

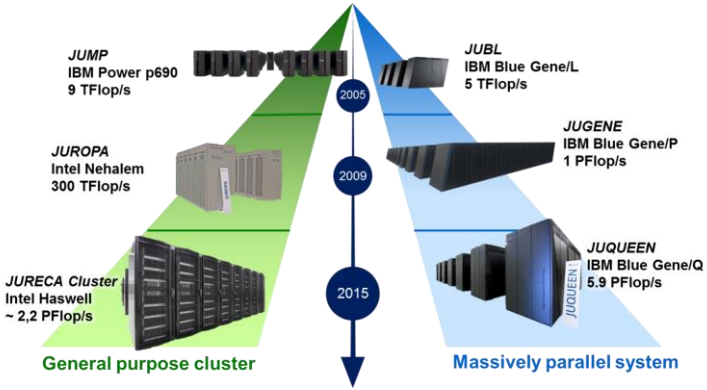
Extreme-scale I/O and Storage Infrastructures in Heterogeneous Modular Supercomputing Architectures

Sarah M. Neuwirth
Goethe-University Frankfurt, Germany
Jülich Supercomputing Centre, FZJ, Germany

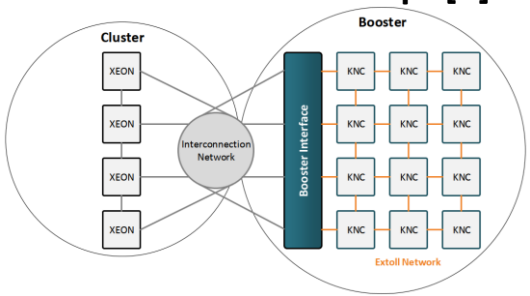
HPC-IODC Workshop, June 2022

Modular Supercomputing Architectures
A "Historical" Overview of a European Paradigm

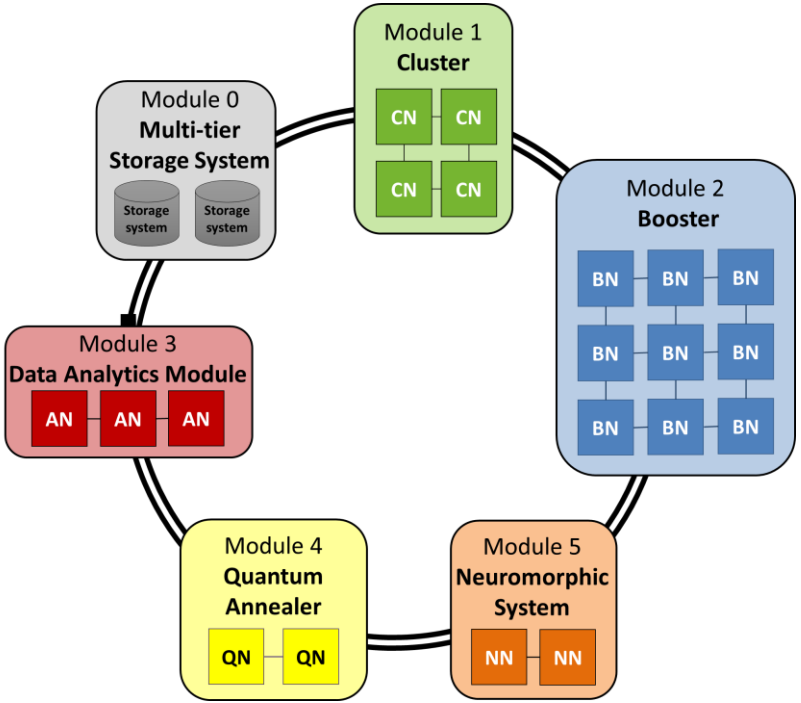
Dual Supercomputer Strategy [1]



Cluster-Booster Concept [2]



Modular Supercomputing Architecture [1]



Supercomputing Systems at JSC
JURECA – Phase 2 (as of May 2021) [3]

Data Centric (DC) module (98,304 CPU cores, 768 GPUs):

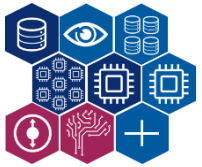
- 768 compute nodes (with 2× AMD EPYC 7742, 2× 64 cores, 2.25 GHz)
  - 480 standard compute nodes with 512 GB DDR4
  - 96 large-memory compute nodes with 1024 GB DDR4
  - 192 accelerated compute nodes with 512 GB DDR4, 4× NVIDIA A100 GPU, and 4× 40 GB HBM2e
- 12 login nodes with 2× AMD EPYC 7742, 2× 64 cores, 2.25 GHz, 1024 GB DDR4, and 2× NVIDIA Quadro RTX8000
- 3.54 (CPU) + 14.98 (GPU) Petaflop/s peak performance
- Mellanox InfiniBand HDR (HDR100/HDR) DragonFly+ network

Booster module (111,520 CPU cores):

- 1640 compute nodes with 1× Intel Xeon Phi 7250-F Knights Landing, 68 cores, 1.4 GHz and 96 GB memory plus 16 GB MCDRAM high-bandwidth memory
- 5 Petaflop/s peak performance
- Intel Omni-Path Architecture high-speed network with non-blocking fat tree topology

Supercomputing Systems at JSC

JUWELS: Jülich Wizard for European Leadership Science [4]



Cluster Module (122,768 CPU cores):

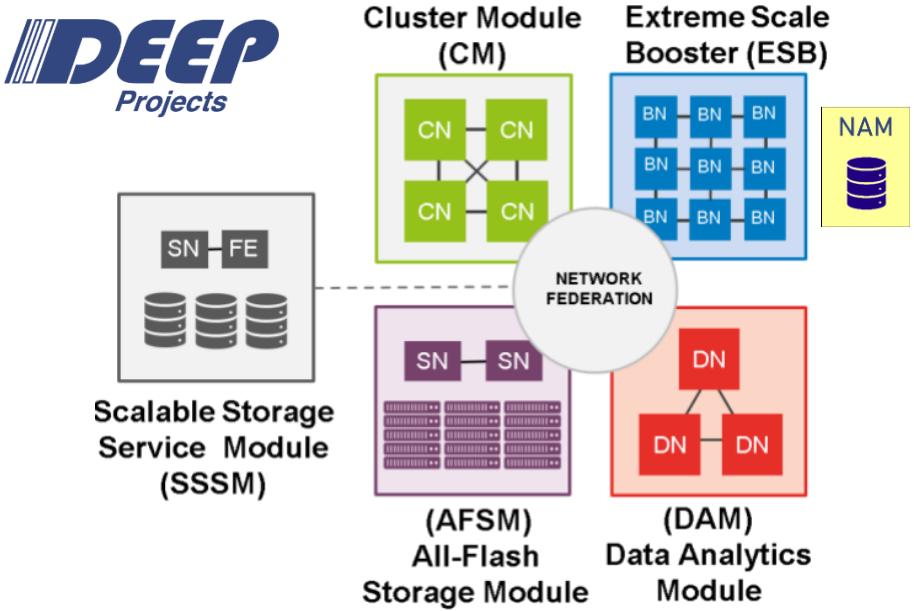
- *All nodes:* 2× Intel Xeon Platinum 8168 CPU, 2× 24 cores, 2.7 GHz
- 2271 standard compute nodes, 96 GB DDR4
- 240 large memory compute nodes, 192 GB DDR4
- 56 accelerated compute nodes: 192 GB DDR4, 4× NVIDIA V100 GPU, 16 GB HBM
- 12 login nodes: 64 GB DDR4, 2× 1TB HDD (RAID 1)
- 4 visualization nodes: 768 GB DDR4 2x 1TB HDD (RAID 1), 1× NVIDIA Pascal P100
- 10.6 (CPU) + 1.7 (GPU) Petaflop/s peak perf.
- InfiniBand EDR fat-tree network with 2:1 pruning at leaf level and top-level HDR switches

Booster Module (3,744 GPUs):

- 936 compute nodes: 2× AMD EPYC Rome 7402 CPU, 2× 24 cores, 2.8 GHz, 512 GB DDR4, 4× NVIDIA A100 GPU, 4× 40 GB HBM2e
- 4 login nodes: 2x 24 cores, 2.7 GHz, 12x 16 GB, 2666 MHz
- 73 Petaflop/s peak performance
- Mellanox InfiniBand HDR DragonFly+ topology with 20 cells - 40 Tb/s connection to Cluster

Supercomputing Systems at JSC
DEEP-EST Prototype

Modular Supercomputing prototype developed within the DEEP-EST project [5, 6].



System architecture from the DEEP system, implementing the Modular Supercomputing Architecture (MSA) [5, 6].

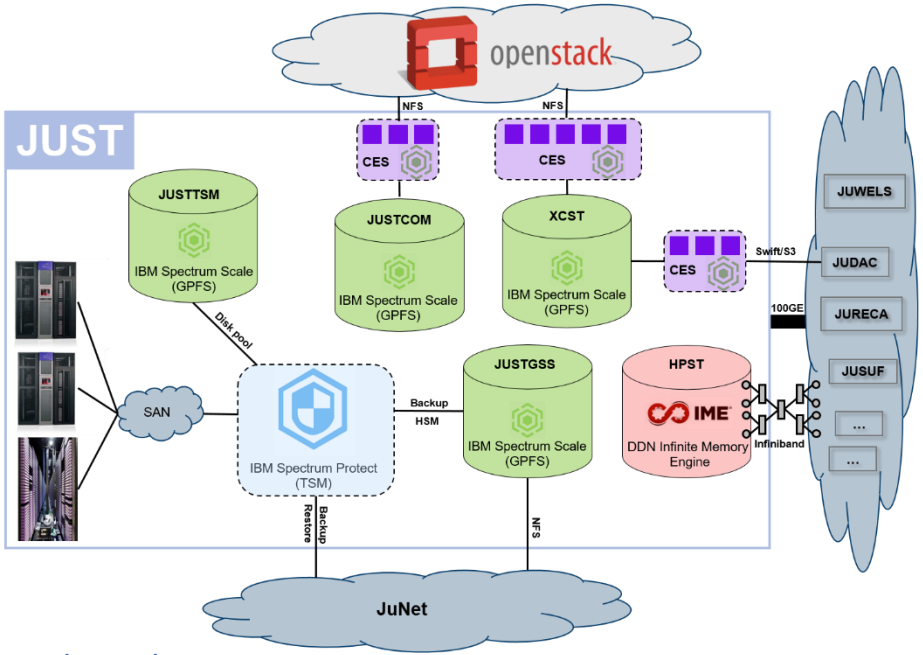
Supercomputing Systems at JSC
Juelich Storage Cluster (JUST)

Juelich Storage Cluster (JUST) [7]:

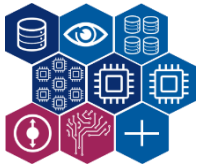
- GPFS Storage Server (DSS/DATA)
- End-to-End integrity
- Fast rebuild time on disk replacement
- GPFS + TSM Backup + HSM

JUST5-DSS [7]:

- Capacity: 75 PB gross
- Hardware: Lenovo Distributed Storage Solution
- Building blocks:
  - 21x DSS-G 24: each 2x Lenovo x3650 M5, 334 NL-SAS Disks and 2 SSDs
  - 1x DSS-G 26: each 2x Lenovo x3650 M5 Systems, 502 NL-SAS Disks and 2 SSDs



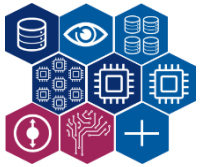
DEEP-EST Prototype
Main Hardware Features



DEEP System	Cluster Module	Booster Module	Data Analytics Module
Usage and design target	Applications and code requiring high single-thread performance and a modest amount of memory. => *typically moderate scalability*	Compute intensive applications and code with regular control and data structures. => *high parallel scalability*	Data-intensive analytics and machine learning applications and code requiring large memory capacity, data streaming, bit- or small datatype processing.
Node Count	50	75	16
CPU Typ **CPU Codename** **Cores @frequency**	Intel Xeon 6146 Skylake 12 @3.2GHz	Intel Xeon 4215 Cascade Lake 8 @2.5GHz	Intel Xeon 8260M Cascade Lake 24 @2.4GHz
Accelerators / node	n.a.	1x NVIDIA V100 GPU	1x NVIDIA V100 GPU 1x Intel Stratix10 FPGA

DEEP-EST Prototype

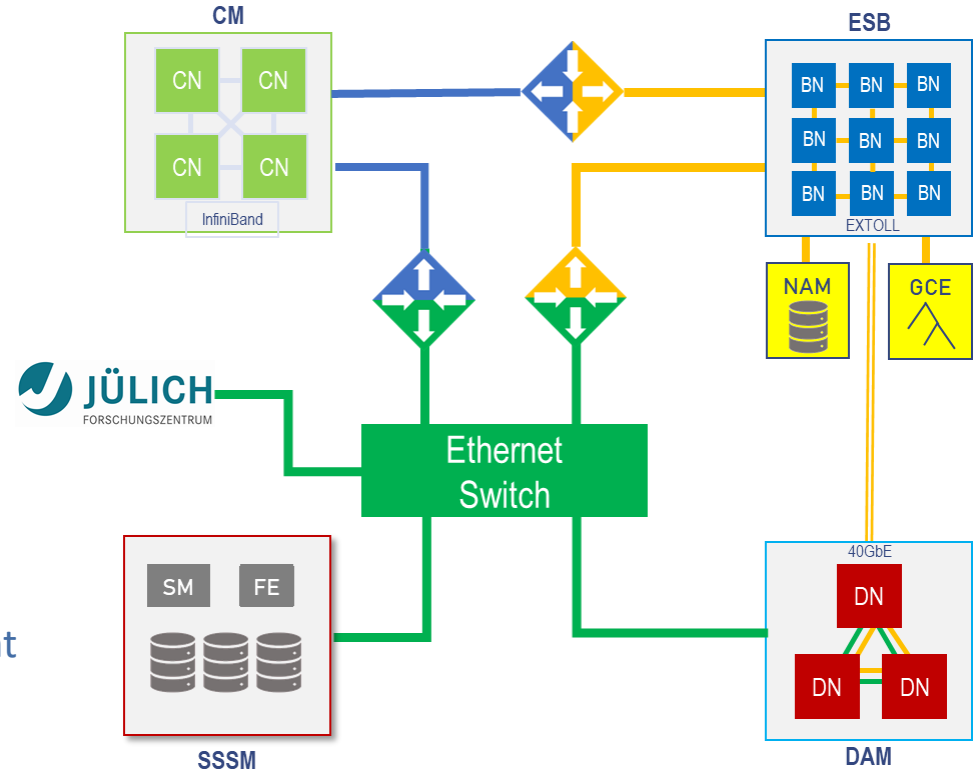
Main Hardware Features cont.



DEEP System	Cluster Module	Booster Module	Data Analytics Module
DDR4 capacity	192 GB	48 GB	384GB+32GB(FPGA)
HBM capacity	n.a.	32 GB (GPU)	32 GB (GPU)
NVMe	n.a.	n.a.	3 TB Intel Optane
Node max. mem BW	256 GB/s	900 GB/s	900 GB/s (GPU)
Storage	1x 512 GB NVMe SSD	1x 512 GB NVMe SSD	2x 1.5 TB NVMe SSD
Network Technology	EDR-IB (100 Gb/s)	EDR-IB (100 Gb/s)	EDR-IB (100 Gb/s) Ethernet (40 Gb/s)
Network Topology	Fat-tree	Tree	Tree
Power / node	500 W	500 W	1600 W
Cooling	Warm-water	Warm-water	Air
Integration	1× Rack MEGWARE SlideSX-LC ColdCon	3× Rack MEGWARE SlideSX-LC ColdCon	1× Rack MEGWARE

DEEP-EST Prototype
Complex Network Federation

- IB-to-EXTOLL Bridge
  - MPI
- IB-to-Ethernet Bridge
  - MPI
- EXTOLL-to-Ethernet Bridge
  - Linux functionality
  - MPI
  - GCE = Global Collective Engine
- Important Features
  - 3 bi-directional network gateways sufficient
  - Accommodating key requirement for fast NAM access by dual fabrics in the DAM



DEEP-EST Prototype
Memory Technologies

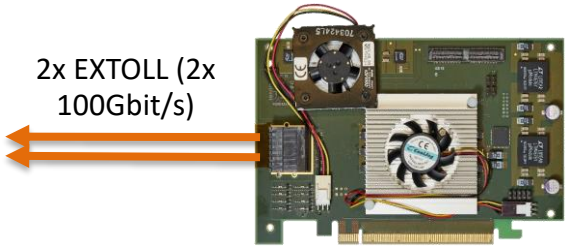
Non-Volatile Memory (NVM)

DC P3700
(Source: Intel)

- First gen Intel NVM device
  - 20nm MLC NAND Flash technology
  - PCIe gen3 ×4 lanes
- IOZONE results vs. SATA 6G SSD
  - Block accesses: Read 5×, write 2.5×
  - Random accesses: Read 2×, write 3×

Network-Attached Memory (NAM) [8]

2x EXTOLL (2x 100Gbit/s)

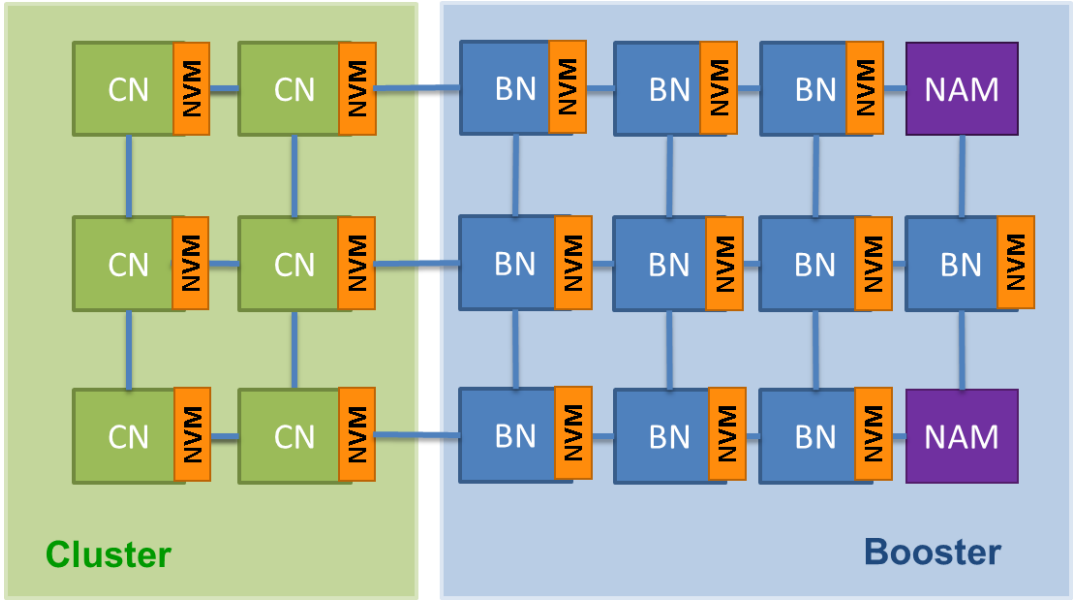


NAM
(Source: Extoll)

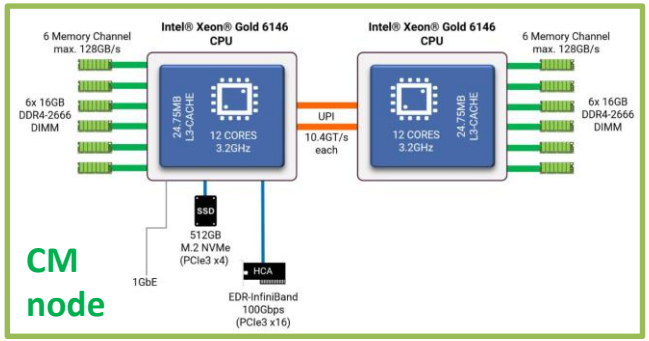
- Architecture:
  - Xilinx Virtex 7 FPGA
  - Memory: hybrid memory cube (HMC)
  - EXTOLL fabric (with 2 links)
- Functionality
  - RDMA functionality across EXTOLL
  - Checkpoint/restart logic

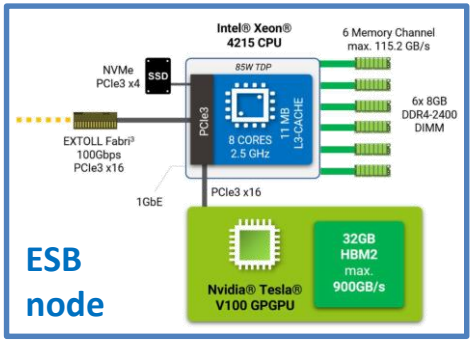
DEEP-EST Prototype
Multi-level Memory Hierarchy Configuration

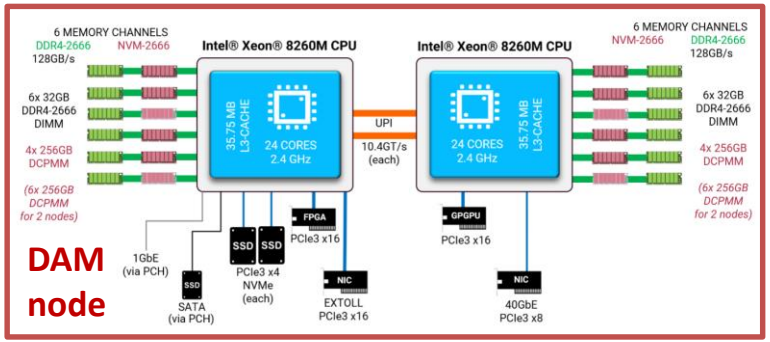
- **CN:** Xeon (Haswell)
- **BN:** Xeon Phi (KNL) => self booting
- **NVM:** Non-Volatile Memory
- **NAM:** Network Attached Memory



DEEP-EST Prototype
Hardware Modules – Architectural Overview



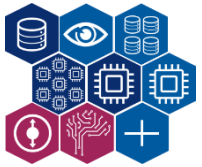




Storage Environment:

- Permanent storage provided through *JUST storage system* (GPFS)
- Shared fast storage on *Scalable Storage and Service Module (SSSM)* – total of 304 TB of storage managed by BeeGFS
- Local ext3/ext4 file systems (hosted on CM, DAM, and ESB nodes)
- Local storage – BeeOND
- *All-Flash Storage Module (AFSM)* – 1.8 PB of data storage capacity based on BeeGFS

DEEP-EST Prototype
Storage Environment: SSSM and AFSM



Scalable Storage and Service Module (SSSM):

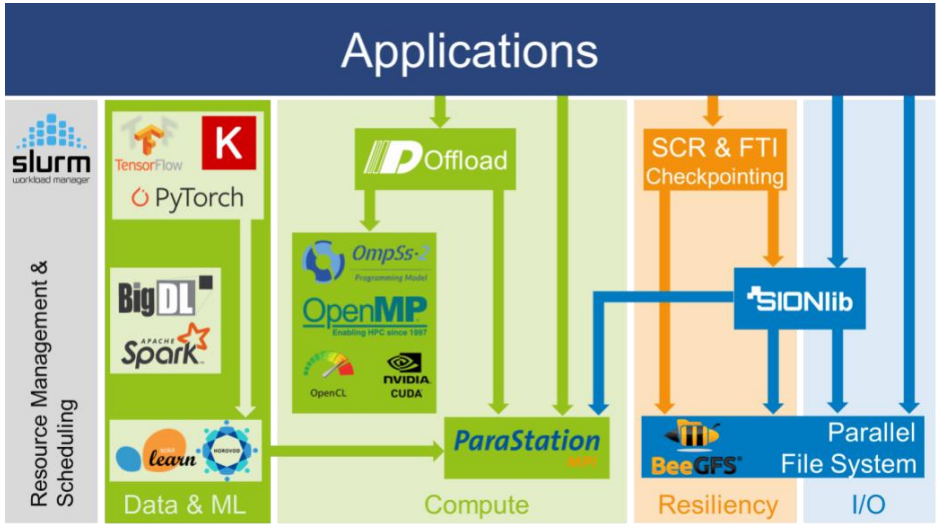
- Hosts a total of 304 TB of storage (spinning disks) managed by the BeeGFS parallel file system
- Data is stored in two RAID arrays with 24 disks each (=> RAID6 storage scheme)
- 4 file system data servers provide access, through BeeGFS clients on CM, DAM, and ESB
- Located under `/work` in the file system tree => standard POSIX interface
- Connected to the system using 40 Gbit/s Ethernet technology => data passed via IP gateways
- Temporary storage device mainly to serve data required by applications run on the DEEP-EST system

All-Flash Storage Module (AFSM):

- Complements the SSSM
- Based on modern PCIe3 NVMe SSD storage devices
- BeeGFS global parallel file system is used to make 1.8 PB of data storage capacity available
- 2 metadata servers and 6 volume data server systems, which are interconnected by a 100 Gbps EDR-InfiniBand fabric
- Integrated into the DEEP-EST EDR fabric topology of the CM, ESB and DAM

DEEP-EST Prototype
Software Environment

- **Low-level SW:** Inter-network bridging
- **Scheduler:** Torque/Maui -> SLURM
- **Filesystem:** BeeGFS
- **Compilers:** Intel, gcc, PGI
- **Debuggers:** Intel Inspector, TotalView
- **Programming:** ParaStation MPI (mpich), OpenMP, OmpSs
- **Performance analysis tools:** Scalasca, Extrae/Paraver, Intel Advisor, VTune…
- **Benchmarking tools:** JUBE
- **Libraries:** SIONlib, SCR, HDF5…



ParaStation MPI

JUBE BENCHMARKING ENVIRONMENT

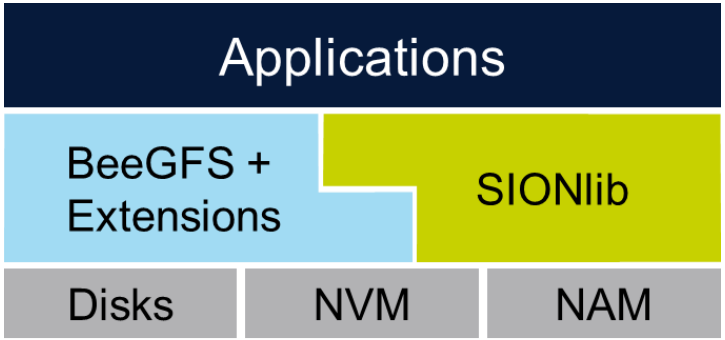
slurm

SIONlib

scalasca

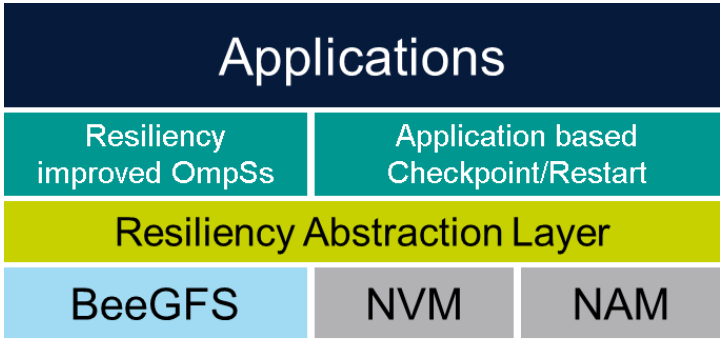
DEEP-EST Prototype
Parallel I/O and Resiliency

I/O Software Architecture:



- **BeeGFS** (parallel FS)
- **SIONlib** (I/O concentrator)

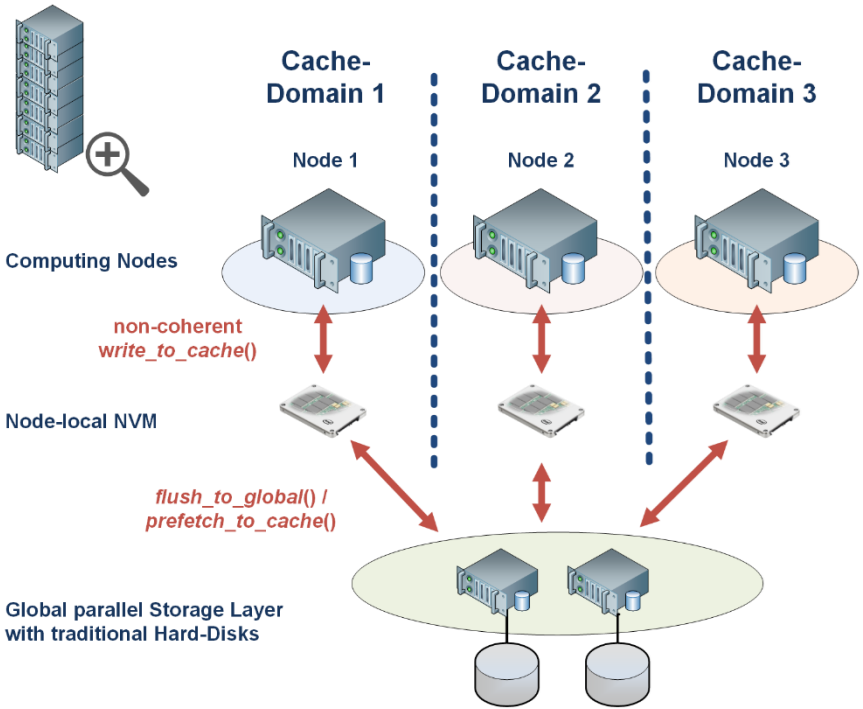
Resiliency Software Architecture:



- **SCR** (checkpointing handling)
- **ParaStation MPI** (process CP)
- **OmpSs** (task checkpointing)

=> *Combination of SW packages provides new functionality and exploits HW*

DEEP-EST Prototype
Scalable I/O via BeeGFS Filesystem



- ***Two instances:***
  - Global FS on HDD server
  - Cache FS on NVM at node
- ***API:*** cache domain handling
  - Synchronous version
  - Asynchronous version

BeeGFS® developed by Fraunhofer

DEEP-EST Prototype

BeeGFS Extension to support MSA

(1) BeeGFS monitoring

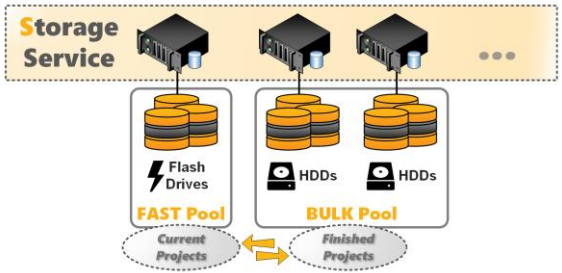
- Adapted and optimized BeeGFS monitoring with a time series DB for a better integration with overall DEEP-EST monitoring system

(2) BeeOND storage plugin

- Support for non-POSIX backend to the BeeGFS storage server and integrate persistent memory devices (NVRAM) as such backend

(3) BeeGFS installation on SSSM and AFSM

- BeeGFS installation used by all nodes as scratch filesystem



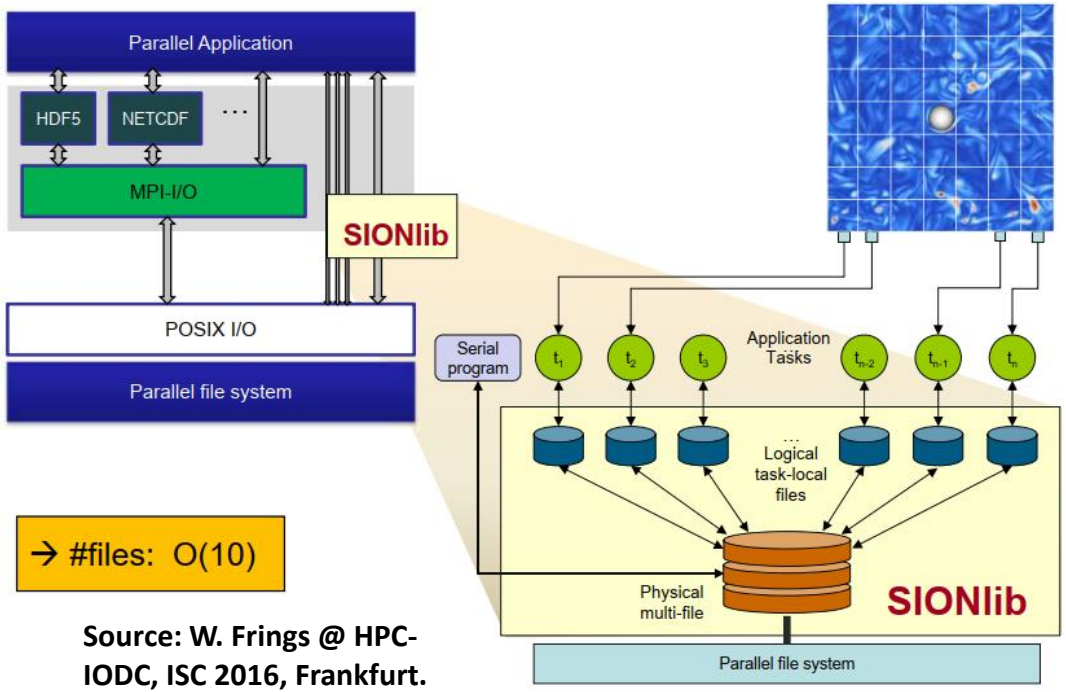
DEEP-EST Prototype
SIONlib: Shared Files for Task-local Data

- Extension of I/O-API (ANSI C or POSIX)
- C and Fortran bindings, implementation language C
- Current versions: 1.7.7
- Open source license: https://www.fz-juelich.de/en/ias/jsc/services/user-support/jsc-software-tools/sionlib

```
/* fopen() → */
sid=sion_paropen_mpi( filename , "bw",
                      &numfiles, &chunksize,
                      gcom, &lcom, &fileptr, ...);

/* fwrite(bindata,1,nbytes, fileptr) → */
sion_fwrite(bindata,1,nbytes, sid);

/* fclose() → */
sion_parclose_mpi(sid)
```



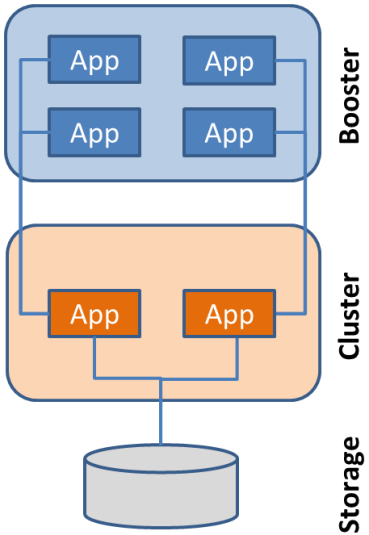
→ #files: O(10)

Source: W. Frings @ HPC-IODC, ISC 2016, Frankfurt.

DEEP-EST Prototype
SIONlib: MSA-aware Extensions

(1) MSA-aware collective I/O

- Collective I/O allows processes to cooperate on storage accesses
- “Collector“ processes perform operations on behalf of other processes
- MSA aware collective I/O allows transferring I/O duties to application tasks running on modules suitable for I/O



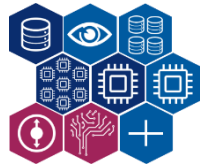
(2) CUDA-aware interface

- Read and write functions accept pointers to on-device buffers as arguments, similar to CUDA aware MPI => simplify I/O in ESB nodes

(3) I/O forwarding

- I/O forwarding allows transferring I/O duties to dedicated non-application I/O proxy tasks running on modules suitable for I/O

DEEP-EST Prototype
NAM Slurm Plugin for Resiliency Support

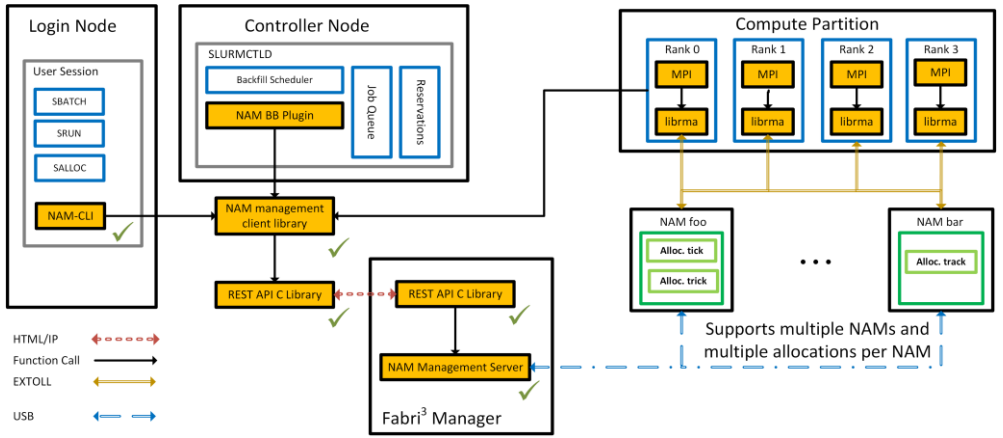


Functionality:

- Supports multiple NAMs and allocations, transient & persistent allocations
- User can create or request NAM allocations
- User can delete existing persistent allocations

Implementation:

- Based on Slurm burst buffer plugin
- Uses NAM management client library
- Passes known list of NAM allocations to PS-MPI layer

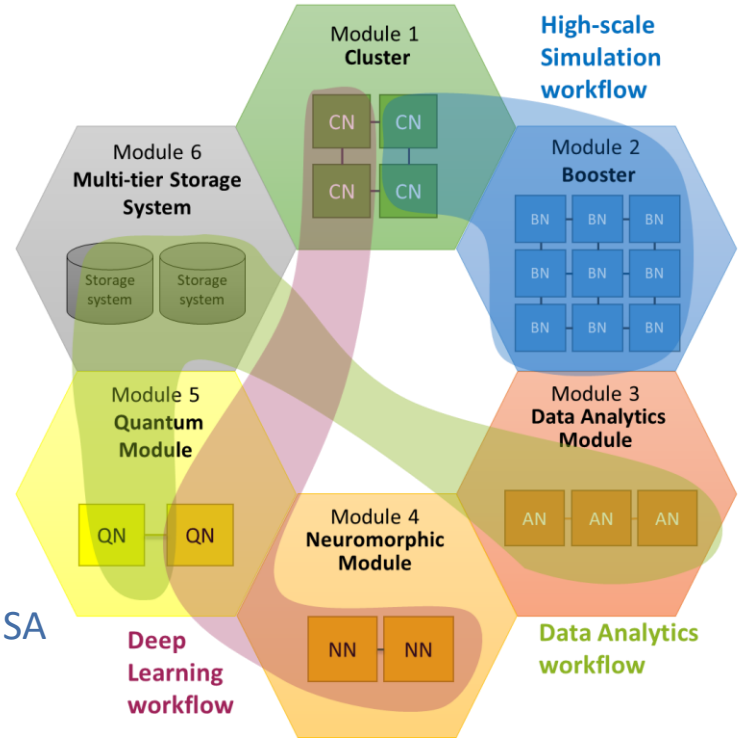



```
#!/bin/bash
#SBATCH --job-name=A-NAM-Job                    # Job name
#SBATCH --nodes=16                              # Run on 16 nodes
#NAM name=Hugo size=1024M                       # Transient NAM allocation
#NAM use_persistent name=Herbert                # Use existing persistent allocation
#NAM create_persistent name=Hans size=16M       # Create a persistent allocation
```

DEEP-EST Prototype

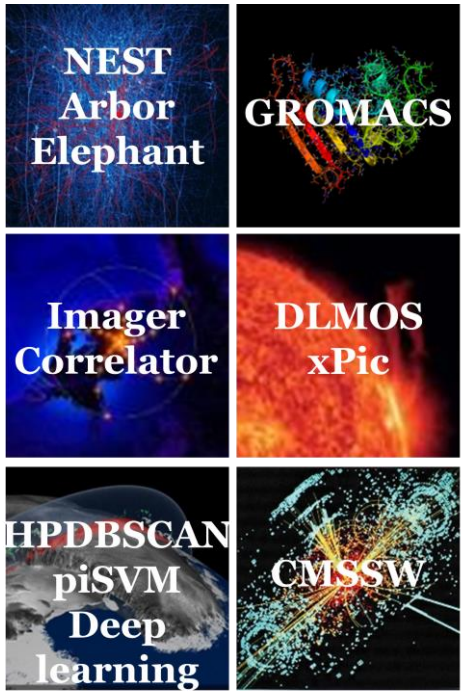
Mapping Applications to Modular Supercomputers

- Cost-efficient scaling
- Effective resource-sharing
- Fit application diversity
  - Large-scale simulations
  - Data analytics
  - Machine- and Deep Learning
- Composability of heterogeneous resources
- Co-design recipe and feedback loop:
  - Gather the requirements from applications
  - Create HW relevant specific application use-cases
  - Identify how to best match the co-design use cases to MSA
  - Assess the MSA as an architecture
  - Adapting the Loop

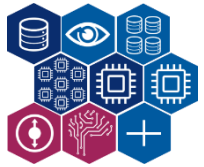


DEEP-EST Prototype
Set of Six Applications Ported [5]

- Neuroscience
  - NEST: Simulation of point-like neurons
  - Arbor: Simulation of detailed neurons
  - Elephant: Analysis of electrophysiological experiments
- Molecular Dynamics – the most widely used code
- Radio Astronomy
  - LOFAR correlator and imager
  - SKA – most important astronomy project to come
- Space Weather
  - CBA-application with modular extension: xPic
- Data Analytics in Earth Science
  - Clustering of big data by PiSVM
- CERN: High Energy Physics
  - Reconstruction workflows on GPUs, FPGAs



Challenges and Future Directions
Overview



- **System scalability:** future supercomputers may include hundreds of thousands of nodes and data is to be accessed by ~10^6 clients. Traditional parallel file systems cannot operate efficiently at this scale. => *access to data becomes a critical issue*
- **Data scalability:** bigger machines mean more data, which means more records or files. => *I/O systems should be able to store hundreds of exabytes or even zettabytes*
- **Data heterogeneity:** Small vs. large files, sequential vs. random access, access patterns (single vs. concurrent).
  => *quite complex "data taxonomy" needs to be supported*
- **Data placement:** To use complex supercomputing architectures such as MSA best, data must be used and produced as close as possible to the place where the simulation code runs.

IO-SEA

DEEP-SEA

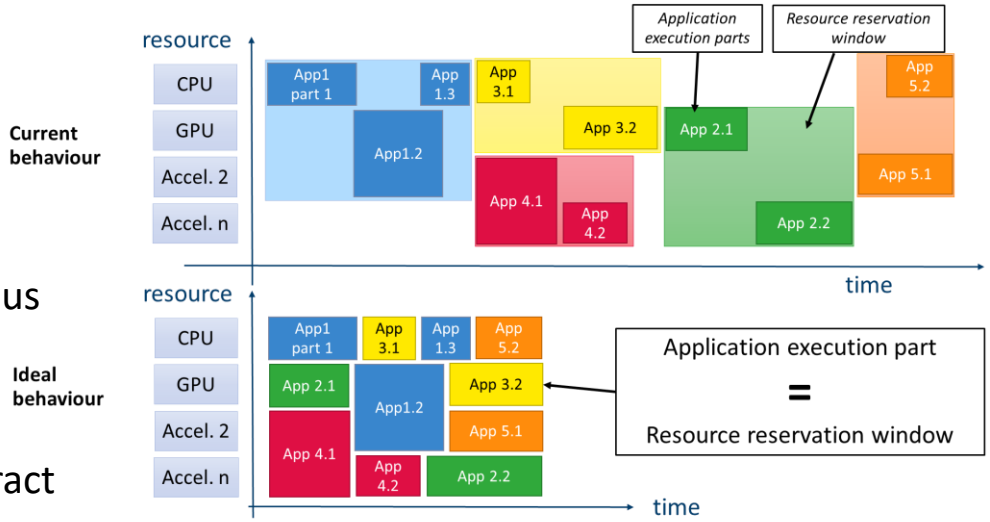
RED-SEA

EUPEX
European Pilot for Exascale

epi
European Processor Initiative

Challenges and Future Directions
DEEP-SEA Project

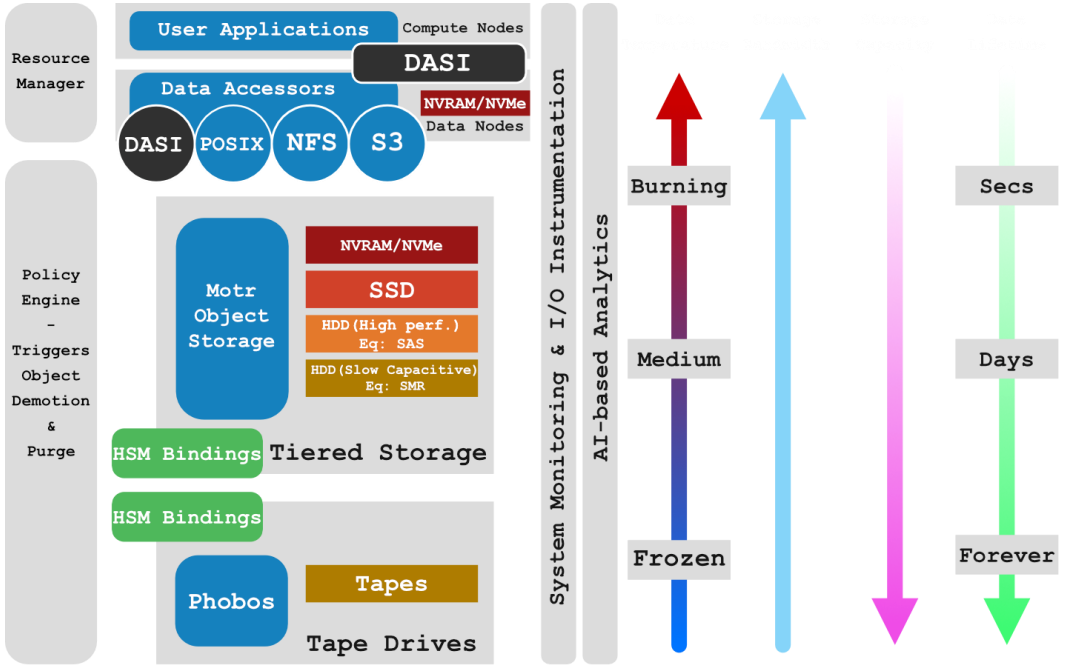
- Co-design the software- and programming environment of the upcoming European exascale systems.
- Provide tools to map complex applications and non-uniform workflows onto heterogeneous and modular computer architectures.
- Enhance the system software, programming paradigms, tools, and runtimes in order to extract the maximum performance from heterogeneous computer platforms and improve performance portability.
- Improve the use and management of new memory technologies and the placement of data in compute devices with deep and heterogeneous memory hierarchies.



https://www.deep-projects.eu/

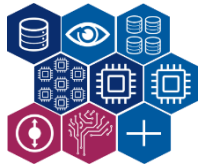
Challenges and Future Directions
IO-SEA Project

- ***Data workflow:*** Significantly improve workflow execution by allowing users / applications to tag data and therefore to add information about its future usage as well as of the usage of resulting data.
- ***Instrumentation:*** Extend existing tools to identify the data lifecycle and set up an optimal data-crentic I/O runtime env.
- ***Data Access and Storage application interface (DASI)*** => *more later today* ☺
- ***Hierarchical storage management:*** HSM mechanisms are rare in object stores. IO-SEA targets to use NVMe devices, HDD, SSD and tapes inside the same tier.



https://iosea-project.eu/

References and Further Reading



[1] E. Suarez, N. Eicker, Th. Lippert (2019). *Modular Supercomputing Architecture: from Idea to Production.* In J. Vetter (Editor), Contemporary High Performance Computing: From Petascale toward Exascale, Volume 3, Chapter 9, pp. 223-251. CRC Press. Available online: http://hdl.handle.net/2128/22212

[2] S. Neuwirth (2018). *Accelerating Network Communication and I/O in Scientific High Performance Computing Environments* (Doctoral dissertation). https://doi.org/10.11588/heidok.00025757

[3] JURECA User Documentation: https://apps.fz-juelich.de/jsc/hps/jureca/index.html

[4] JUWELS User Documentation: https://apps.fz-juelich.de/jsc/hps/juwels/index.html

[5] A. Kreuzer, Th. Lippert, E. Suarez, and N. Eicker (2021). *Porting Applications to a Modular Supercomputer – Experiences from the DEEP-EST Project.* http://hdl.handle.net/2128/30498.

[6] DEEP Prototype System: https://www.fz-juelich.de/en/ias/jsc/systems/prototype-systems/deep_system

[7] JUST User Documentation: https://apps.fz-juelich.de/jsc/hps/just/index.html

[8] J. Schmidt (2017). *Accelerating Checkpoint/Restart Application Performance in Large-Scale Systems with Network Attached Memory* (Doctoral dissertation). https://doi.org/10.11588/heidok.00023800

Questions?