB		input	
	Configuration	5 GB	5 mins	output	$\sim 10^5$/ensemble
B: Propagator solver	Parameter input	200 kB		input	
	Configuration	5 GB	once	input	
	Propagator	20 GB	2 min	output	$\sim 10^2$/configuration
C: Contractions	Parameter input	200 kB		input	
	Propagator	20 GB	1 min	input	$\sim 10^2$ re-reads /config.
	Correlator	1 MB	20 min	output	
D: Analysis	Correlator	1 MB		oflline	

Table 1.: Quantitative description of LQCD data flow. A project or campaign may consist of 10 to 100 ensembles of $\sim 10^5$ gauge configurations each. A subset (10%) are re-read for measurement. A sparse system solver produces propagator solutions (*B*), which are re-read in relevant combinations to be contracted (*C*) to generate hadron correlators which are analysed offline (*D*).

A second example involves a technique called stochastic sources. In this case the source vector for the linear solver is filled with random variables. Correlators estimates are noisy for a single evaluation, but with N evaluations the uncertainty decreases like $1/\sqrt{N}$.

A traditional way of evaluating contractions with two stochastic propagators would be to run N iterations of the combined [*B* – *C*] solve–contract–discard cycle, where the solver step produces two propagators. This would give N estimates of the correlator for the $2N$ propagators. However, if *B* and *C* are separate and the propagators are stored, they can be read in to form $N(N-1)/2$ estimates of the correlator by using all combinations. Furthermore, in this case the solver and contractions can run concurrently and on heterogeneous hardware, e.g., GPUs for the solver step *B* and cheaper CPUs for the contraction step *C*. This is an example of modular supercomputing and this case is particularly interesting in the context of IO-SEA features.

12.4. Applying features of the IO-SEA solution to LQCD

Several features of the IO-SEA solution should improve efficient and scientific output of LQCD calculations.

High-speed data nodes on a modular supercomputer could provide great advantages to both the configuration generation phase and the measurement phase of the LQCD workflow. In workflow step *A*, the data nodes in burst-buffer mode will allow the compute nodes to continue generating updates with less time lost to check-pointing.

The data nodes provide clear advantages to the measurement phase in the case described above, where the number of contraction combinations is much larger than the number of propagators. Here, the nodes in workflow step *C* must repeatedly read in different subsets of the available propagators. Having the propagators staged on fast data nodes will accelerate the workflow step.

It is expected that by tagging different workflows the instrumentation and monitoring features will help diagnose workflow inefficiencies and aid in better resource allocation.

We expect that tagging data objects and providing hints about where it will be needed will allow the WP4 features to more effectively position data in the storage system.

The DASI interface proposed in WP5 will be used initially to provide access to generated gauge configurations within the hierarchical storage system. Such gauge configurations are categorised by metadata including physical parameters, simulation size, and algorithm parameters. For the purposes of DASI categorisation, we will need to generate an additional unique string to identify the run, for protection in the case of later generation of a similar data set that differs only by an "irrelevant" algorithm parameter. Furthermore, we will need the ability to set some read/write access policy at the user or group level.

Full integration of the DASI interface into the LQCD applications would require a re-write of the application I/O library as well as more minor changes to the application itself. Currently, many LQCD community codes rely on a common stack of libraries, with the QIO library handling I/O of lattice data fields. While an extension or re-write of QIO is possible, a preferable strategy is at the moment to produce a staging utility that moves gauge configurations between the data nodes and longer-term storage.

13. Summary

During the first four months of the IO-SEA project, the activity has been focused on sharing the knowledge about use cases and all the technical aspects of the IO-SEA solution. Many workshops and cross-work-package discussions have been held to produce this initial description of the IO-SEA stack and to sketch how use cases will leverage it.

The structuring features of the IO-SEA solution are now identified and shared with IO-SEA use cases. The data flow diagram that has been chosen to describe use cases workflows will be helpful to map them to our use cases:

- They will be used to define their data sets organization, the name spaces and access APIs (DASI, POSIX or objects), and the data accessors and ephemeral I/O services they will use.
- Having identified the volumetry and the frequency of accesses to data sets will help to focus the instrumentation on the most significant data access sequences.
- They will help to define the data set life cycles and the data movements to be done before, during and after workflow executions.
- The opportunities to use an “in-transit” processing strategy are also easier to identify on these diagrams.

Moreover, the innovative DASI API will be considered by all use cases and initial discussions have led to identifying more features to be added into the DASI ecosystem, such as import/export or backup/archival utilties.

Lower level technical aspects are still being designed and the co-design effort will be continued in the coming months and will be presented in the deliverables D2.1, D3.1, D4.1 and D5.1, which are due at M12. However, this initial effort highlighted that the use cases are very relevant to demonstrate how the IO-SEA innovations will help to solve the forthcoming challenges in their respective areas.

List of Acronyms and Abbreviations

A

ATOS	ATOS is Europe's largest digital services deliverer.

C

CEA	The French Alternative Energies and Atomic Energy Commission.
CEITEC	Central European Institute of Technology, Brno, Czech Republic.
CLM	Community Land Model.
COSMO	Consortium for Small-scale Modeling.
CUDA	Compute Unified Device Architecture, API and programming language for GPU accelerators.

D

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

E

ECMWF	European Centre for Medium-Range Weather Forecasts.

F

FZJ	Forschungszentrum Jülich, in Jülich, Germany, is one of the largest research centres in Europe and a member of the Helmholtz Association.

G

GPU	Graphics Processing Unit.

H

HDD	Hard Drive Disk.
HSM	Hierarchical Storage Management.
HTTP	HyperText Transfer Protocol: common Internet protocol to access web pages or download files.

I

I/O Input/Output.

IFS Integrated Forecasting System, ECMWF's operational weather forecasting system.

IT4I IT4Innovations National Supercomputing Centre at VSB Technical University of Ostrava, Czech Republic.

J

JSC Jülich Supercomputing Centre at IO-SEA partner Forschungszentrum Jülich.

L

LQCD Lattice quantum-chromodynamics is a numerical framework for calculating physical properties of hadrons, composite particles composed of quarks.

M

MARS Meteorological Archival and Retrieval System, ECMWF's perpetual archive service.

MPMD Multiple Program Multiple Data.

MSA Modular Supercomputing Architecture.

N

NVMe Non-Volatile Memory Express.

NVMe-oF NVM Express over Fabrics.

NVRAM Non-volatile Random Access Memory.

O

OASIS OASIS3-MCT is a software allowing synchronized exchanges of coupling information between numerical codes representing different components of the Earth System.

P

ParFlow A physically-based and spatially distributed hydrological model solving surface and subsurface flows in a massively parallel computational framework.

ParTec ParTec is one of the leading SMEs in the HPC domain in Europe.

PDAF Parallel Data Assimilation Framework.

PID Persistent Identifier.

POSIX Portable Operating System Interface (POSIX) is a family of standards specified by the IEEE Computer Society for maintaining compatibility between operating systems.

S

S3 Amazon's Simple Storage Service: HTTP-based protocol to access data. Initially developed by Amazon, it generalisation made it a de facto standard for data access in cloud services.

SDLTS Simulation and Data Laboratory: Terrestrial Systems.

SSD Solid State Drive.

Swift Object-based interface of the OpenStack suite.

T

TCO Total Cost of Ownership.

TSMP Terrestrial System Modelling Platform is an open source scale-consistent, highly modular, massively parallel regional Earth system model.

Bibliography

[1] Estela Suarez, Norbert Eicker, and Thomas Lippert. *Modular Supercomputing Architecture: From Idea to Production*, pages 223–255. 05 2019.

[2] ParTec AG. ParaStation HealthChecker Administrator's Guide. https://docs.partec.com/html/pshealthcheck-adminguide/index.html, 2010.

[3] Jülich Supercomputing Centre. Llview — graphical monitoring of batch system controlled cluster. http://www.fz-juelich.de/jsc/llview, 2021.

[4] Seagate. Motr/cortx. https://github.com/Seagate/cortx-motr, 2021.

[5] Shawn Q Zheng, Eugene Palovcak, Jean-Paul Armache, Yifan Cheng, and David A Agard. Anisotropic correction of beam-induced motion for improved single-particle electron cryomicroscopy. *bioRxiv*, page 061960, 2016.

[6] Kai Zhang. Gctf: Real-time ctf determination and correction. *Journal of structural biology*, 193(1):1–12, 2016.

[7] J Kai Zhang. Gautomatch. https://www2.mrc-lmb.cam.ac.uk, 2017.

[8] Thorsten Wagner, Felipe Merino, Markus Stabrin, Toshio Moriya, Claudia Antoni, Amir Apelbaum, Philine Hagel, Oleg Sitsel, Tobias Raisch, Daniel Prumbaum, et al. Sphire-cryolo is a fast and accurate fully automated particle picker for cryo-em. *Communications biology*, 2(1):1–13, 2019.

[9] Tristan Bepler, Andrew Morin, Micah Rapp, Julia Brasch, Lawrence Shapiro, Alex J Noble, and Bonnie Berger. Positive-unlabeled convolutional neural networks for particle picking in cryo-electron micrographs. *Nature methods*, 16(11):1153–1160, 2019.

[10] Sjors HW Scheres. Relion: implementation of a bayesian approach to cryo-em structure determination. *Journal of structural biology*, 180(3):519–530, 2012.

[11] Sphire. http://sphire.mpg.de/wiki.

[12] Damien Lecarpentier, Peter Wittenburg, Willem Elbers, Alberto Michelini, Riam Kanso, Peter Coveney, and Rob Baxter. Eudat: a new cross-disciplinary data infrastructure for science. *International Journal of Digital Curation*, 8(1):279–287, 2013.

[13] Andrii Iudin, Paul K Korir, José Salavert-Torres, Gerard J Kleywegt, and Ardan Patwardhan. Empiar: a public archive for raw electron microscopy image data. *Nature methods*, 13(5):387–388, 2016.

[14] Michèle Weiland, Holger Brunst, Tiago Quintino, Nick Johnson, Olivier Iffrig, Simon Smart, Christian Herold, Antonino Bonanni, Adrian Jackson, and Mark Parsons. An early evaluation of intel's optane dc persistent memory module and its impact on high-performance scientific applications. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '19, New York, NY, USA, 2019. Association for Computing Machinery.

[15] ECMWF. ECMWF – Data Handling System. https://www.ecmwf.int/en/computing/our-facilities/data-handling-system, 2019.

[16] P. Shrestha, M. Sulis, M. Masbou, S. Kollet, and C. Simmer. A scale-consistent terrestrial systems modeling platform based on COSMO, CLM, and ParFlow. *Monthly Weather Review*, 142(9):3466–3483, sep 2014.

[17] Jessica Keune, Fabian Gasper, Klaus Goergen, Andreas Hense, Prabhakar Shrestha, Mauro Sulis, and Stefan Kollet. Studying the influence of groundwater representations on land surface-atmosphere feedbacks during the european heat wave in 2003. *Journal of Geophysical Research: Atmospheres*, 121(22):13,301–13,325, nov 2016.

[18] Carina Furusho-Percot, Klaus Goergen, Carl Hartick, Ketan Kulkarni, Jessica Keune, and Stefan Kollet. Pan-european groundwater to atmosphere terrestrial systems climatology from a physically consistent simulation. *Scientific Data*, 6(1), dec 2019.

[19] K. Goergen, A. Belleflamme, A. Ghasemi, C. Hartick, B. S. Naz, L. Poshyvailo, W. Sharples, N. Wagner, and S. J. Kollet. First results from a convection-permitting pan-European fully coupled TSMP simulation. In *AGU Fall Meeting Abstracts*, volume 2020, pages A096–0005, December 2020.

[20] Bibi S. Naz, Wolfgang Kurtz, Carsten Montzka, Wendy Sharples, Klaus Goergen, Jessica Keune, Huilin Gao, Anne Springer, Harrie-Jan Hendricks Franssen, and Stefan Kollet. Improving soil moisture and runoff simulations at 3 km over europe using land surface data assimilation. *Hydrology and Earth System Sciences*, 23(1):277–301, jan 2019.