I/O and Scheduling aspects in DEEP-EST

Norbert Eicker

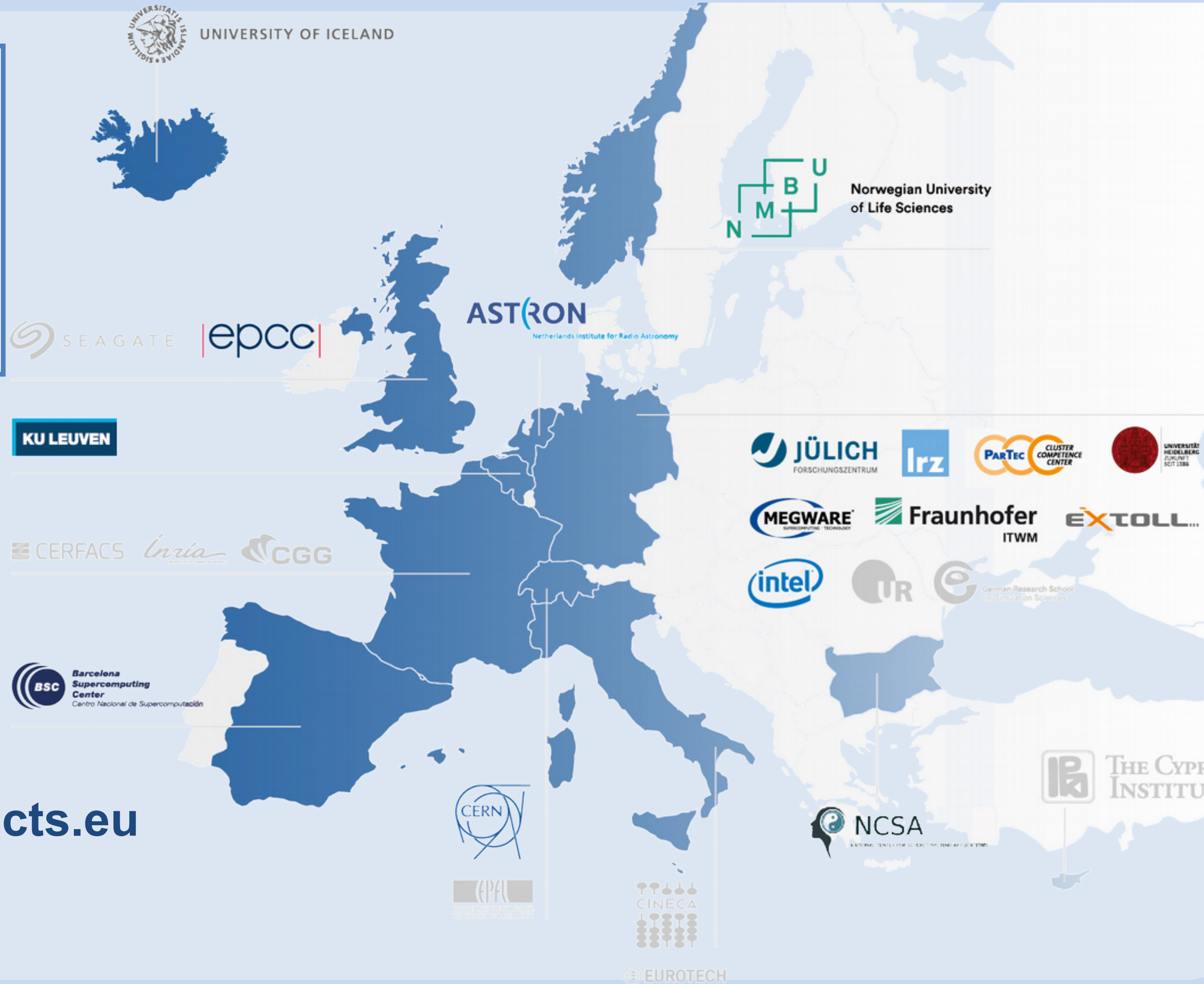
Jülich Supercomputing Centre & University of Wuppertal

26 September 2017

The research leading to these results has received funding from the European Community's Seventh Framework Programme (*FP7/2007-2013*) under *Grant Agreement* n° 287530 and n° 610476

Collaborative R&D in DEEP projects
(DEEP + DEEP-ER + DEEP-EST)

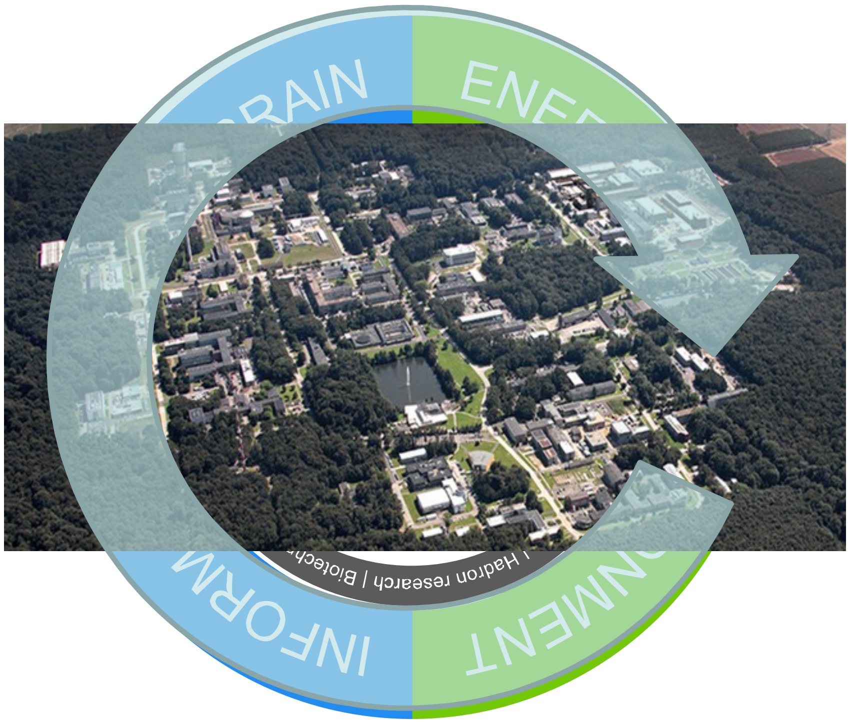
EU-Exascale projects
27 partners
Total budget: 44 M€
EU-funding: 30 M€
Nov 2011 – Jun 2020



www.deep-projects.eu

JÜLICH
FORSCHUNGSZENTRUM

Science Campus Jülich



5,700 staff members

Budget (2015): 558 mio. €

- Institutional funding: **320 mio. €**
- Third party funding: **238 mio. €**

Project management: 1,6 billion €

ptJ Projektträger Jülich Forschungszentrum Jülich

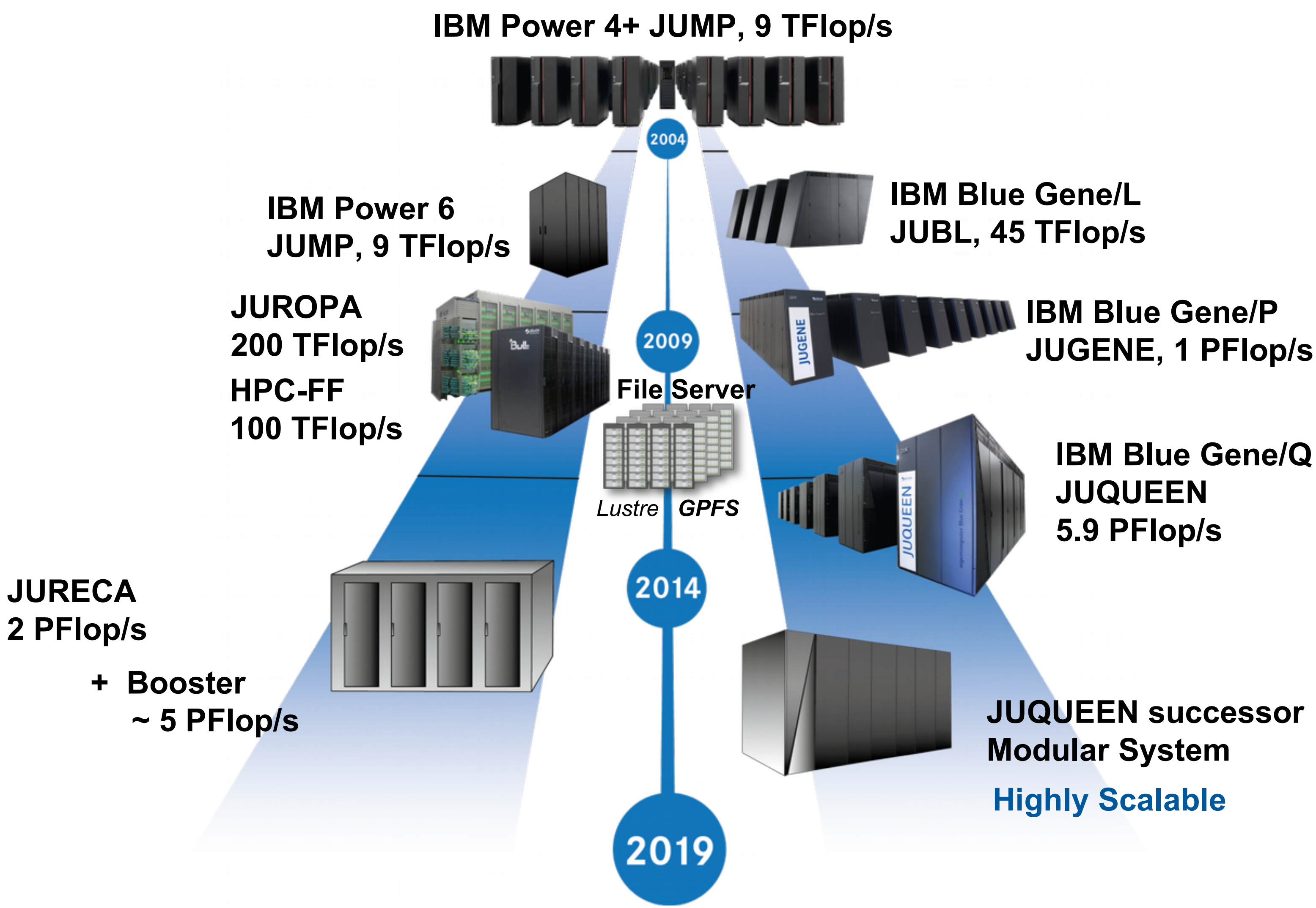
ETN Projektträger Energie • Technologie • Nachhaltigkeit

Teaching:

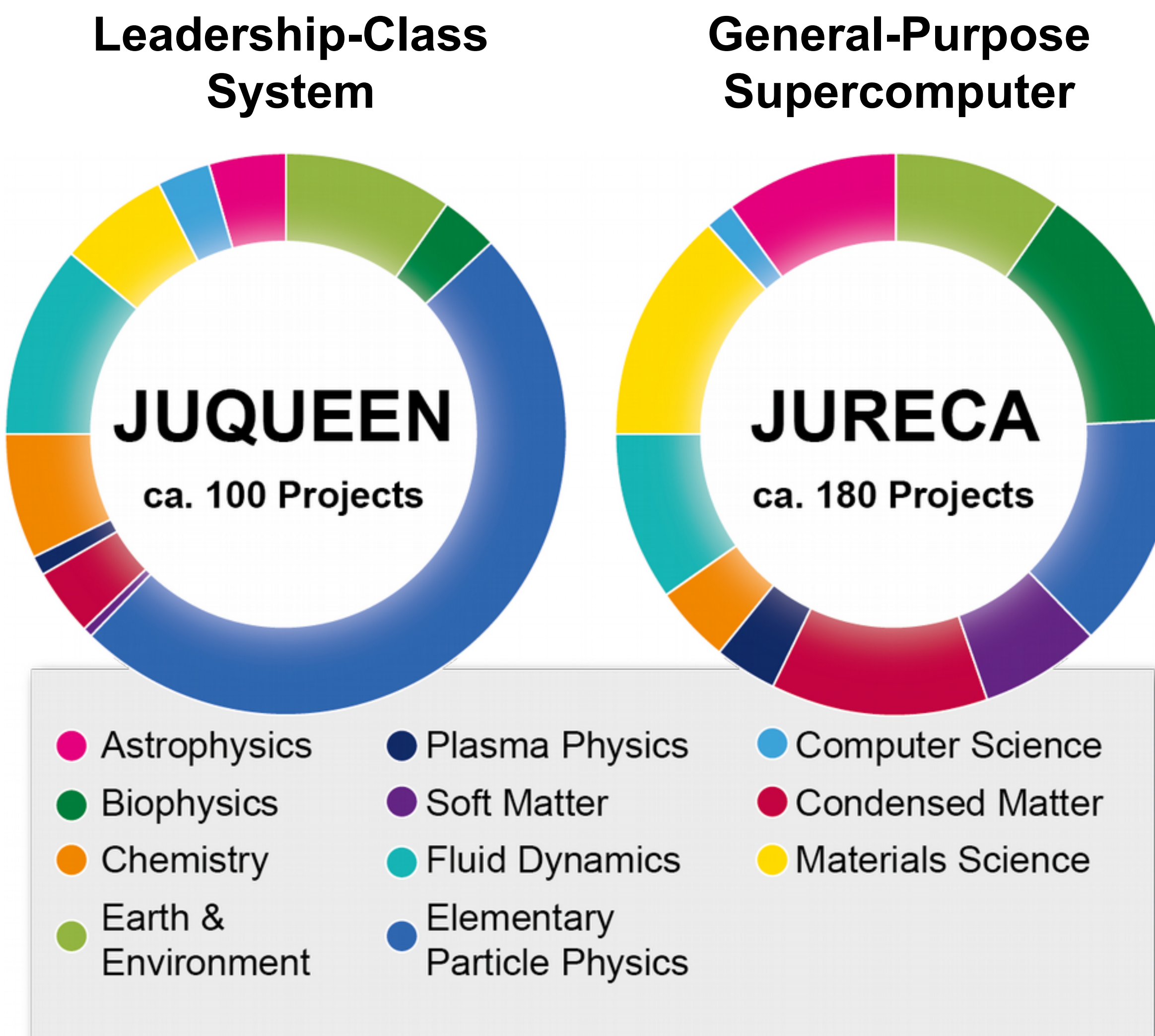
~ 900 PhD students (Campus Jülich)
~ 350 Trainees

Research for the future for
key technologies of the **next generation** and **Information**

Past: Supercomputer evolution @ JSC



Research Field Usage 11/2015-04/2017



Granting periods
05/2016 – 04/2017
11/2015 – 10/2016

Application's Scalability

Only few application capable to scale to O(450k) cores

- Sparse matrix-vector codes
- Highly regular communication patterns
- Well suited for BG/Q

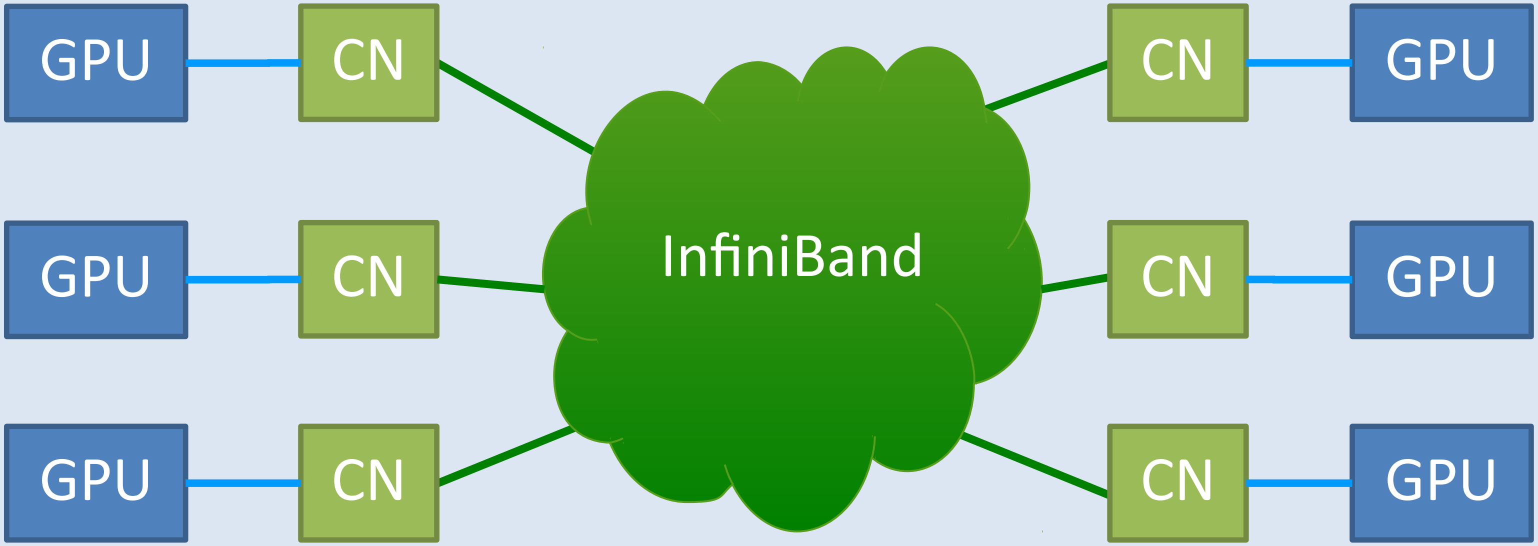
Most applications are more complex

- Less regular control flow / memory access
- Complicated communication patterns
- Less capable to exploit accelerators

How to map different requirements to most suited hardware

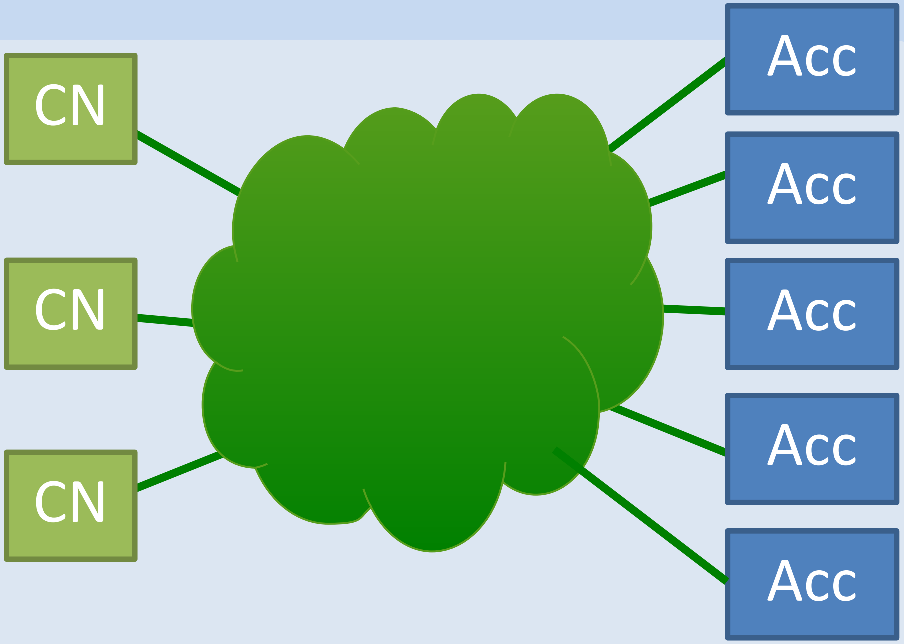
- Heterogeneity might be beneficial
- Do we need better programming models?

Heterogeneous Clusters



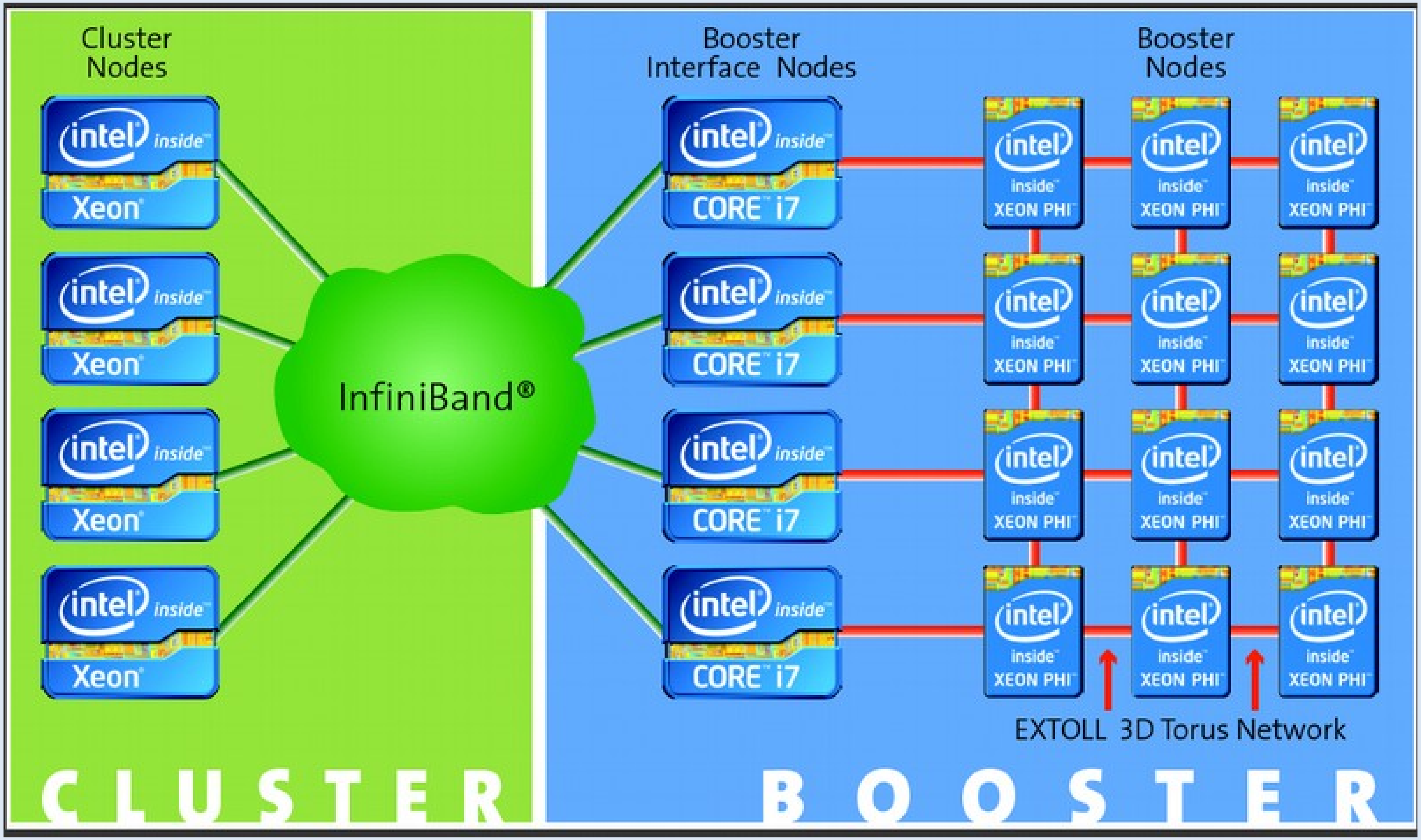
- Flat IB-topology
- Simple management of resources
- Static assignment of CPUs to GPUs
- Accelerators not capable to act autonomously

Alternative Integration

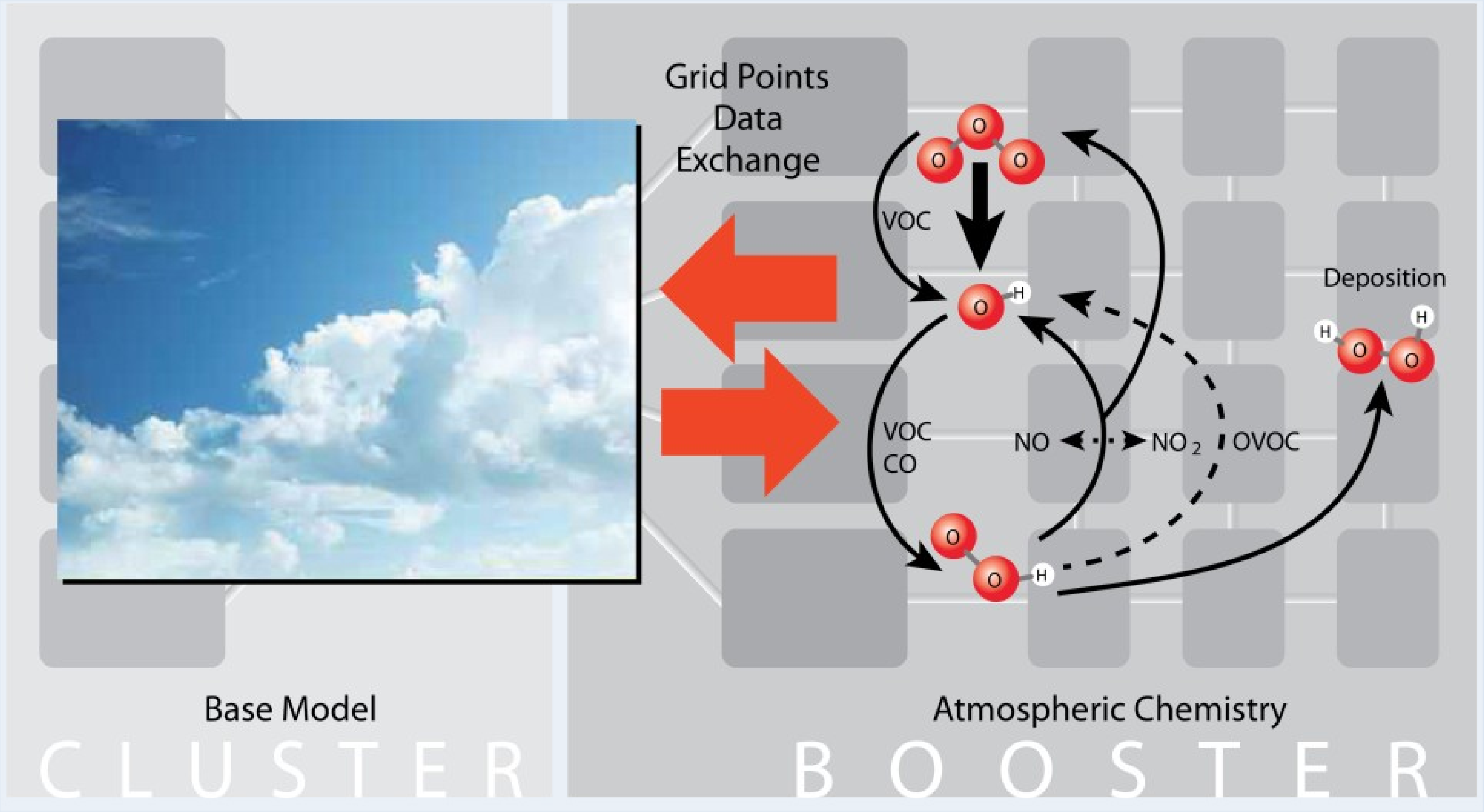


- Go for more capable accelerators (e.g. MIC)
- Attach all nodes to a low-latency fabric
- All nodes might act autonomously
- Dynamical assignment of cluster-nodes and accelerators
  - IB can be assumed as fast as PCIe besides latency
- Ability to off-load more complex (including parallel) kernels
  - communication between CPU and Accelerator less frequently
  - larger messages i.e. less sensitive to latency

Hardware Architecture



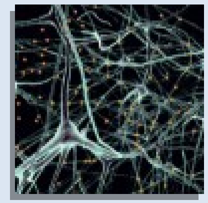
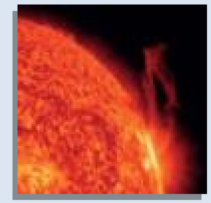
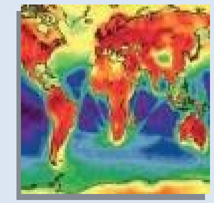
CyI – Climate Simulation

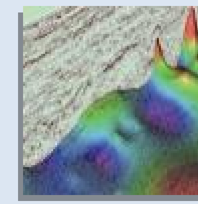
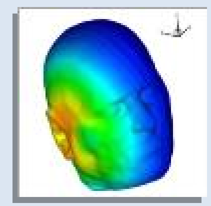


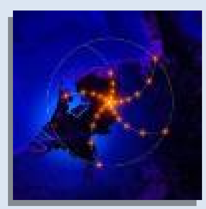
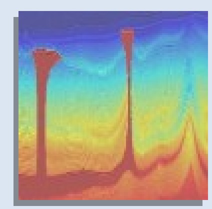
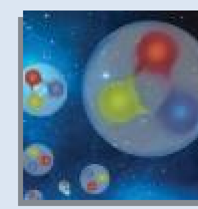
DEEP Projects

Application-driven approach

- DEEP+DEEP-ER applications:
  - Brain simulation (EPFL)
  - Space weather simulation (KULeuven)
  - Climate simulation (Cyprus Institute)
  - Computational fluid engineering (CERFACS)
  - High temperature superconductivity (CINECA)
  - Seismic imaging (CGG)
  - Human exposure to electromagnetic fields (INRIA)
  - Geoscience (LRZ Munich)
  - Radio astronomy (Astron)
  - Oil exploration (BSC)
  - Lattice QCD (University of Regensburg)

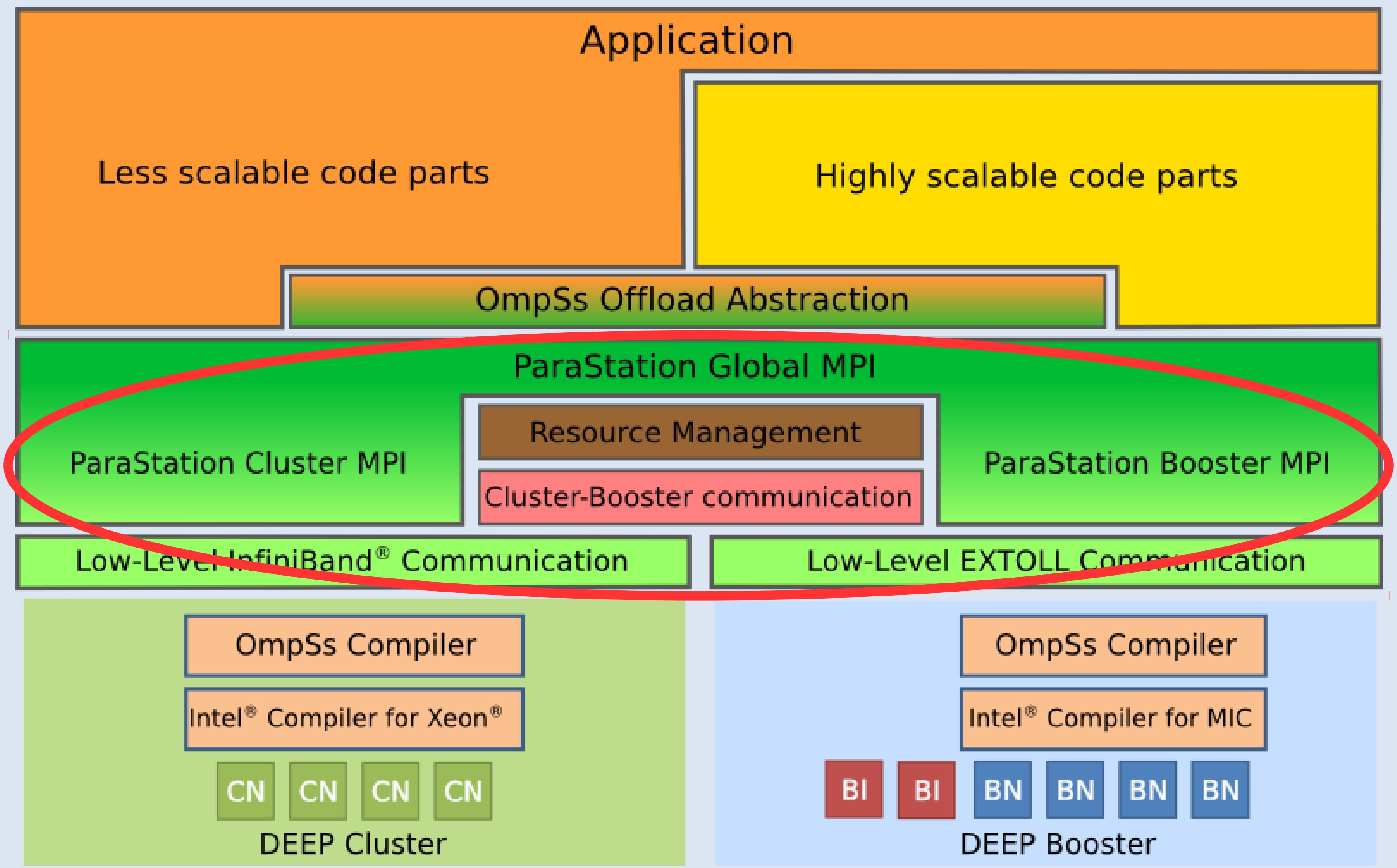




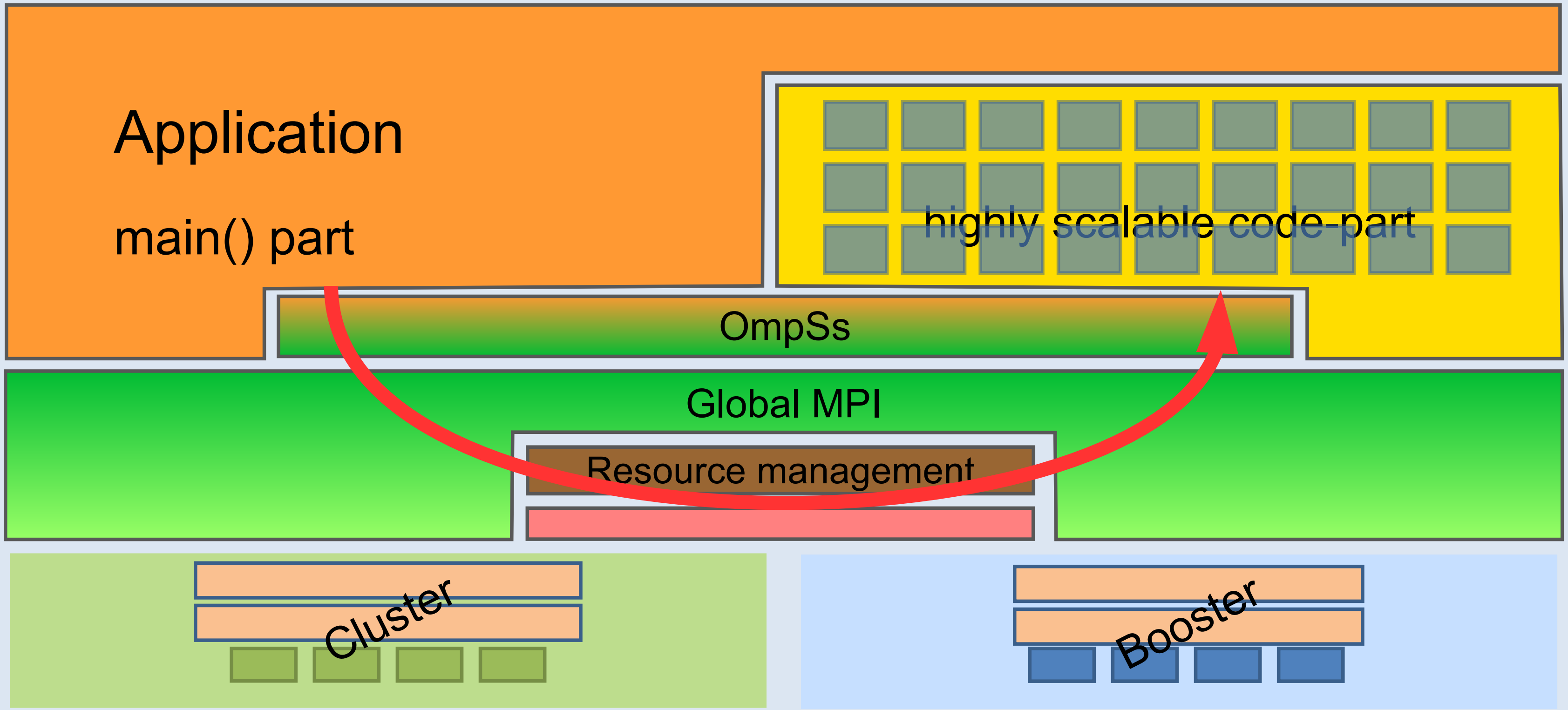




- Goals:
  - Co-design and evaluation of architecture and its programmability
  - Analysis of the I/O and resiliency requirements of HPC codes

Software Architecture



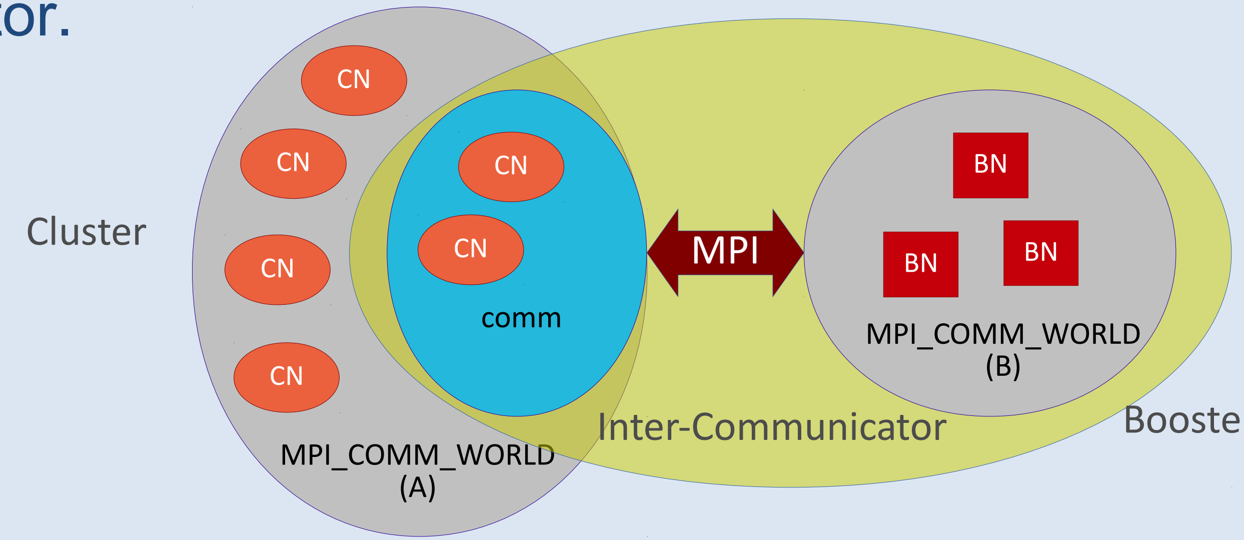
Application Startup



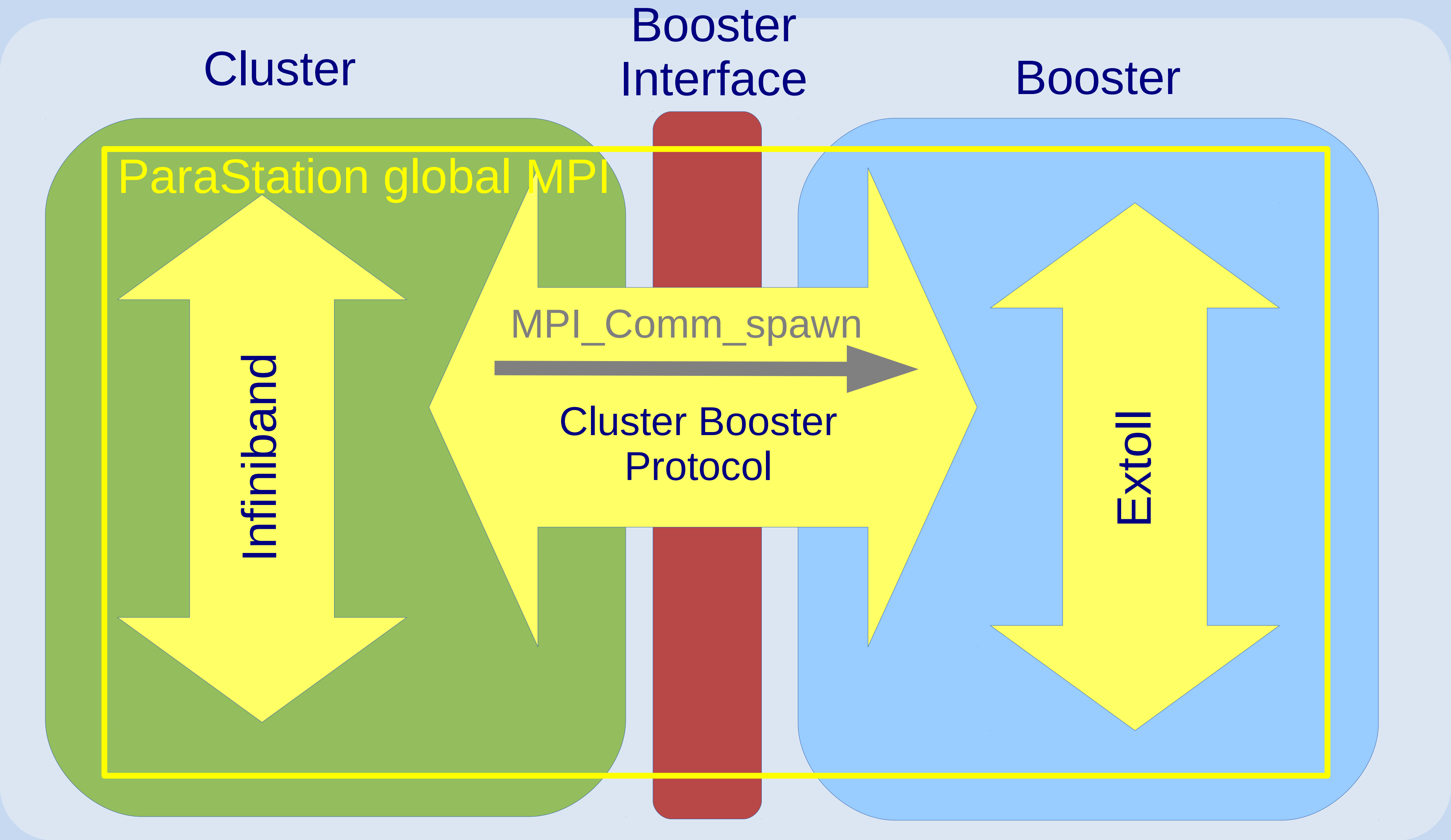
- Application's main()-part runs on Cluster-nodes (CN) only
- Actual spawn done via global MPI
- OmpSs acts as an abstraction layer
- Spawn is a collective operation of Cluster-processes
- Highly scalable code-parts (HSCP) utilize multiple Booster-nodes (BN)

MPI Process Creation

- The inter-communicator contains all parents on the one side and all children on the other side.
  - Returned by MPI_Comm_spawn for the parents
  - Returned by MPI_Get_parent by the children
- Rank numbers are the same as in the the corresponding intra-communicator.



Programming



Application running on DEEP

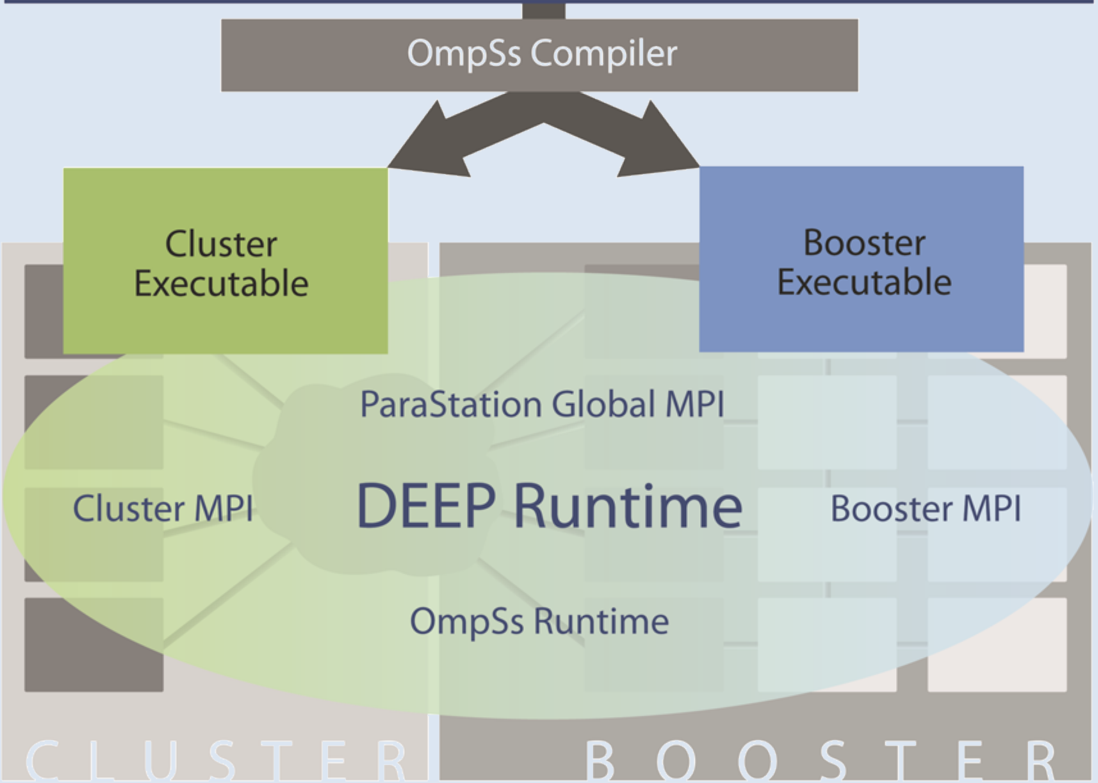
Source code

```
int main(int argc, char *argv[]){
    /*...*/
    for(int i=0; i<3; i++){
        #pragma omp task in(…) out (…) onto (com, size*rank+1)
        foo_mpi(i, …);}}
```

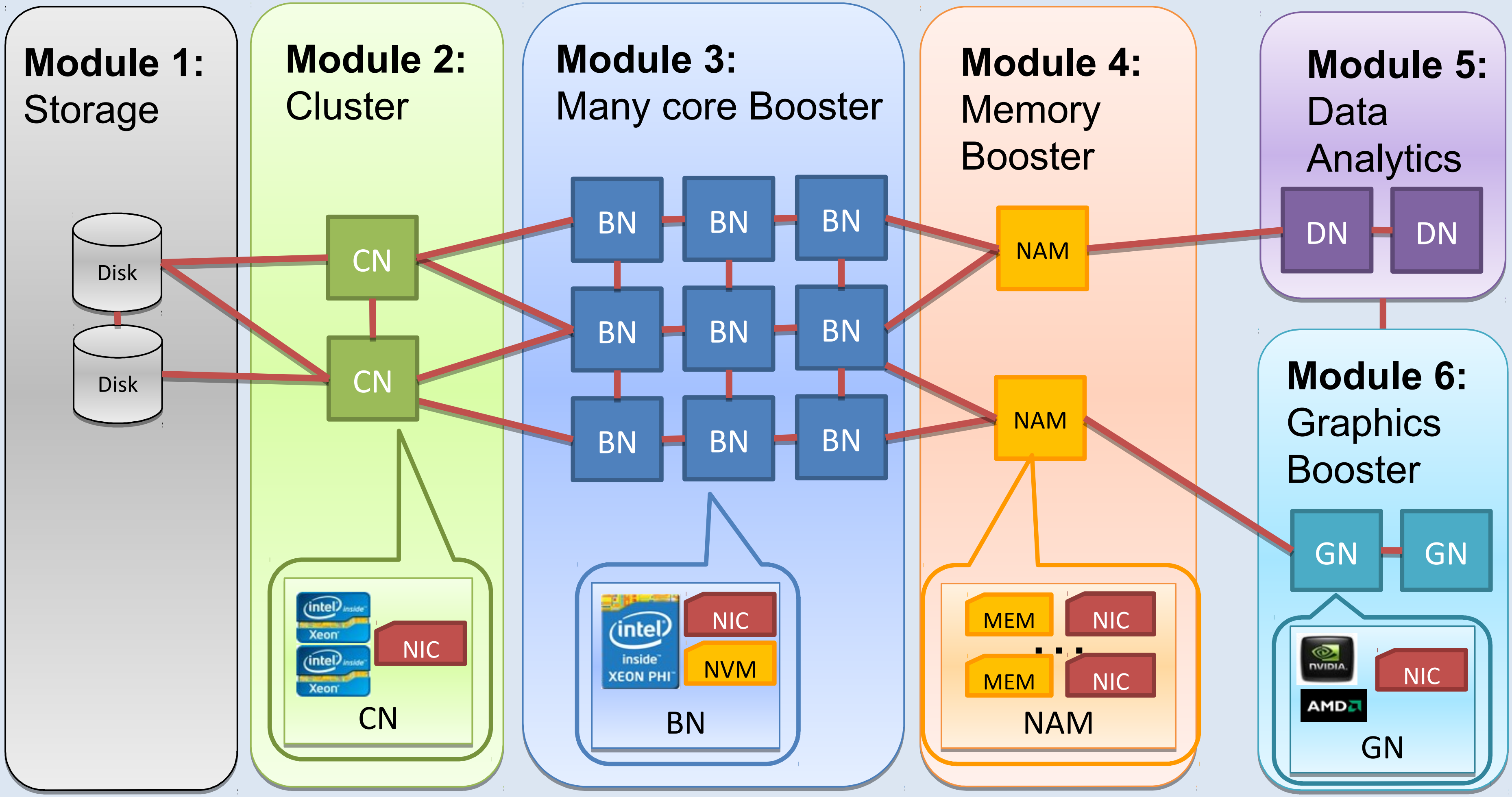
Compiler

Application binaries

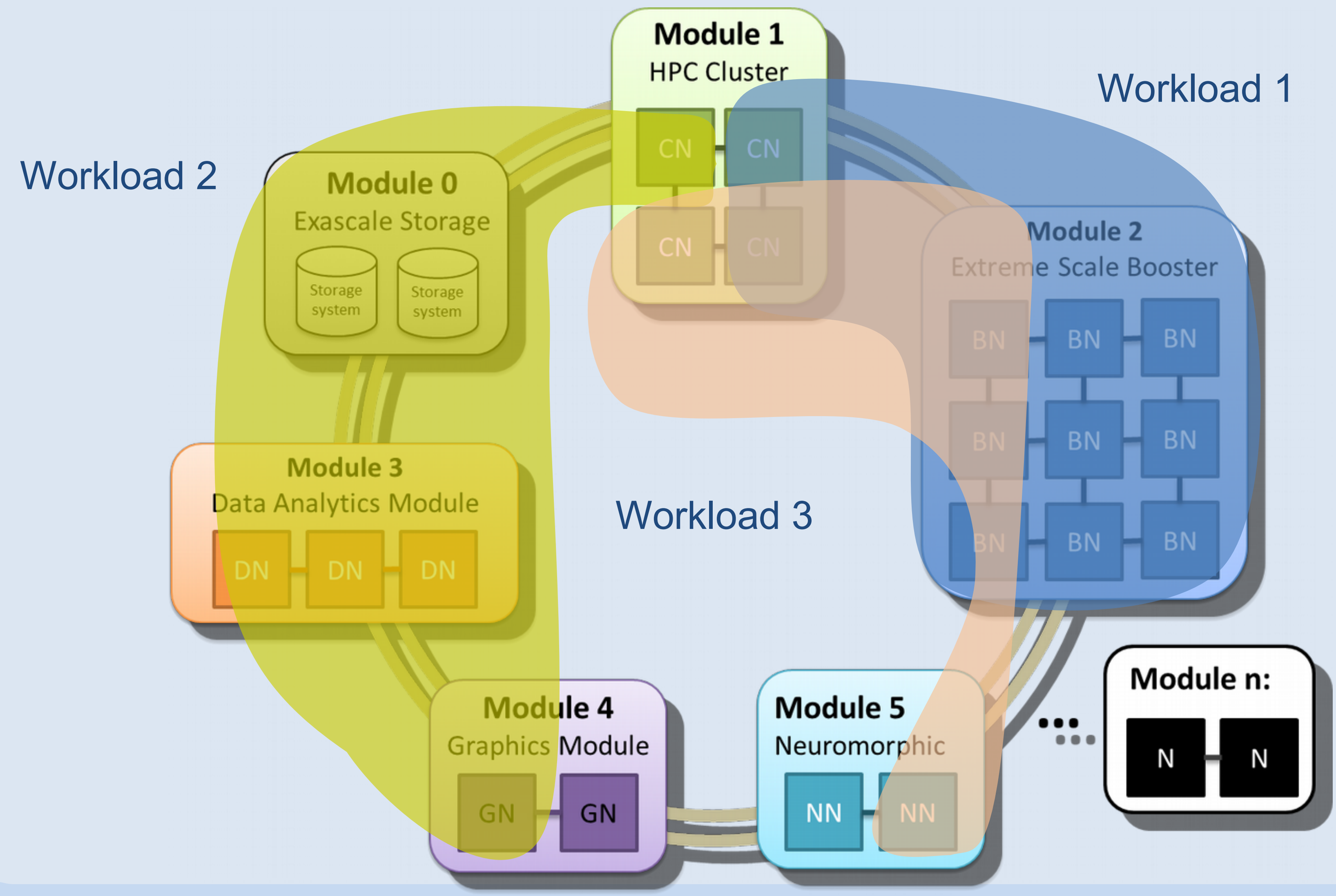
DEEP Runtime



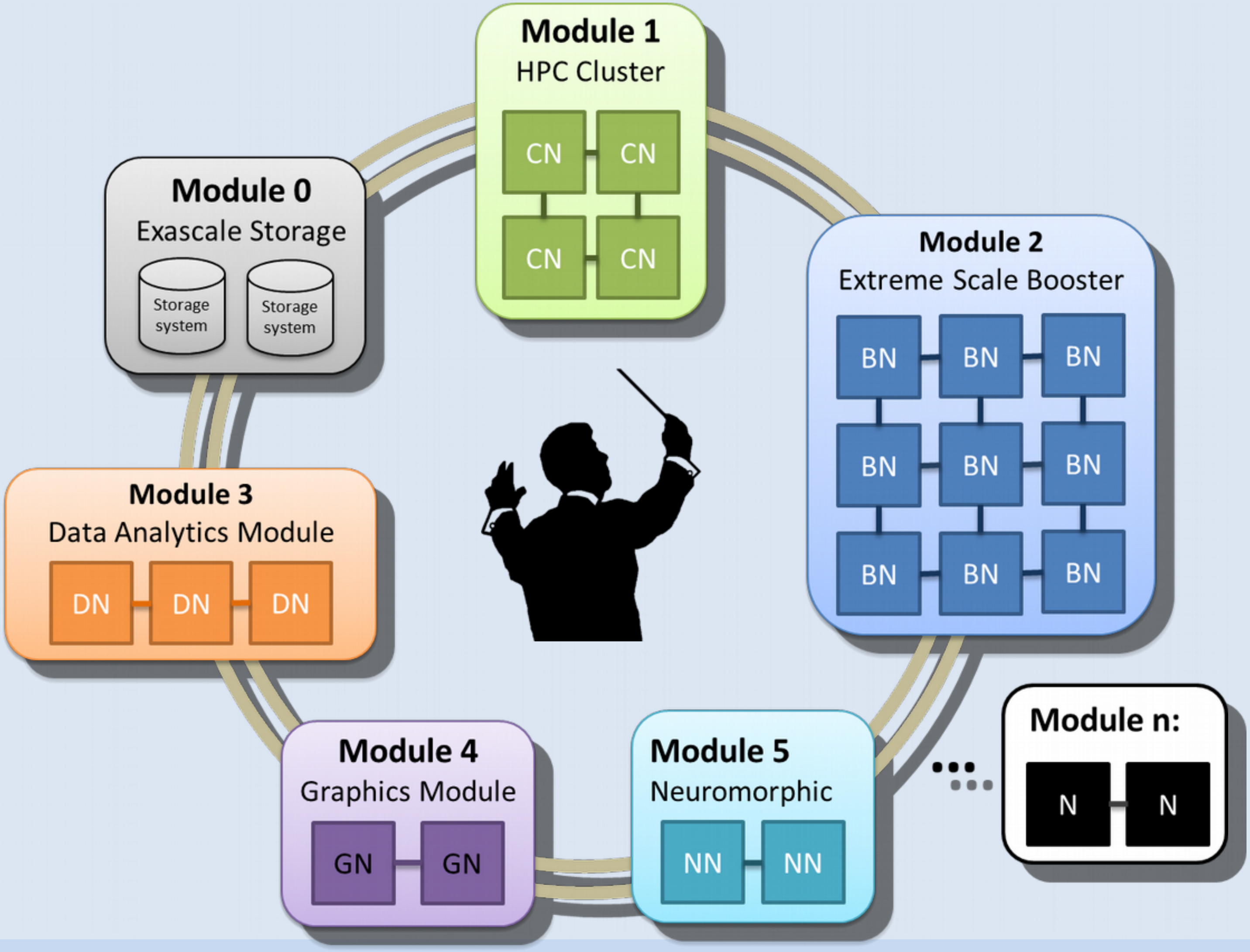
Generalization of the Cluster-Booster concept



Modular Supercomputing

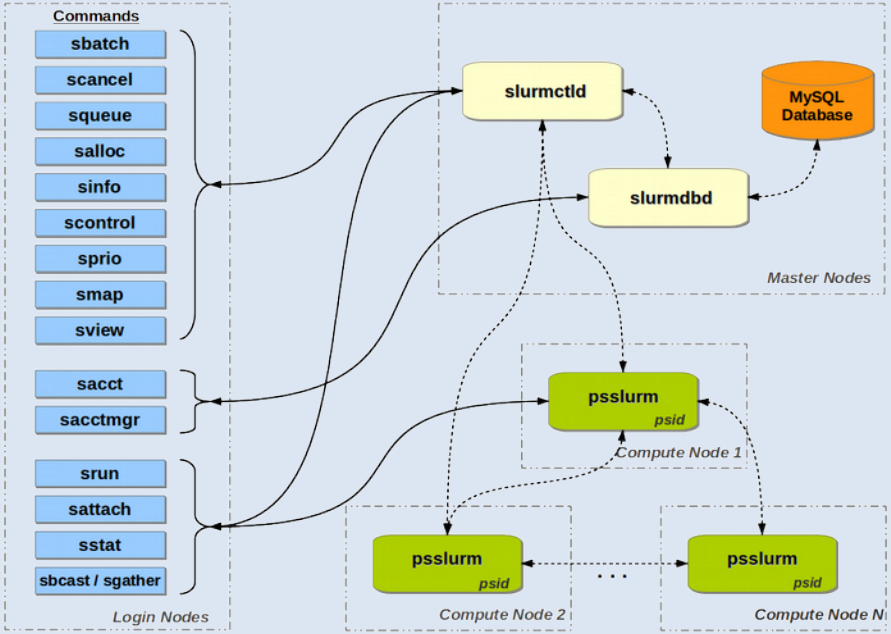


Modular Supercomputing

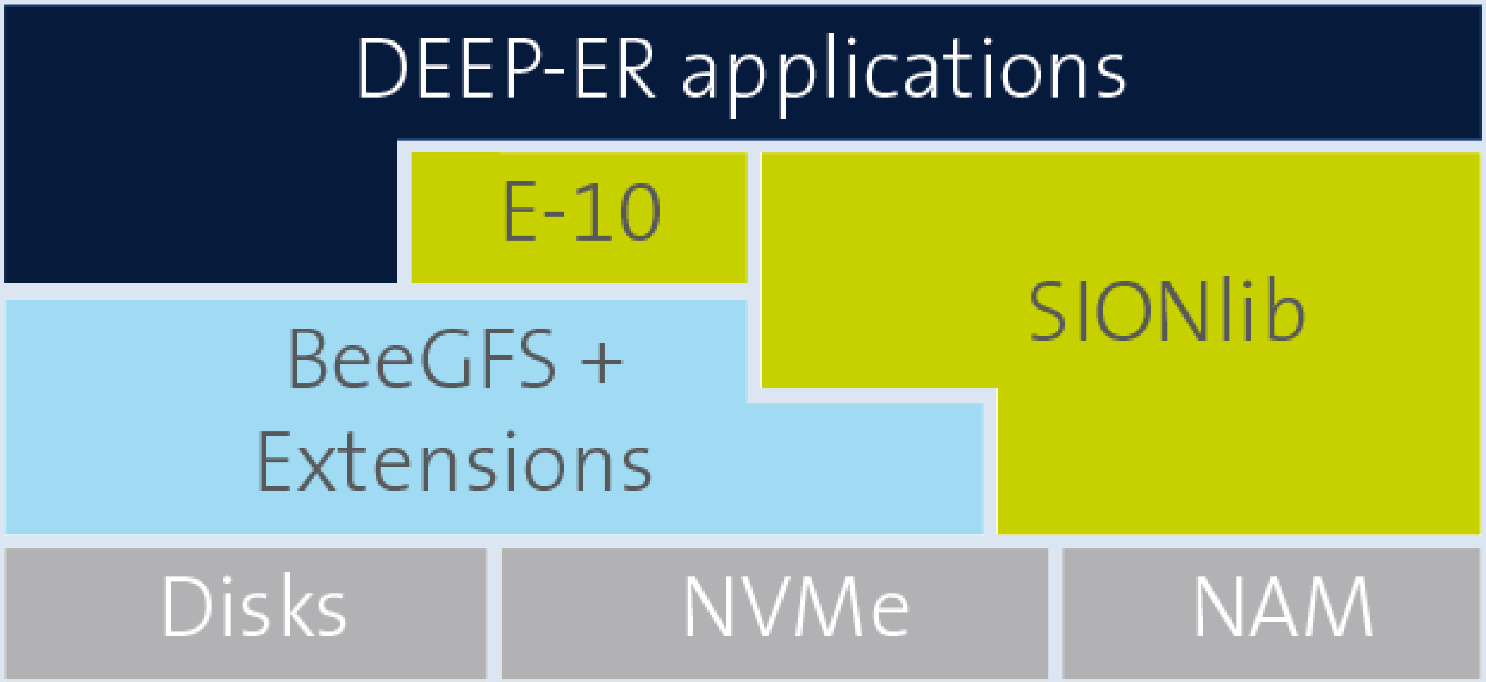


SLURM

- JSC decided to go for SLURM with the start of JURECA in 2015
- Close collaboration with ParTec for deep integration with PS-MPI
- Currently waiting for job-packs
- We expect to extend the scheduling capabilities for support of complex job requirements
- Workflows without communication via the file-system



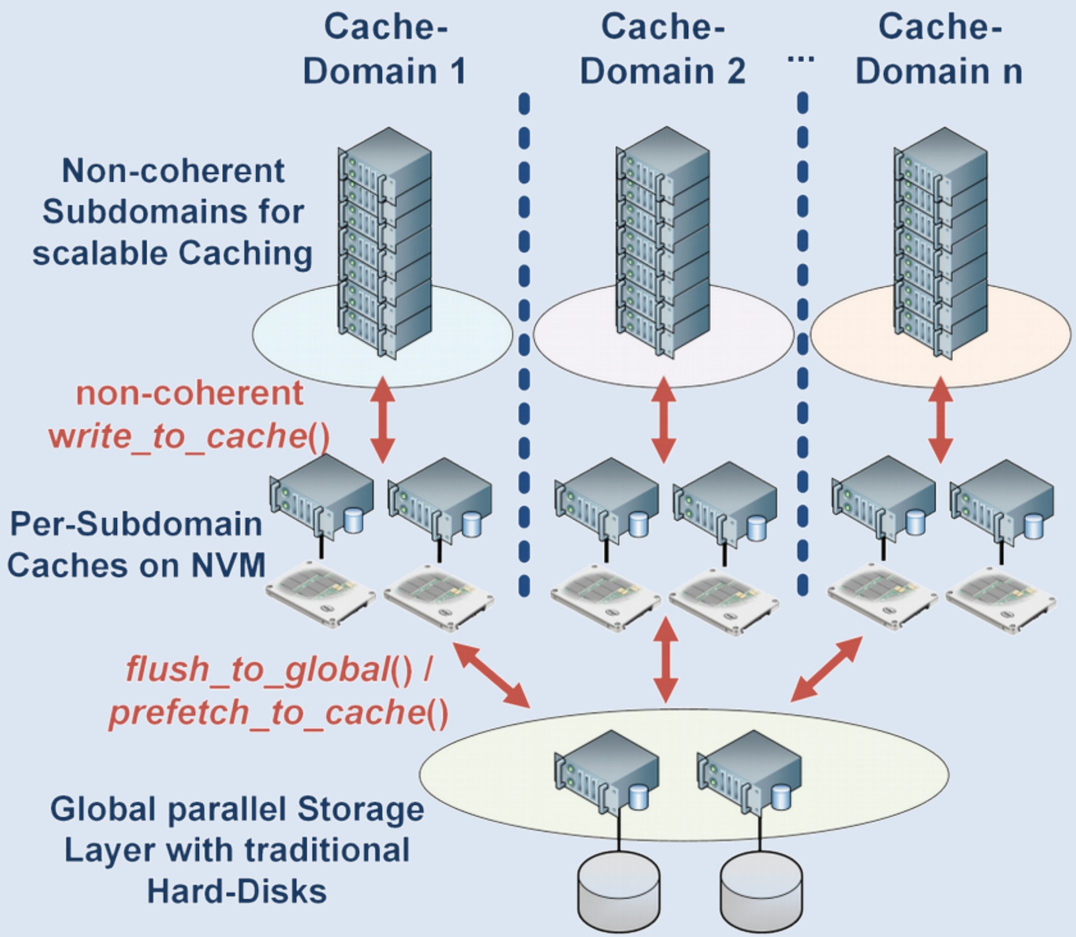
Scalable I/O in DEEP-ER



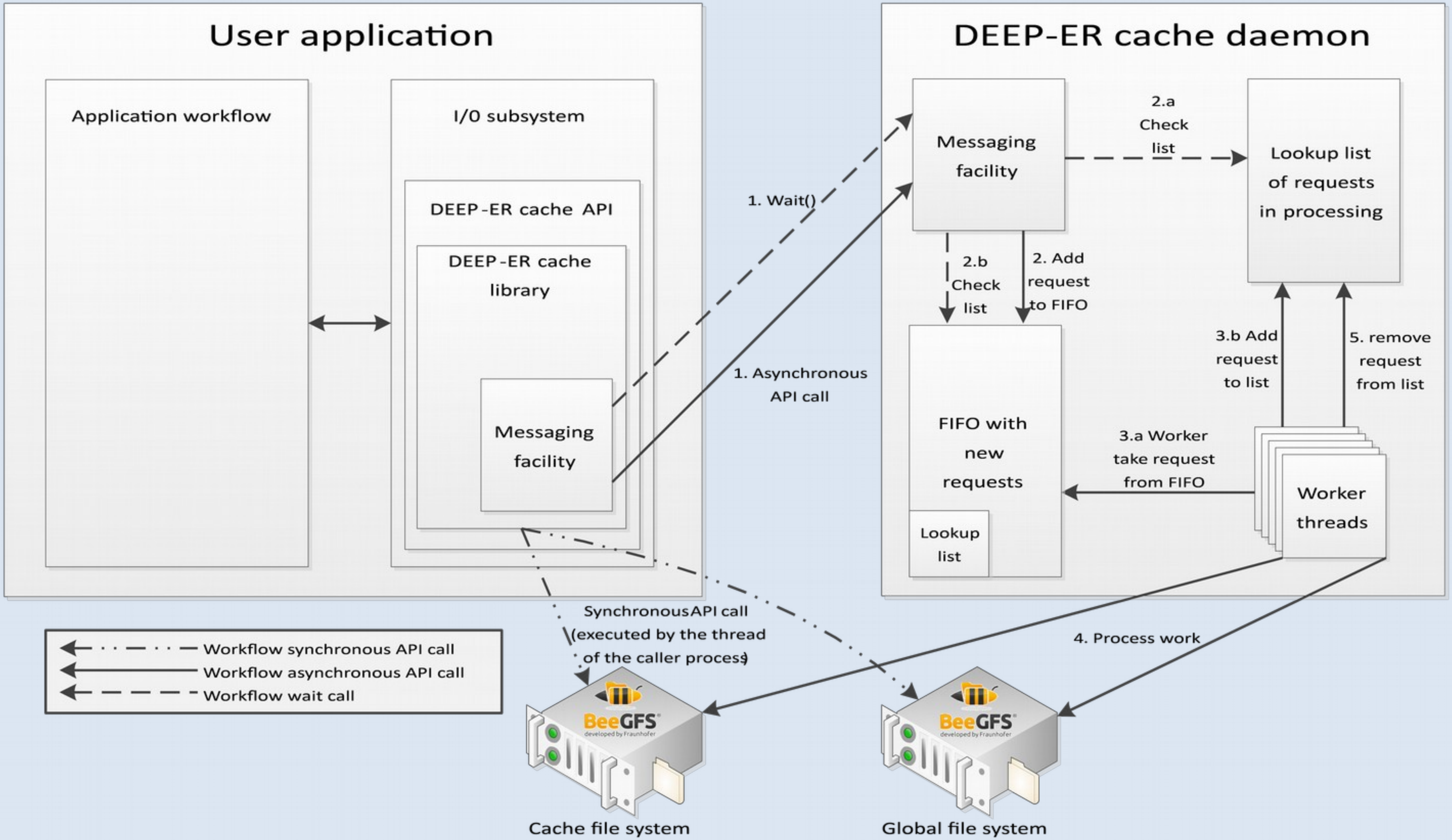
- Improve I/O scalability on all usage-levels
- Used also for checkpointing

BeeGFS and Caching

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

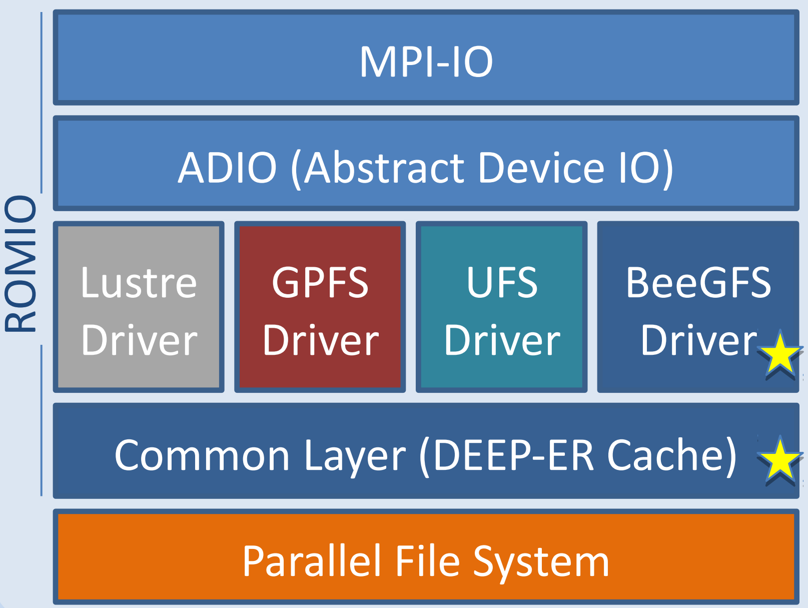


Asynchronous Cache API

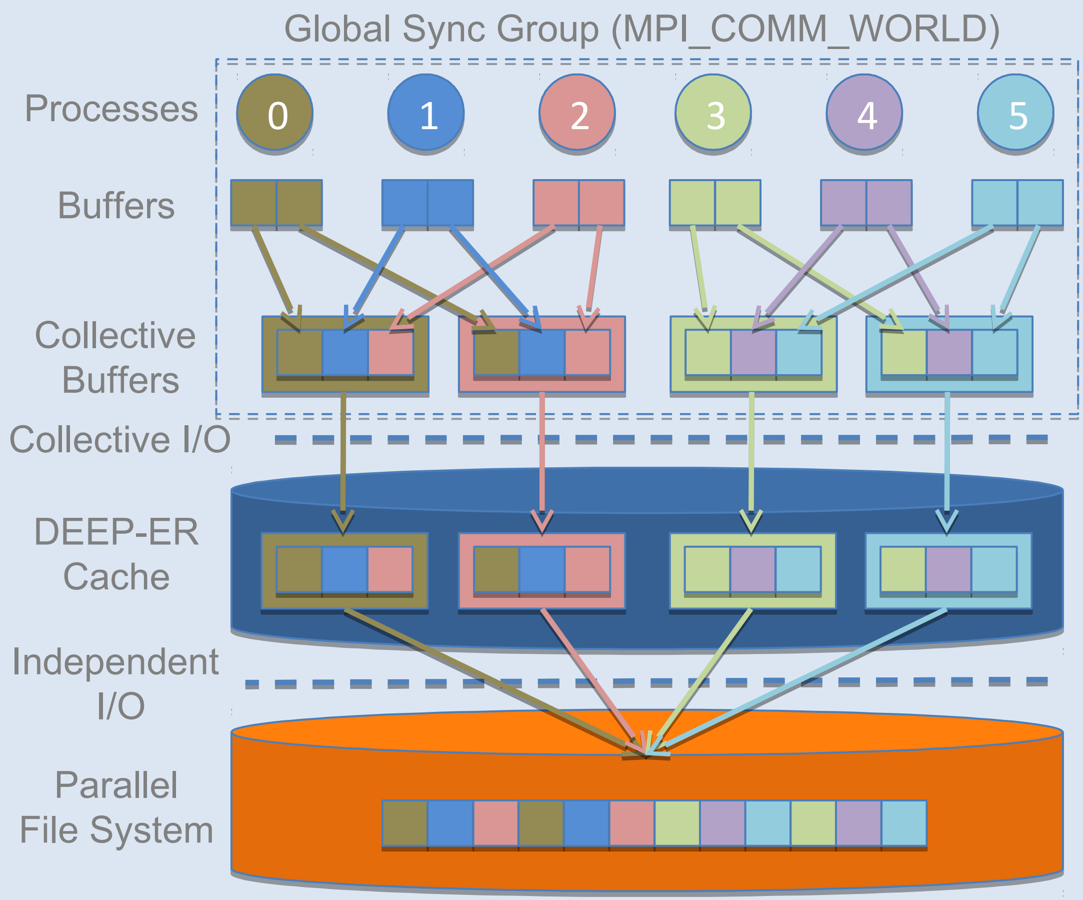


Cache Integration in ROMIO

New MPI-IO Hints ★
e10_cache
e10_cache_path
e10_cache_flush_flag
e10_cache_discard_flag
e10_cache_threads

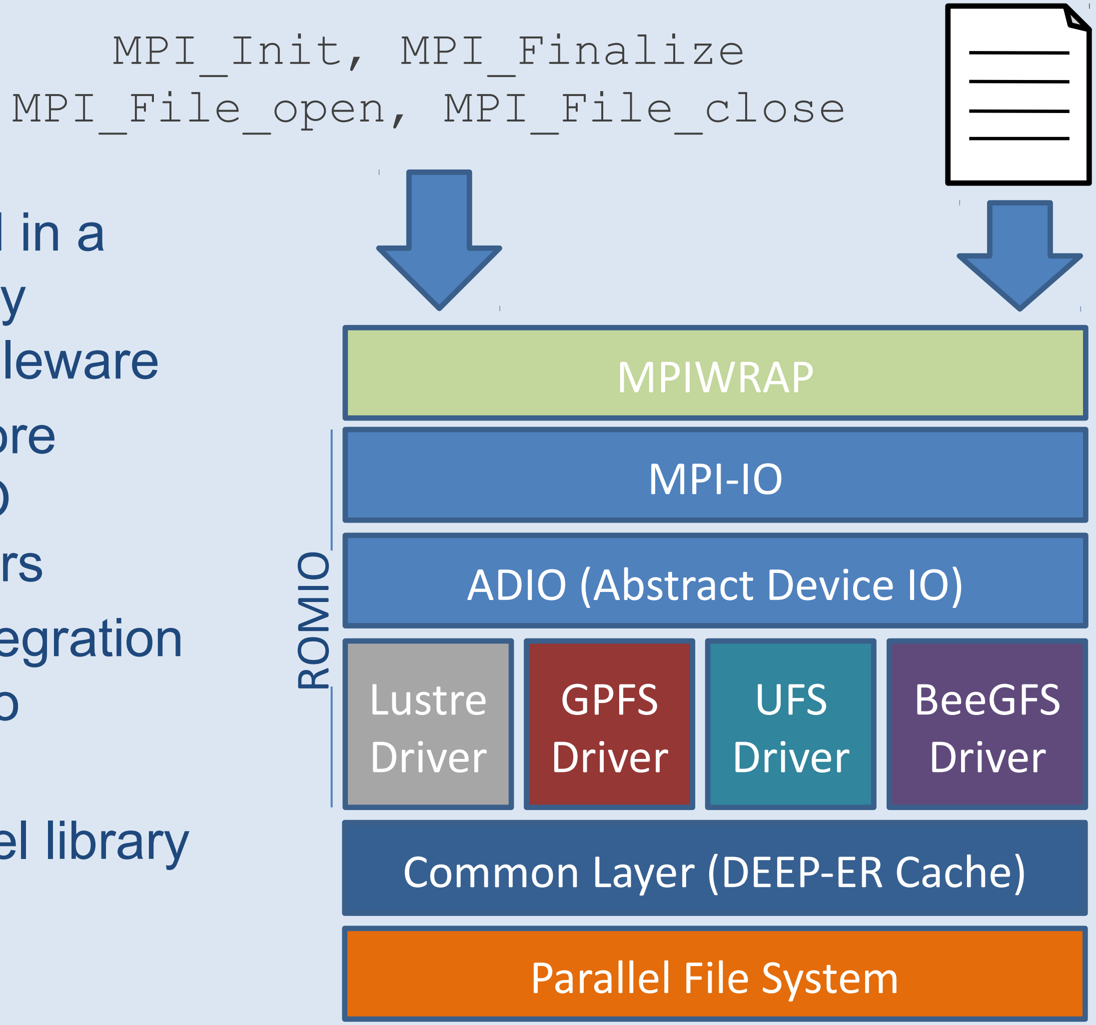


★ Developed in DEEP-ER and tested on DEEP Cluster



MPIWRAP Support Library

- MPI-IO hints are defined in a config file and injected by libmpiwrap into the middleware
- Provides deeper and more flexible control of MPI-IO functionalities to the users
- Provides transparent integration of E10 functionalities into applications
- Works with any high level library (e.g. pHDF5)



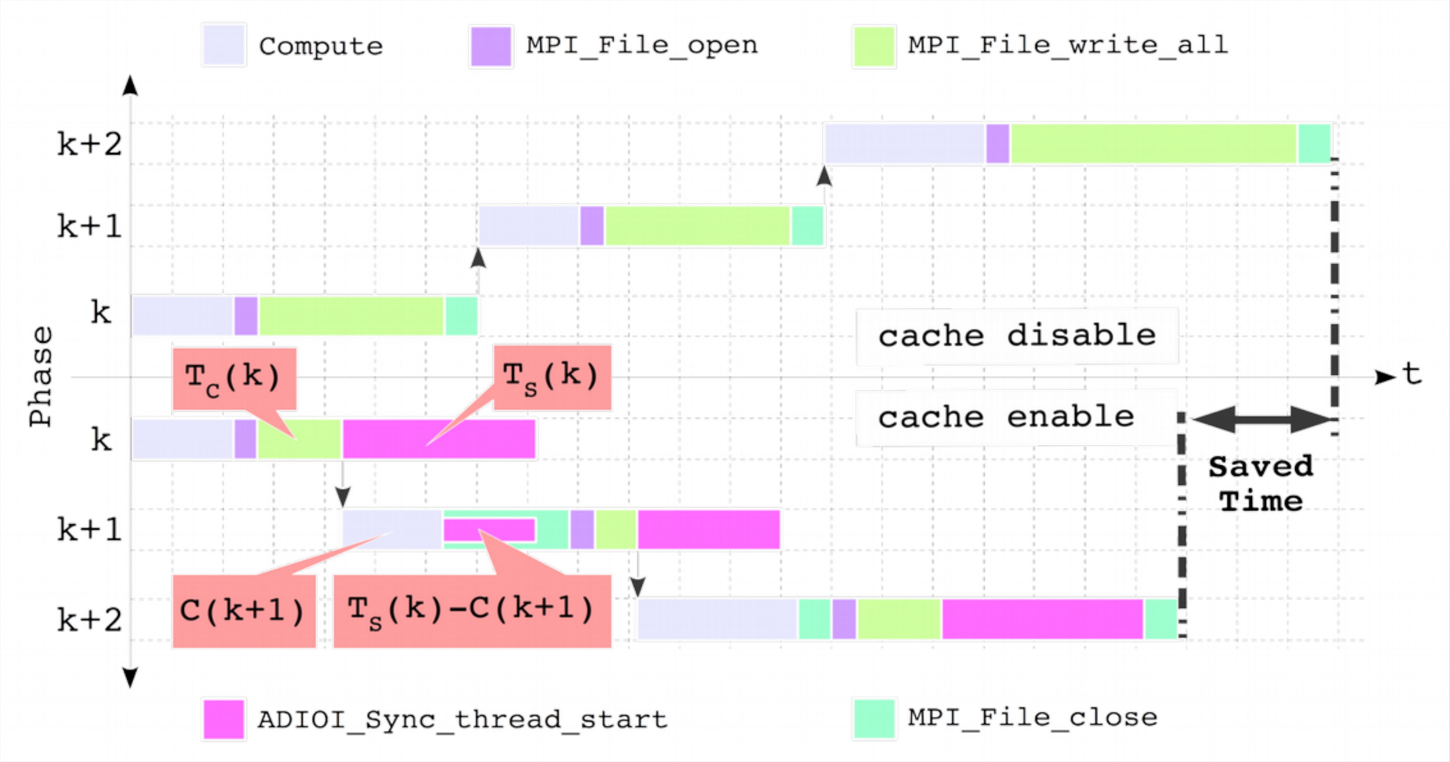
Effects of the Cache

S (k): amount of data written to the file at phase k

T_c(k): time to write S(k) to the cache

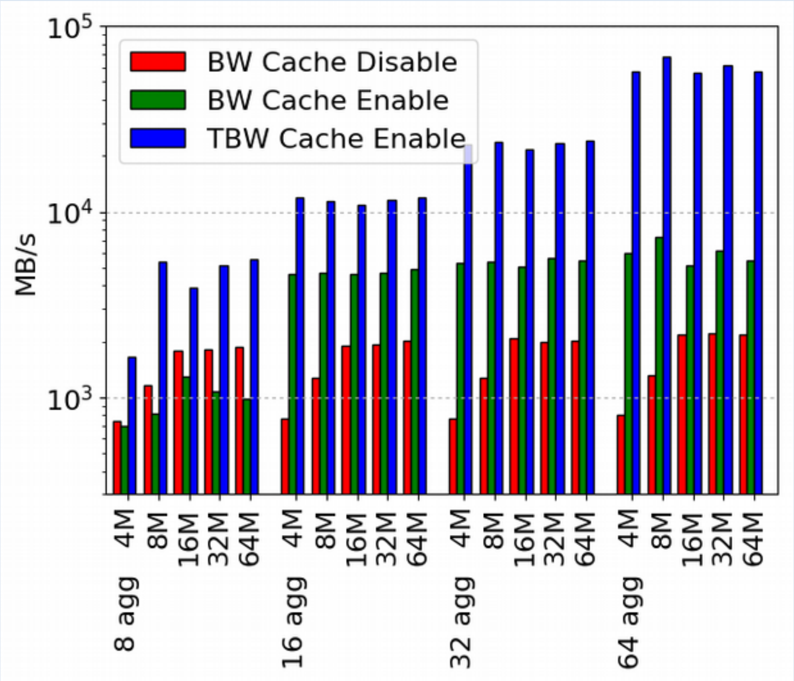
T_s(k): time to sync S(k) with the parallel file system

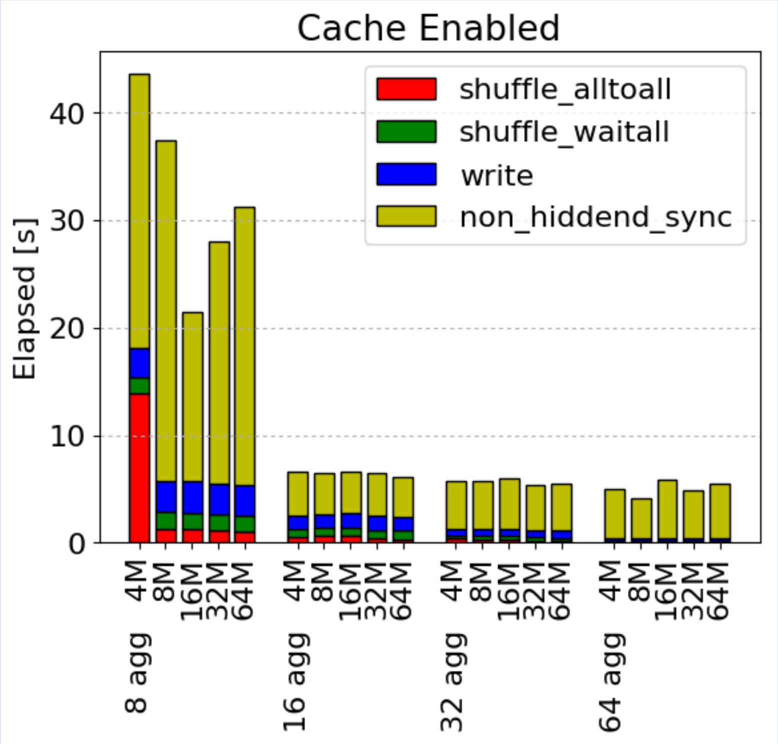
C (k): compute time at phase k



Exper. Results – IOR

512 processes (8/node) writing a 32GB share file varying # of aggregators and collective buffer size





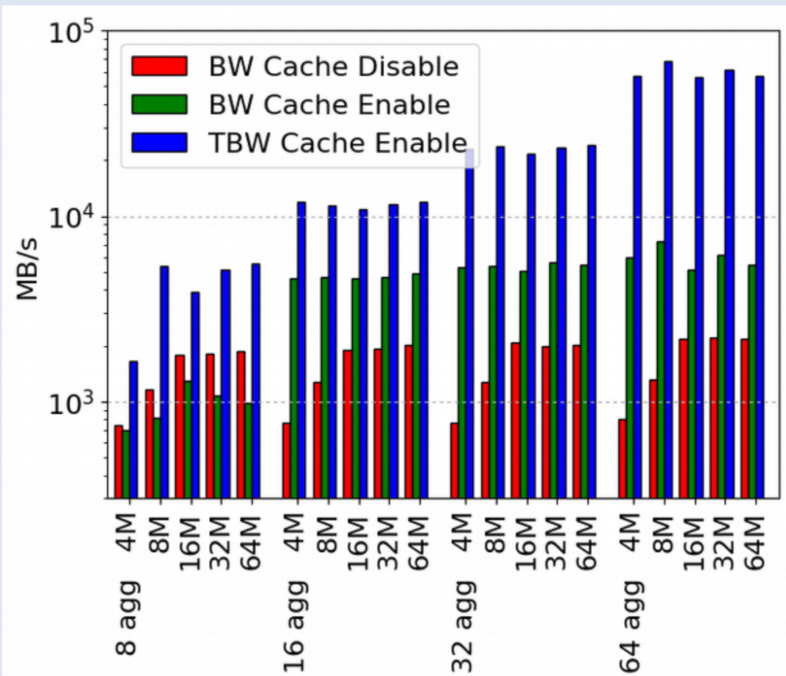
TBW represents the maximum theoretical bandwidth achievable when writing to the cache without flushing it to the parallel file system

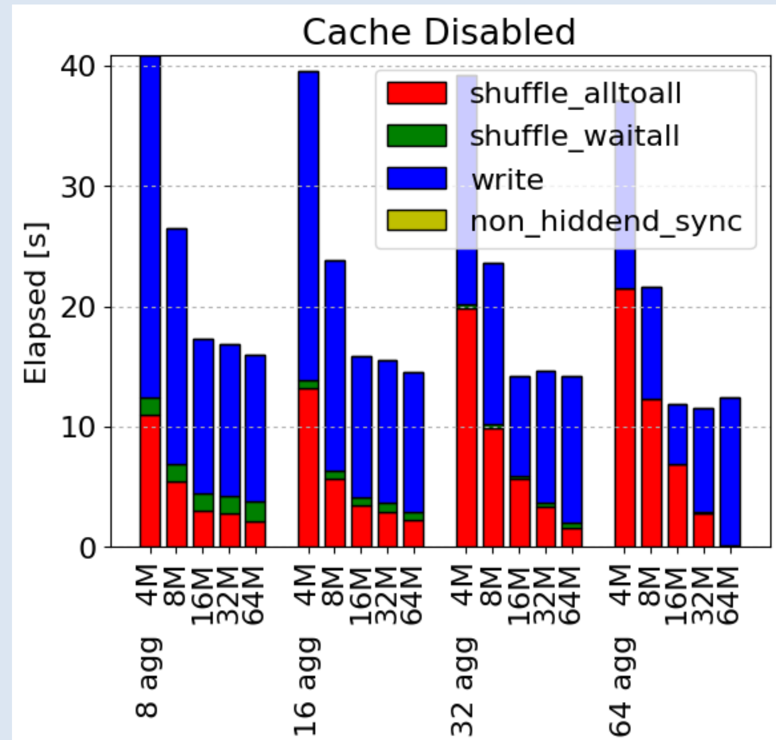
- In IOR the last write sync phase is not overlapped with any computation and thus it is affecting the overall bandwidth performance

CONGIU, G., et al. 2016 IEEE

Exper. Results – IOR

512 processes (8/node) writing a 32GB share file varying # of aggregators and collective buffer size





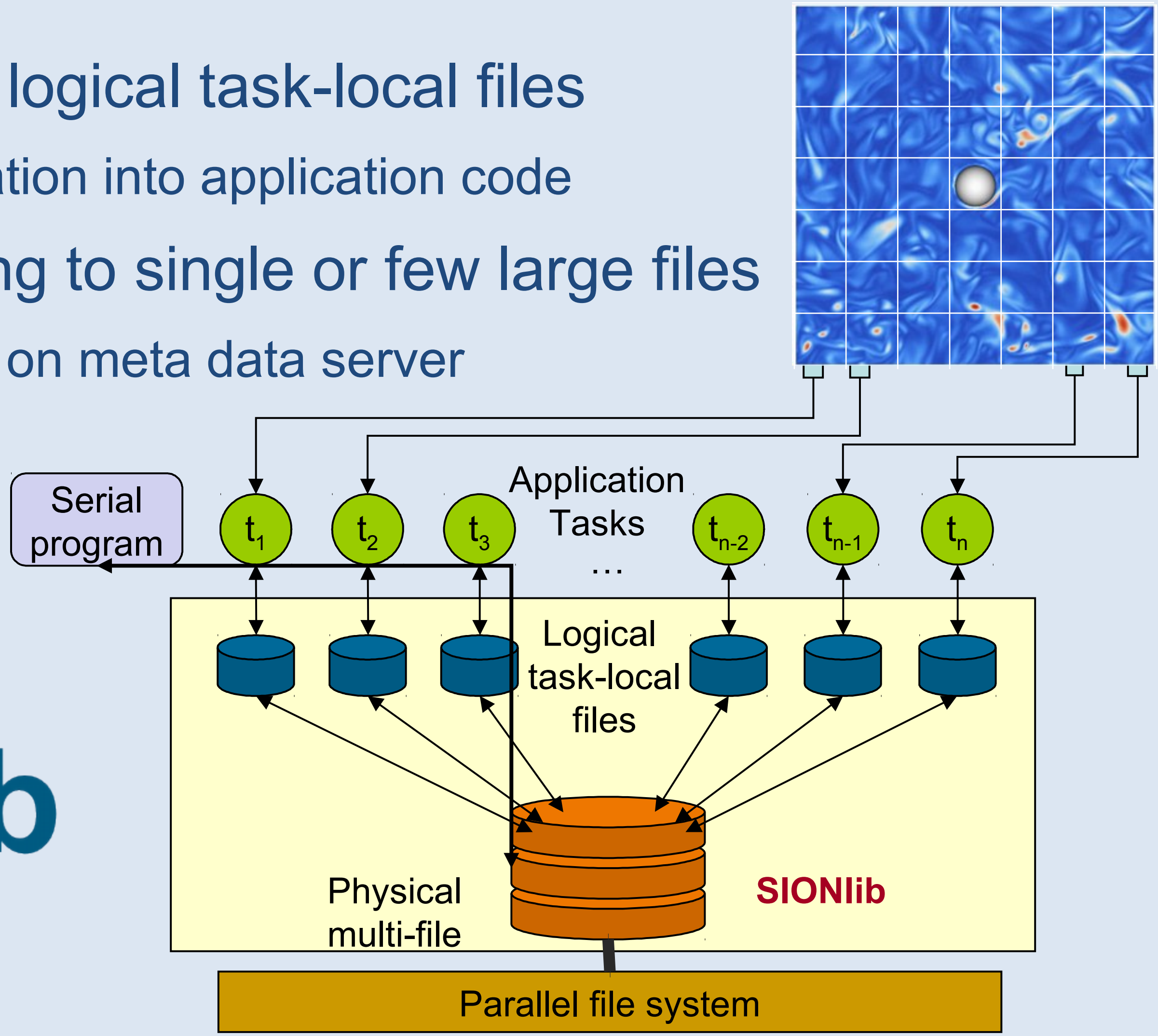
TBW represents the maximum theoretical bandwidth achievable when writing to the cache without flushing it to the parallel file system

- ➢ In IOR the last write sync phase is not overlapped with any computation and thus it is affecting the overall bandwidth performance

CONGIU, G., et al. 2016 IEEE

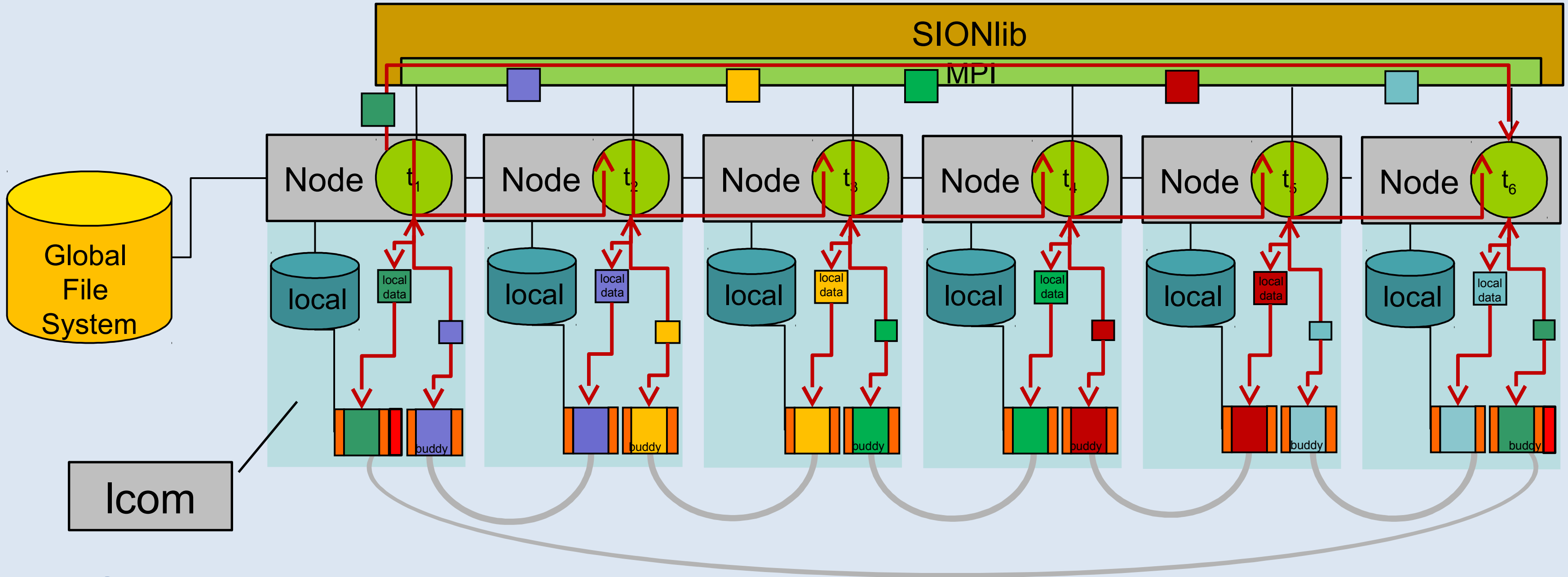
SIONlib

- API resembles logical task-local files
  - Simple integration into application code
- Internal mapping to single or few large files
  - Reduces load on meta data server



SIONlib

Write buddy checkpoint

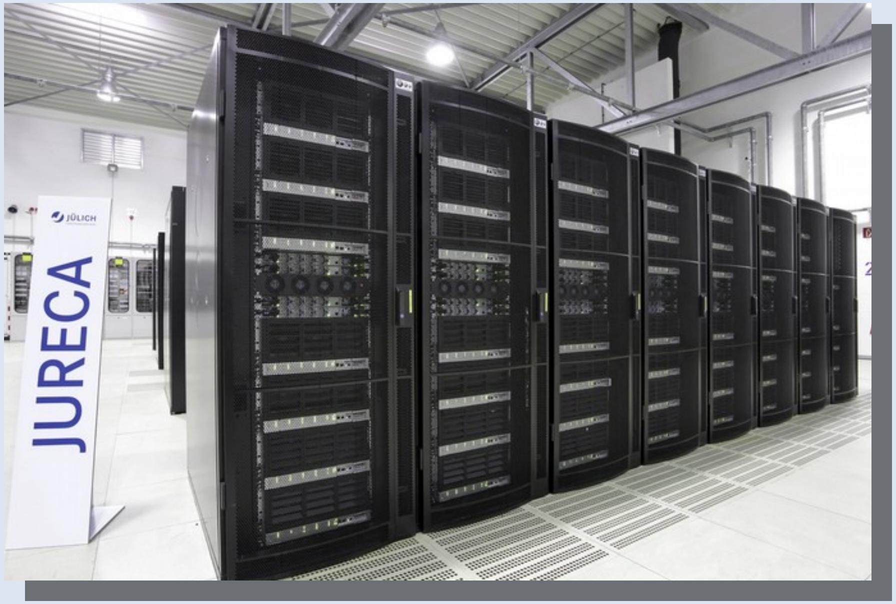


- Open: `sid=sion_paropen_mpi(…,"bw,buddy",MPI_COMM_WORLD, lcom,…)`
- Write: `sion_coll_write_mpi(data,size,n,sid)`
- Close: `sion_parclose(sid)`
- Write-Call will write data first to local chunk, and then sent it to the associated buddy which writes the data to a second file

JURECA Cluster

- Hardware
  - T-Platforms V210 blade server solution
    - 28 Racks
    - 1884 Nodes: Intel Xeon Haswell (24 cores)
    - DDR4: 128 / 256 / 512 GiB
    - InfiniBand EDR (100 Gbps) - Fat tree topology
    - NVIDIA GPUs: 75×2 K80 + 12×2 K40
  - Peak performance: 1.8PF (CPUs) + 0.4PF(GPUs)
  - Main memory: 281 TiB
- Software
  - CentOS 7 Linux
  - SLURM batch system
  - ParaStation Cluster Management
  - GPFS file system

Installed at JSC since July 2015

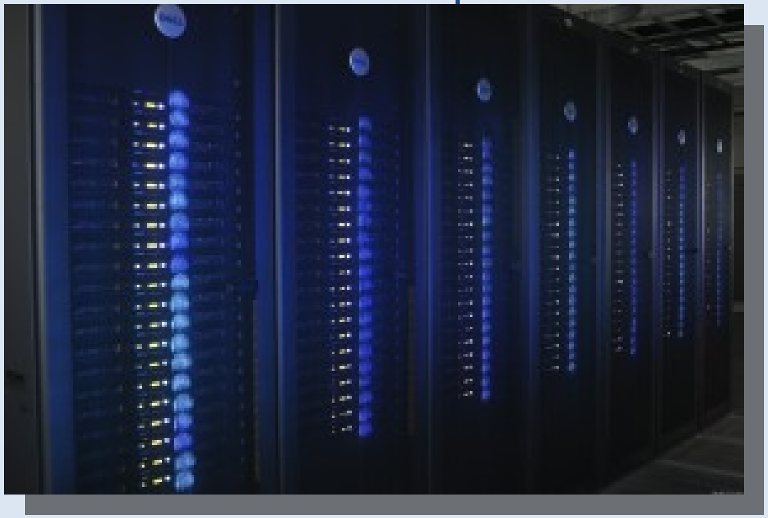


JURECA Booster

- Collaboration of JSC, Intel (+DELL) & ParTec
- Network Bridge EDR-OPA (Cluster-Booster)
- Management of heterogeneous resources in SLURM

- Hardware
  - 1640 nodes KNL 7250-F
    - 96 GB DDR4
    - 16 GB MCDRAM
    - 200 GB local SSD
  - Intel OmniPath (OPA) – Fat tree topology
  - Fully integrated with JURECA Cluster
    - 198 OPA-to-EDR bridges to connect to Cluster
    - Same login nodes
- Software
  - SLURM (orchestrating jointly Cluster and Booster)
  - ParaStation Cluster Management
  - GPFS file system

Installation to be completed in 2017

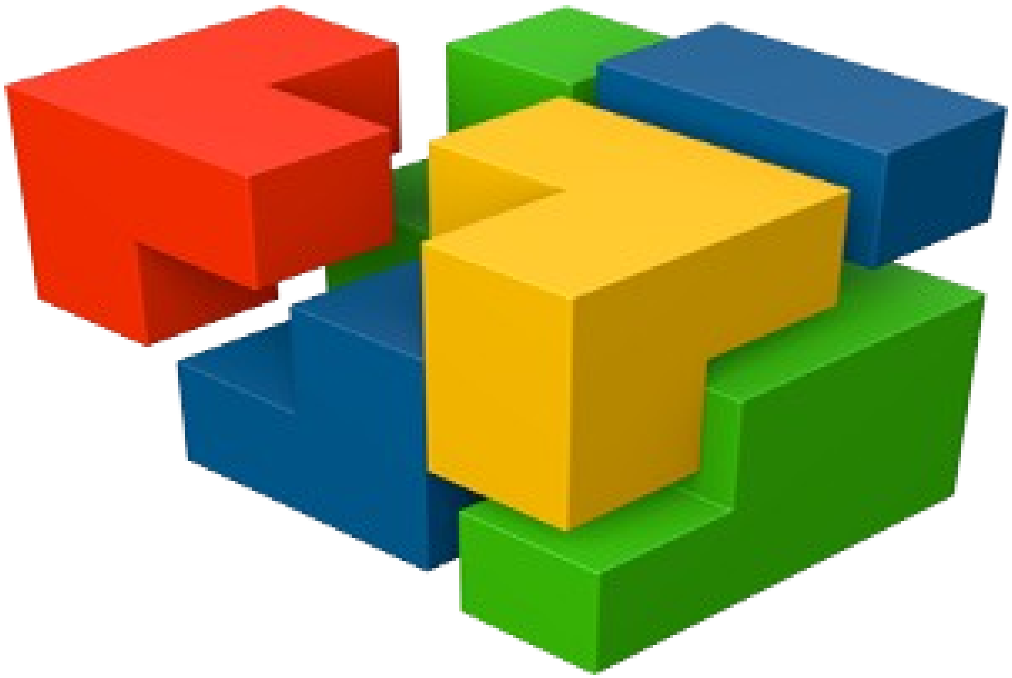


(Not the real Booster, just to give an impression)

Summary

- The DEEP projects bring a new view to heterogeneity
  - Cluster-Booster architecture
  - Hardware, software and applications jointly developed
  - Strongly co-design driven
- DEEP-ER explored future directions of I/O
  - On filesystem level → BeeGFS
  - On MPI-IO level → E10
  - On POSIX optimization level → SIONlib
- Test and combine the approaches
- Step into production in preparation
  - Booster to be attached to JURECA Cluster in coming months
- Future: Modular Supercomputing
  - More modules to come…

Future: Modular Design Principle



First Step: JURECA will be enhanced by a highly scalable Module