IO-SEA introduction, concepts, and challenges

Philippe Couvée, ATOS R&D, IO-SEA Work Package 2 leader

This project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 955811. The JU receives support from the European Union's Horizon 2020 research and innovation programme and France, the Czech Republic, Germany, Ireland, Sweden, and the United Kingdom.

Introduction

Philippe COUVEE

ATOS Distinguished Expert

Working at ATOS HPC SW R&D since 20 years, leading the « Data Management » group with 15 engineers and PhD students

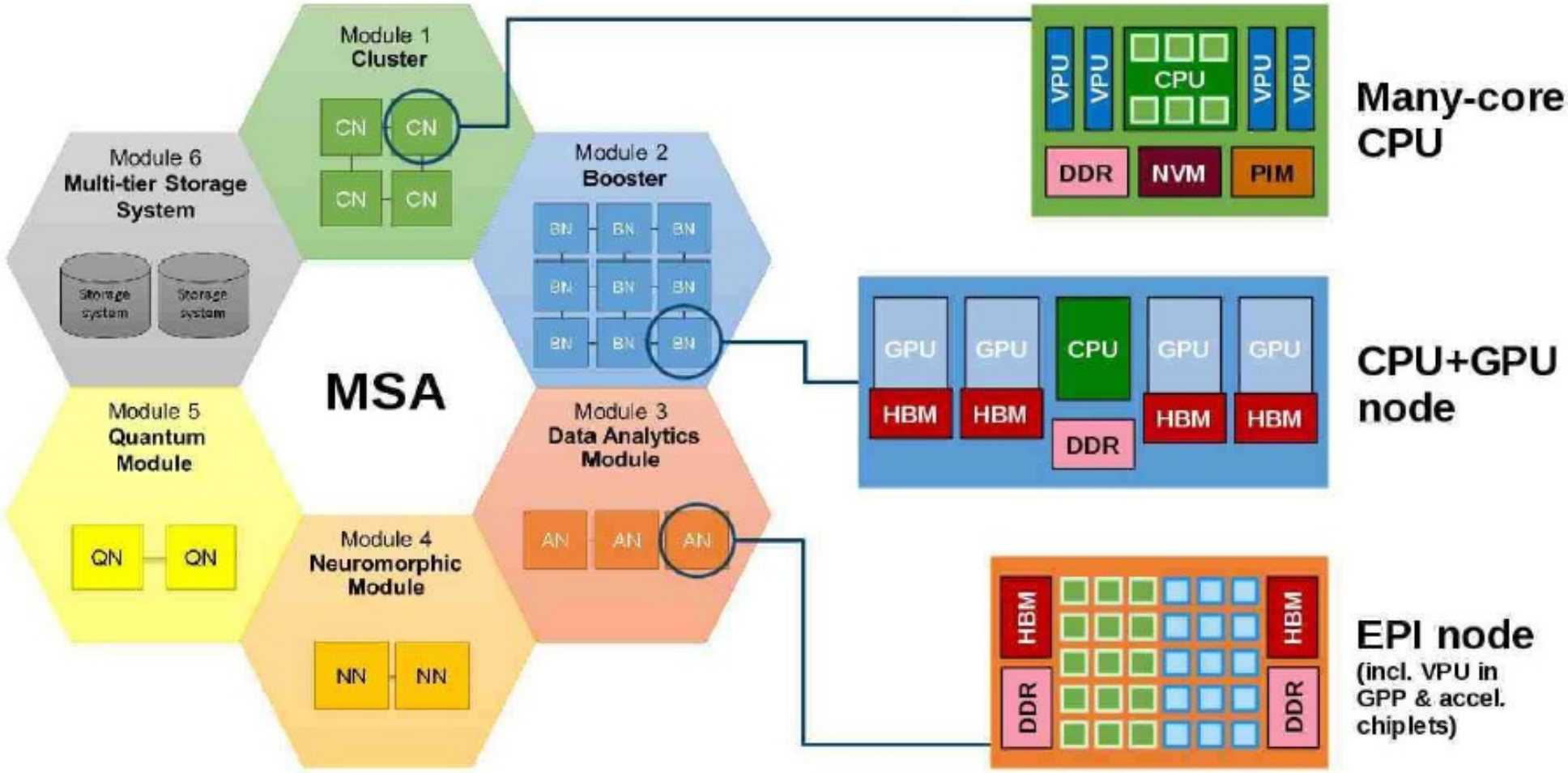
IO-SEA Work Package 2 Leader : https://iosea-project.eu/

IO-SEA

Context

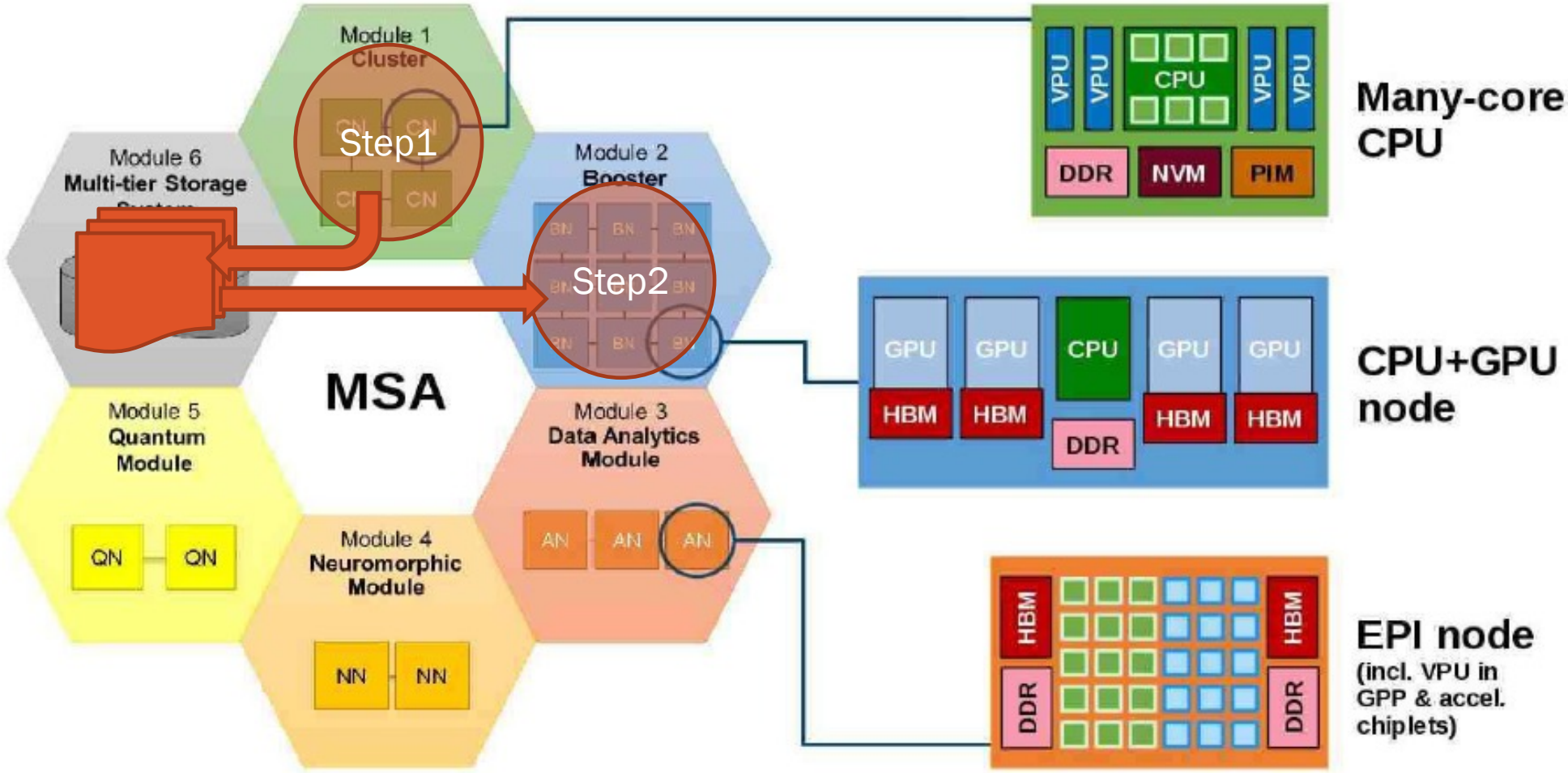
Modular SuperComputer Architecture

- Heterogeneous Compute modules
- « Long Term » Storage module with multiple technologies
- No cluster-wide interconnect
  - High speed Ethernet to connect modules



Modular SuperComputer Architecture

- Complex workflows running simultaneously and/or sequentially on multiple modules
  - Exchanging data through files

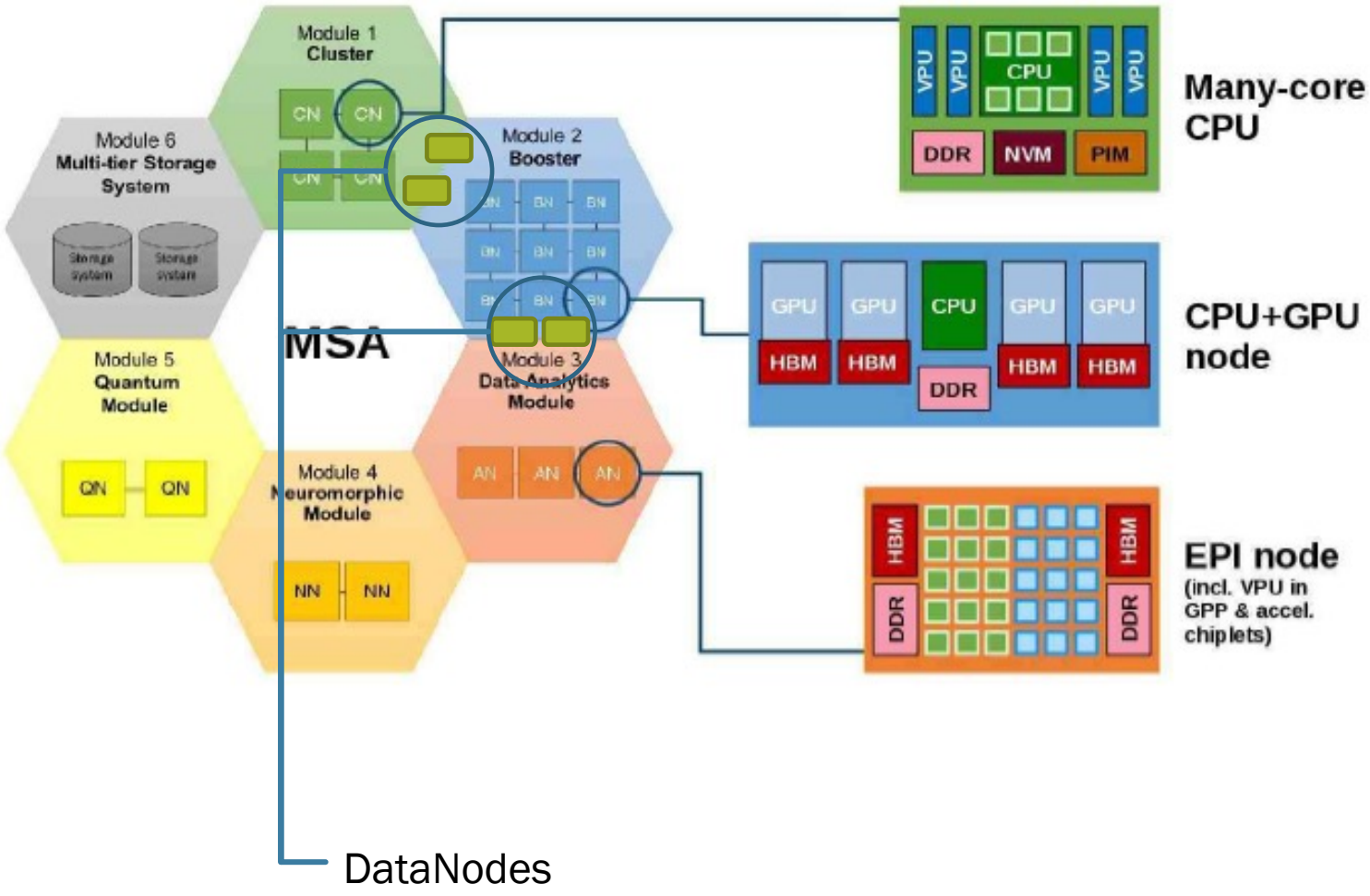


The IO-SEA Solution

IO-SEA Main Concepts 1: Data Nodes

Add **Data Nodes** between compute nodes and Long Term Storage Module

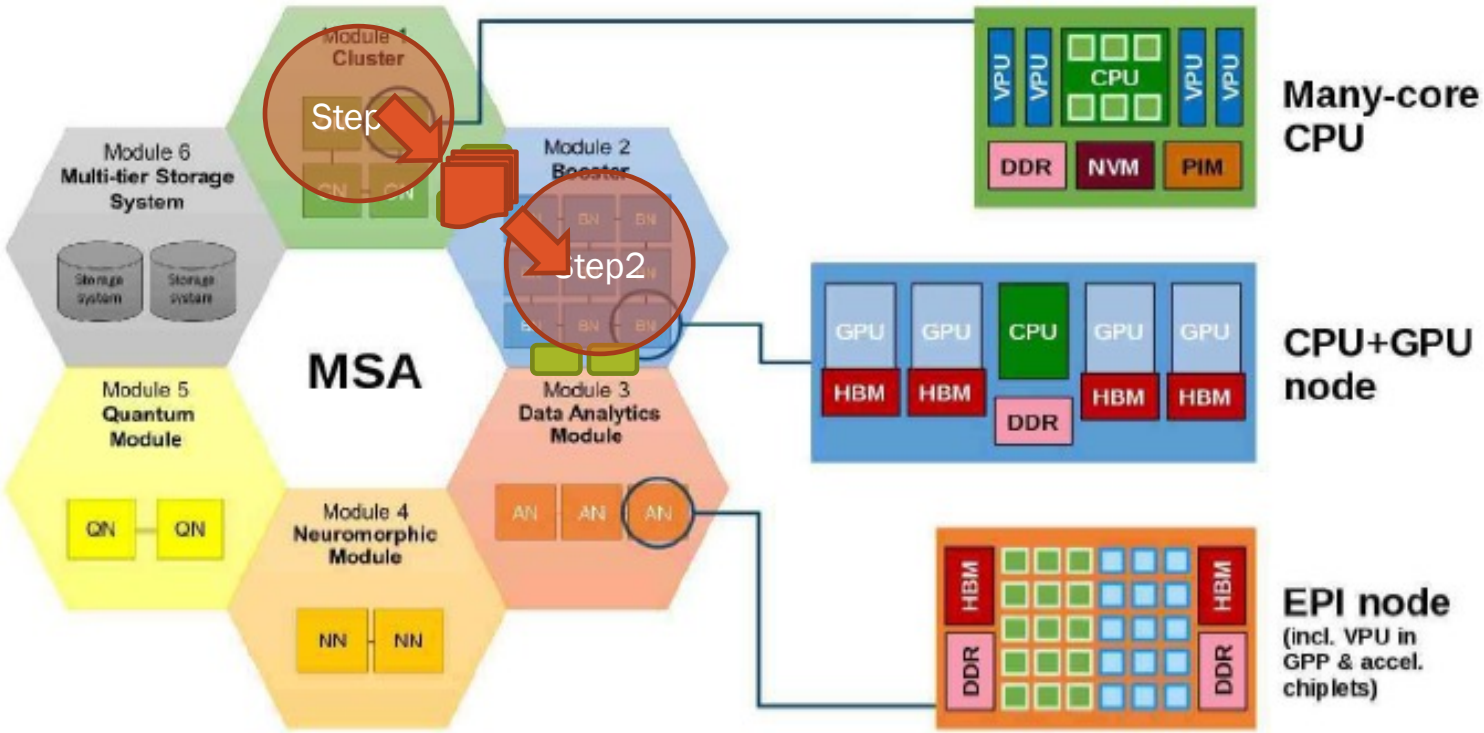
- connected to Compute Modules on their High Speed interconnects, and to the Long Term Storage Module
- Equipped with high speed storage devices
  - NVMe disks
  - NVRAM



IO-SEA Main Concepts 2: Workflows Scheduling

Consider **Workflows** rather than individual applications for scheduling

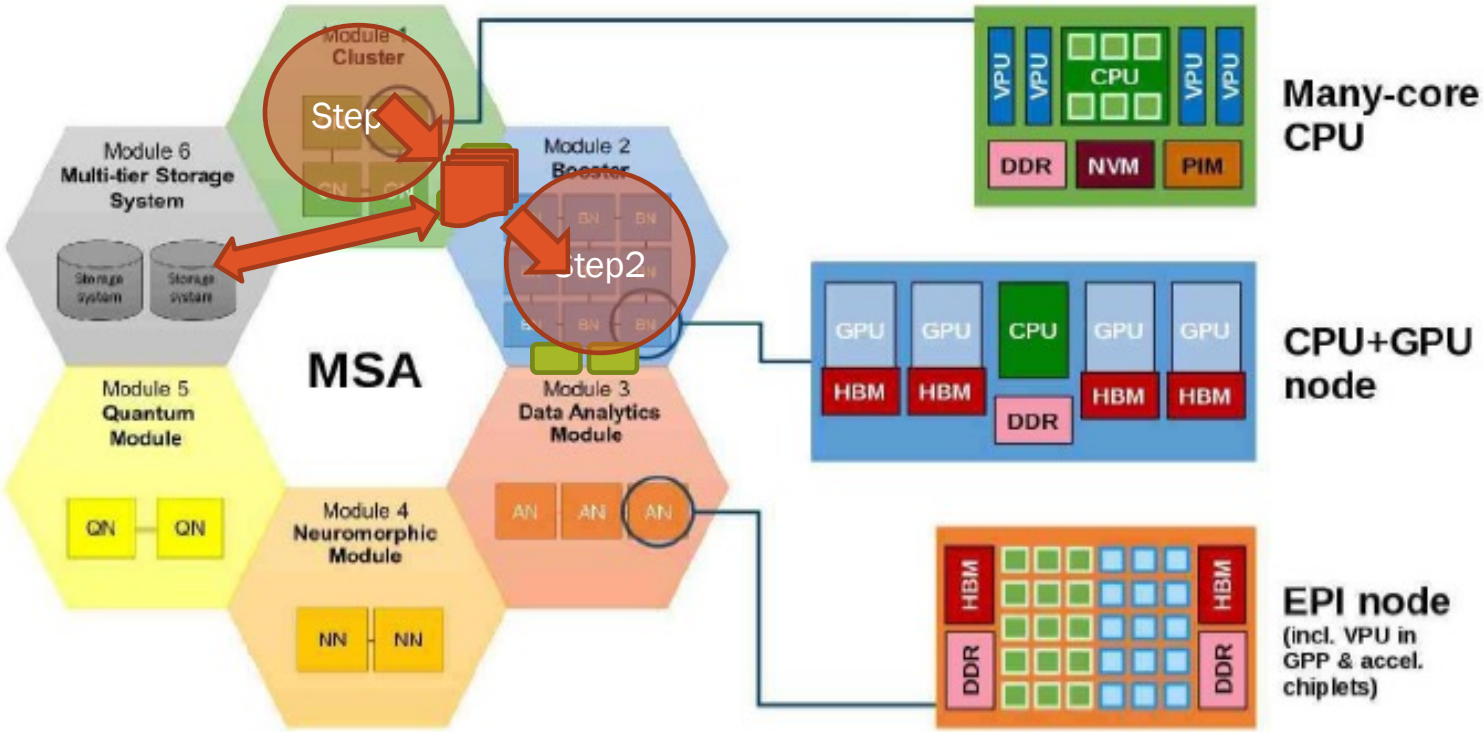
- Workflows composed of many applications (steps) running on different Modules
- Allocate data nodes resources in addition to Compute nodes



IO-SEA Main Concepts 3: Datasets & Ephemeral Services

Store Workflow data in **Datasets**

- Datasets are « data containers » stored in the **long term storage as « objects »,** exposed to compute nodes by **Ephemeral Services** running on Data Nodes



IO-SEA Stack

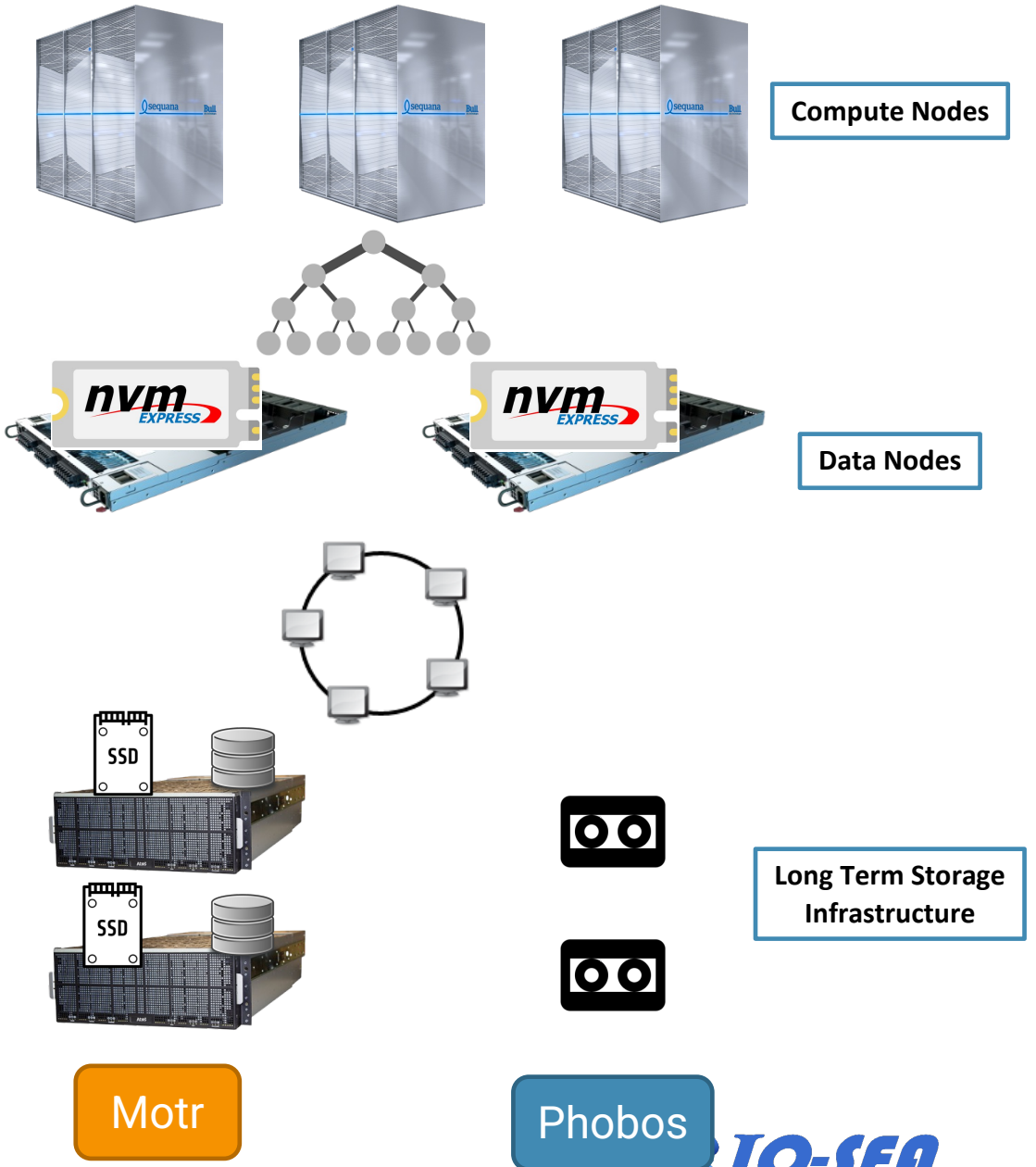
2 HW layers

- Data nodes equipped with high-speed storage devices (NVMe, NVRAM)

Ephemeral

- Long Term Storage composed of different tiers unified by an HSM software solution

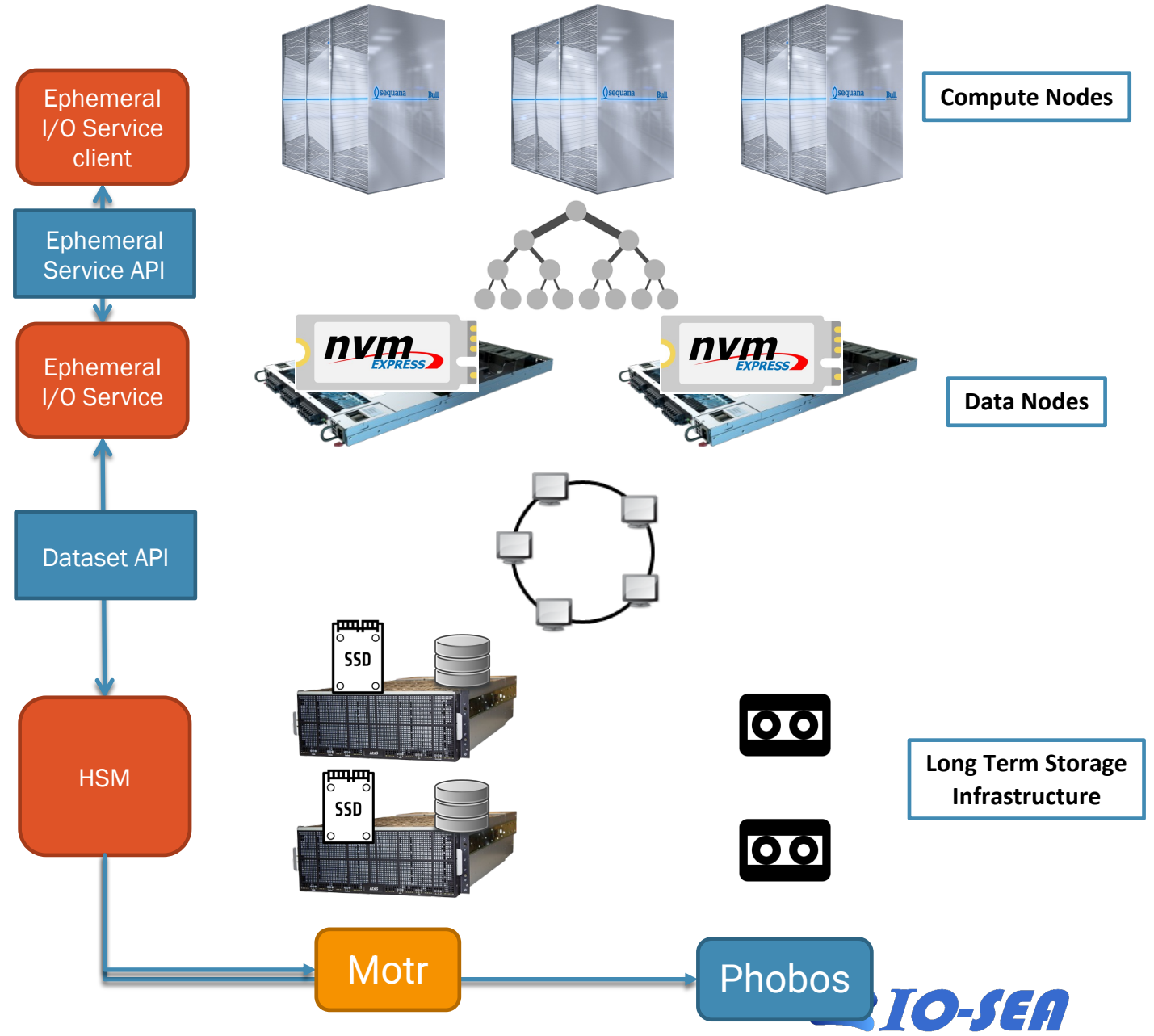
Long Term



IO-SEA Stack

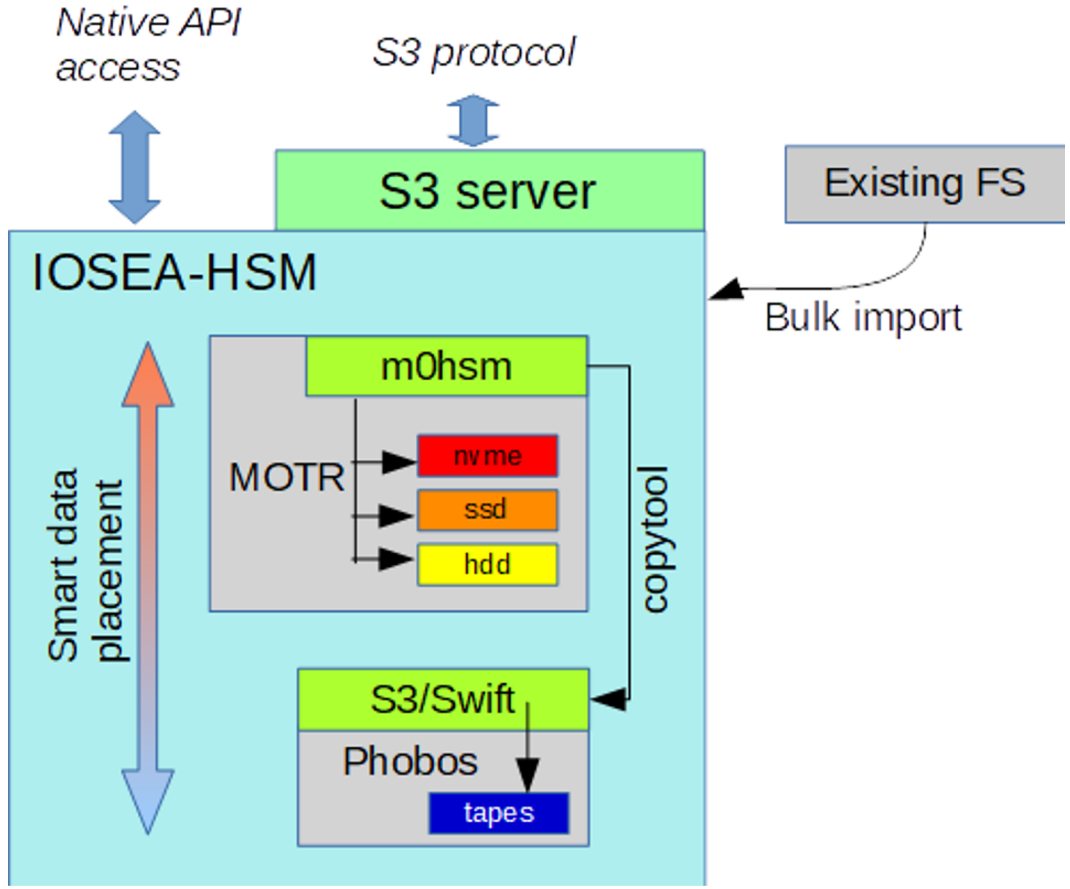
- Datanodes resources (cores, RAM, NVMe) are allocated for a **workflow** to run Ephemeral I/O services to give access to Datasets from compute nodes
  - POSIX, S3, DASI protocols

- Long Term Storage composed of different tiers unified by an HSM software solution
  - **Object Storage based**, no POSIX limitations



IO-SEA Long Term Storage : Hierarchical Storage

- Leverage a wide variety of **storage technologies** in a single system
  - from NVMe to tapes
- Implement **transparent** data migration between many storage tiers
- Smart **data placement** / **movements**:
  - ➢ Place data according to application needs
  - ➢ Gather needs from all running applications and develop a global data placement strategy
  - ➢ Prevent tiers from being fullfilled
  - ➢ Arbitrate conflicting requests / resource usage



DataSets & Namespaces

- Datasets are **data containers** hosting objects
  - Could be seen as private file systems or object stores
  - No data organization, just collections of objects…
- Objects in Datasets are organized with Namespaces
  - Different Namespaces can be created for each Dataset
  - Namespaces can expose the same dataset through different protocols
    - POSIX, S3

Datasets/Namespaces

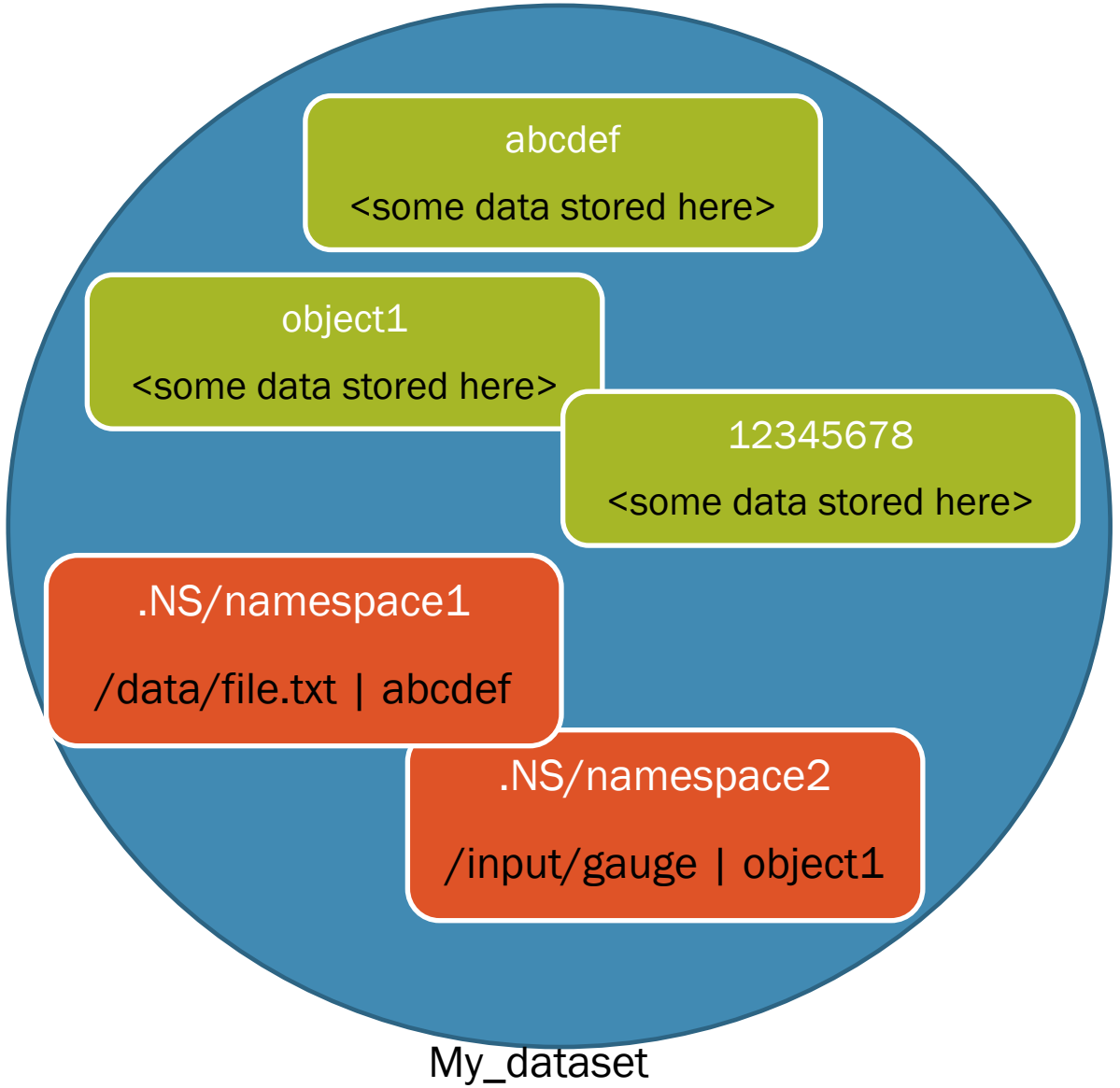
A Dataset is stored as 1 S3 Bucket

Files stored as S3 object

All namespaces stored in « metadata » objects

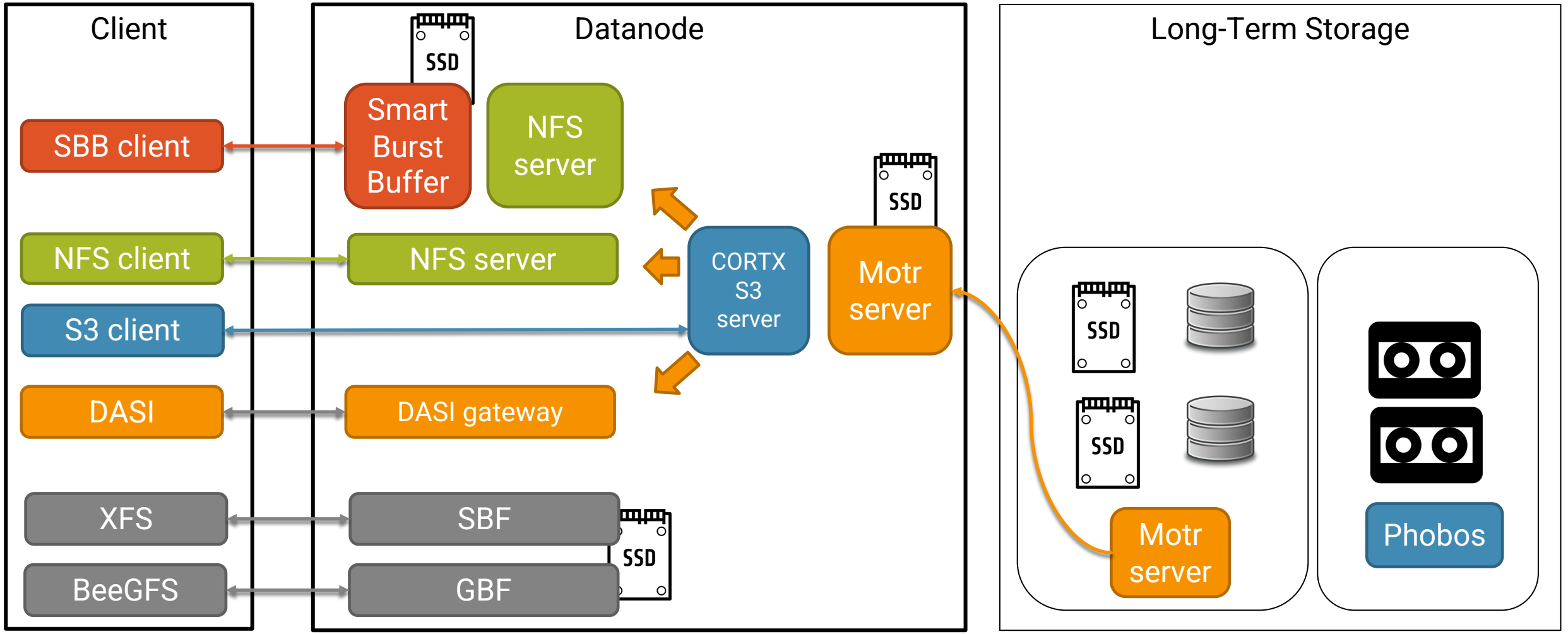
Think of it as a set of directories and symbolic links to objects in dataset

- .NS/namespace1
- .NS/namespace2



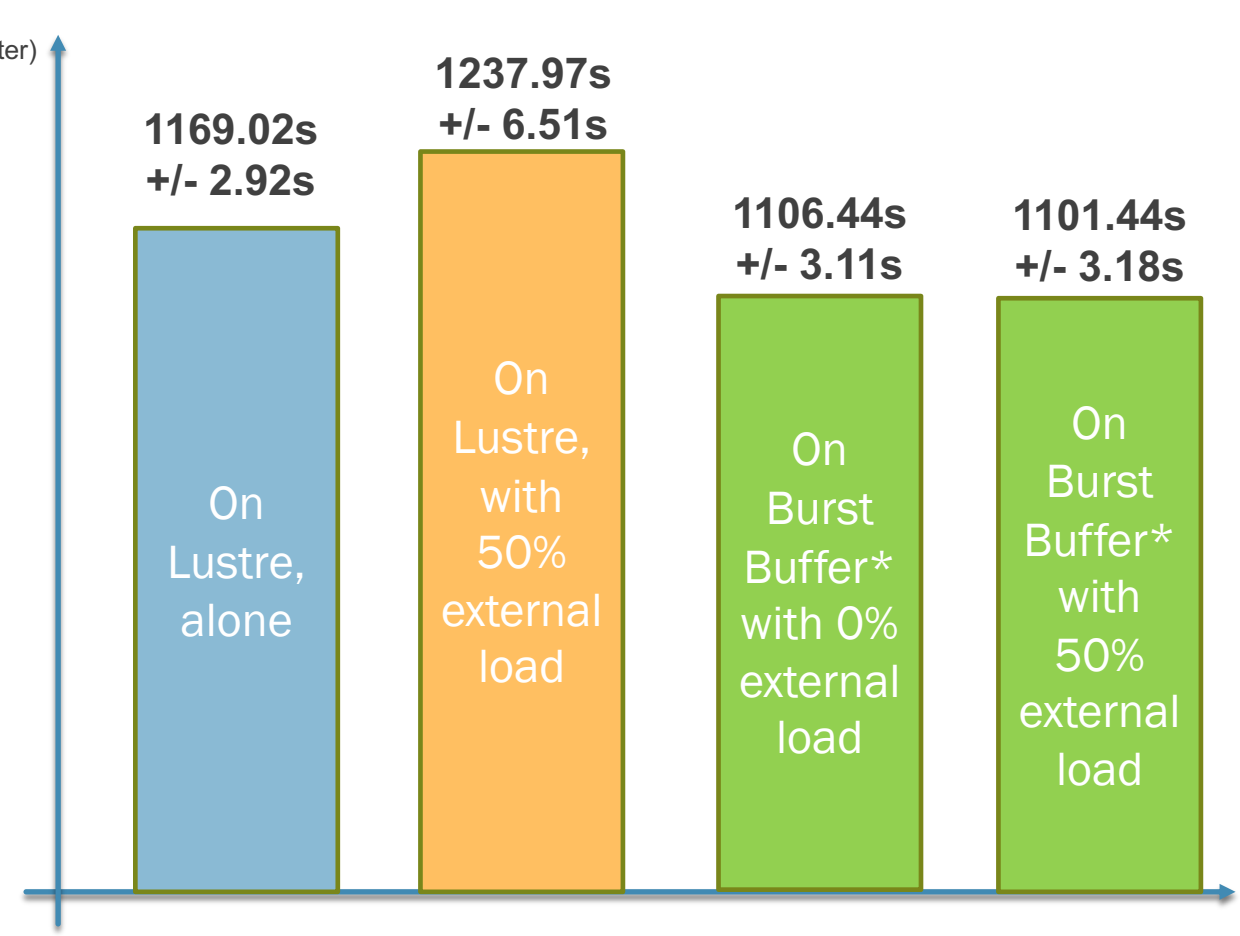
6 Ephemeral I/O Services

- **NFS, BB-NFS, S3, DASI, SBF, GBF**



Example : NAMD

- Filesystem: Lustre, max measured bandwidth ~ 2GBps
- 2 Datanodes, each 50GB RAM, 1.5T Flash
  200Gbps network, 16 allocated CPU cores
- NAMD:
  12 computes nodes (2x 16c/32t@3GHz)
  200Gbps network interconnect
  MPI mpich 3.4.2
  1 checkpoint each ~ 3 min (checkpoint size ~ 12GB)
  input: STMV_210M, size ~ 17GB uncompressed,
  source https://repository.prace-ri.eu/ueabs/NAMD/2.2/NAMD_TestCaseC.tar.gz

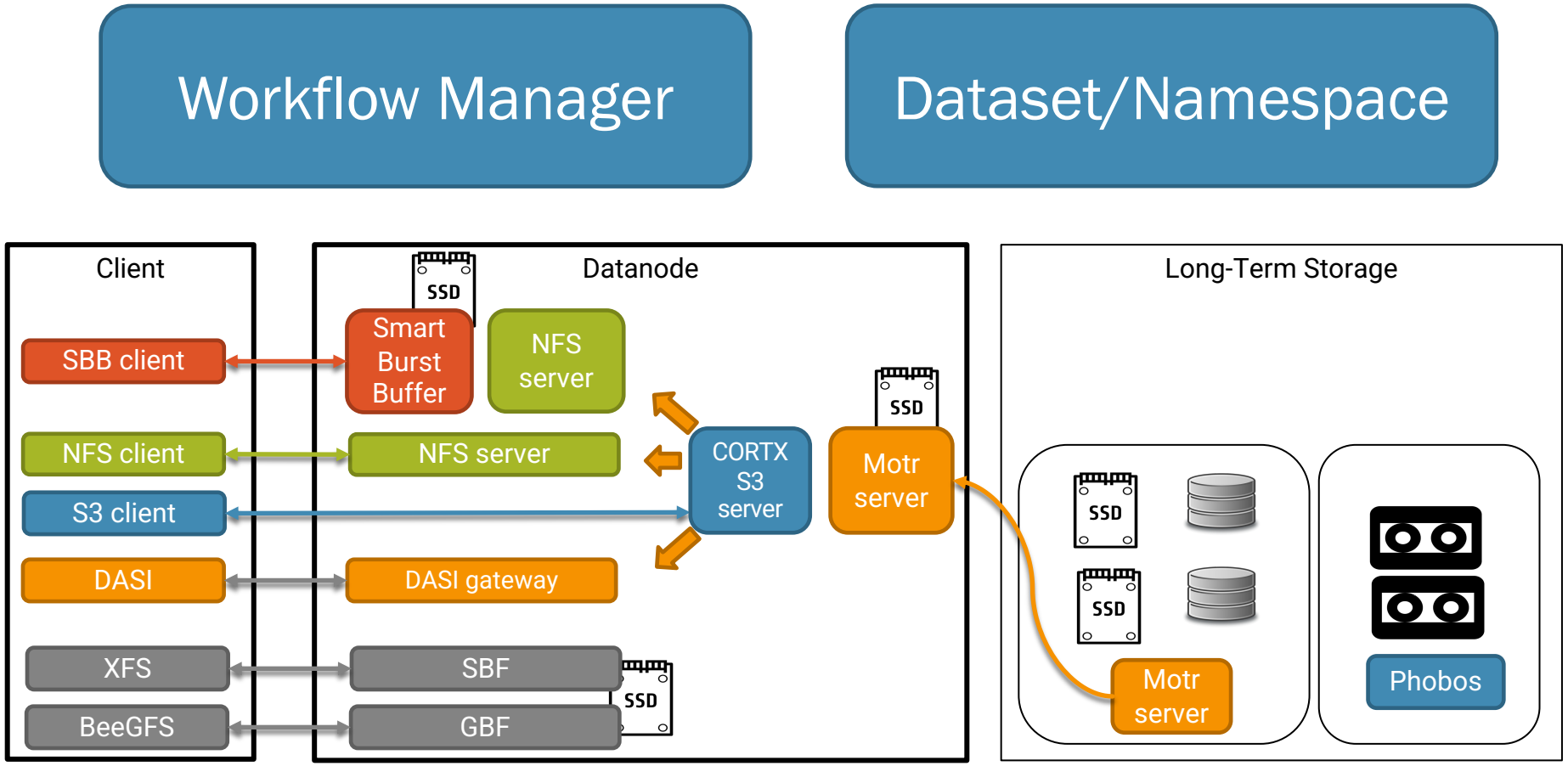


* : prefetch of input data into SBB not included in application runtime measurement

End User Interfaces & APIs

User Interfaces

- **Workflows are described in a « Workflow Description File » . yaml, and managed through command line tools**
- **Datasets/Namespaces are created with command line tools**



Workflow Sessions

- The steps processing the same data are run within a « **session** »
  - **Access tokens** protect against dataset access by multiple sessions in parallel
- Sessions have a user provided name
  - (UID, Session_Name) as identifier

```
# start a session for my workflow described in the WDF.yaml file
iosea-wf start WORKFLOW=WDF.yaml SESSION=My_Session

#run the workflow steps
iosea-wf run SESSION=My_Session STEP=step1
iosea-wf run SESSION=My_Session STEP=step2
iosea-wf run SESSION=My_Session STEP=step3

#stop the session and release the datanode
iosea-wf stop SESSION=My_Session
```

IO-SEA

Workflow Session Management

- **status** & **list** commands to report information
- **access** command to launch an interactive access environment
  - Slurm salloc to launch a shell in which Ephemeral Services clients will be configured
  - Will be « shareable » with team members

```
# display info about jobs & ephemeral services
iosea-wf status SESSION=My_Session

# list all active sessions of the user (report the session-names)
iosea-wf list

# Start an interactive access environment for all or limited to [<service>]
iosea-wf access SESSION=My_Session [NS=service]
```

Workflow Description File (WDF)

- **services** describe the ephemeral services needed to run the workflow

- **steps** describe how to configure the run time environment to run the steps

```
workflow:
  name:      My_Workflow

services:
  - name: ephemeral_service_1
    type: NFS
    attributes:
      namespace: dataset.My_namespace
      mountpoint: /mnt/USER/My_Workflow
      flavor: medium

steps:
  - name: step_A
    location:
      - gpu_module
    command: "srun My_Step_A"
    services:
      - name: ephemeral_service_1
```

Parametric Workflow Description File (WDF)

- To run multiple sessions in parallel, or to make the WDF more generic, variables can be used for most fields
- Variables must be defined « before use »

```
services:
  - name: ephemeral_service_1
    type: NFS
    attributes:
      namespace: {{ NS1 }}
      mountpoint: /mnt/USER/{{ SESSION }} My_Workflow
      flavor: medium
```

```
# start a session for my workflow
iosea-wf start WORKFLOW=WDF.yaml SESSION=My_Session NS1=My_namespace

#run the workflow steps
iosea-wf run SESSION=My_Session STEP=step1 NS1=My_other_namespace
```

Status command

- to monitor steps progress

```
> iosea-wf status SESSION=My_session
Workflow Name       : My_Workflow
Workflow SessionID : My_session_<timestamp>

Pending Steps
Step Name    Slurm ID     Tag      Command
step_B       2767         last     srun My_Step_B
SaveResults ----                   srun My_Copy_Script

Active Steps
Step Name    Slurm ID     Tag      Command
step_B       2763                  srun My_Step_B

Terminated Steps
Step Name    Slurm ID     Tag      Command
step_A       2760         init     srun My_Step_A

Ephemeral Services....
```

Location in the WDF

```
steps:
  - name: step_A
    location:
      - gpu_module
    command: "sbatch My_Step_A"
    services:
      - name: ephemeral_service_1
 - name: step_B
    location:
      - gpu_module
      - cpu_module
    command: "sbatch My_Step_B"
    services:
      - name: ephemeral_service_2
 - name: step_C
    location:
      - datanodes
    command: "sbatch My_Step_C"
    services:
      - name: ephemeral_service_2
```

- Steps indicate the « location » of their compute nodes
- Multiple locations are possible for complex jobs (MSA architecture)
- Location can be also « datanodes » for « on the fly » processing

IO-SEA

Data Movers in the WDF

```
services:
  - name: ephemeral_service_1
    type: NFS
    attributes:
      namespace: {{ NS1 }}
      mountpoint: /mnt/USER/{{ SESSION }}
      flavor: medium
    datamovers:
      - name: datamover1
        trigger: step_start
        target: flash
        operation: copy
        elements:
          - "gauges/*.hdf5"
          - "input/*"
```

- To ensure the elements of a namespace are located in the proper storage tier, a data_mover can be activated before (step_start) and after (step_stop) a step
- Operations are either move or copy
- Data Movers are defined at the service level, but activated per step

```
steps:
  - name: step_A
    location:
      - gpu_module
    command: "sbatch My_Step_A"
    services:
      - name: ephemeral_service_1
        datamovers:
          - datamover1
```

IO-SEA

Hints

"Hints" are optional information given by users about their future use of data, when the workflow is terminated

- **intended_access**: Intended access in the short term (will access, won't change…)
  e.g. intended_access="wont_change"

- **estimated_lifetime**: Estimated time until the object will be deleted (in seconds)
  e.g. estimated_lifetime=2592000 (1 month)

- **estimated_atime**: Estimated time the object will be used (in seconds)

- **access_period**: How often the object will be accessed (in seconds)
  e.g. access_period=60 (every minute)

- **predefined_policy**: Name of a pre-configured policy
  e.g. predefined_policy="temporary_data"

Setting hints

- User Control: CLI
  - `iosea-ns {locate|move|copy|release}`
  - `iosea-ns hints DATASET=My_dataset hint1=value1 hint2=value2 ...`

- User Control: through POSIX Ephemeral Services
  - `Set_attr on files and directories`

Datasets/Namespaces : Command line tools

```
# create namespaces
iosea-ns create my_dataset.my_namespace
iosea-ns create DATASET=my_dataset my_2nd_namespace
# fill my_namespace with a file
io-sea-ns put DATASET=my_dataset NAMESPACE=my_namespace ./gauge.cfg
```

- Enables leveraging standard POSIX tools and scripting

ls, find, …

Challenges

Challenges

- Need an enhanced Batch & Resource Manager
  - Workflow Aware
    - Allocating Datanode resources as part of the workflow
    - Sheduling ***workflows steps*** rather than ***isolated jobs***
      - To minimize data nodes resources allocation time
- Find the balance between compute nodes, data nodes and back end storage resources
  - New « dimensioning rules » to be invented