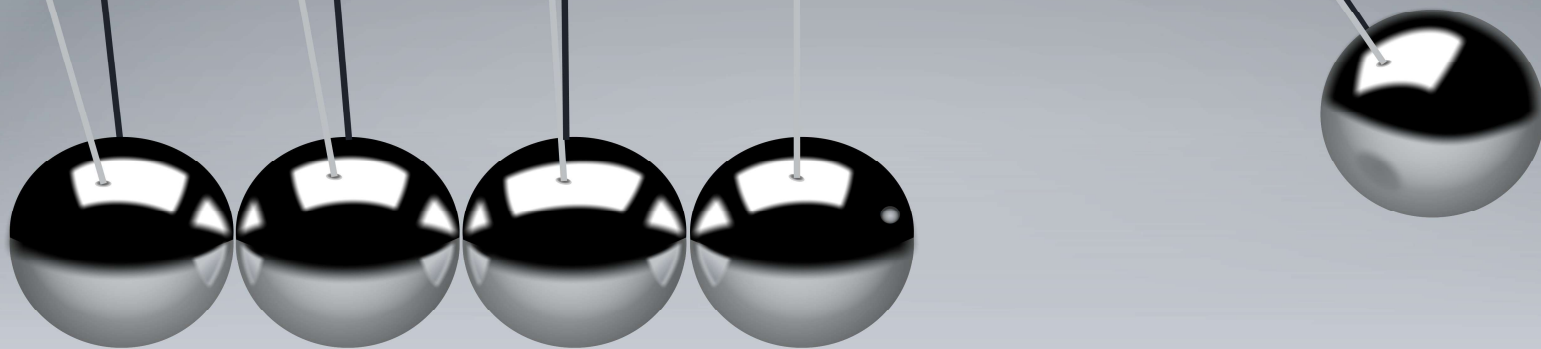


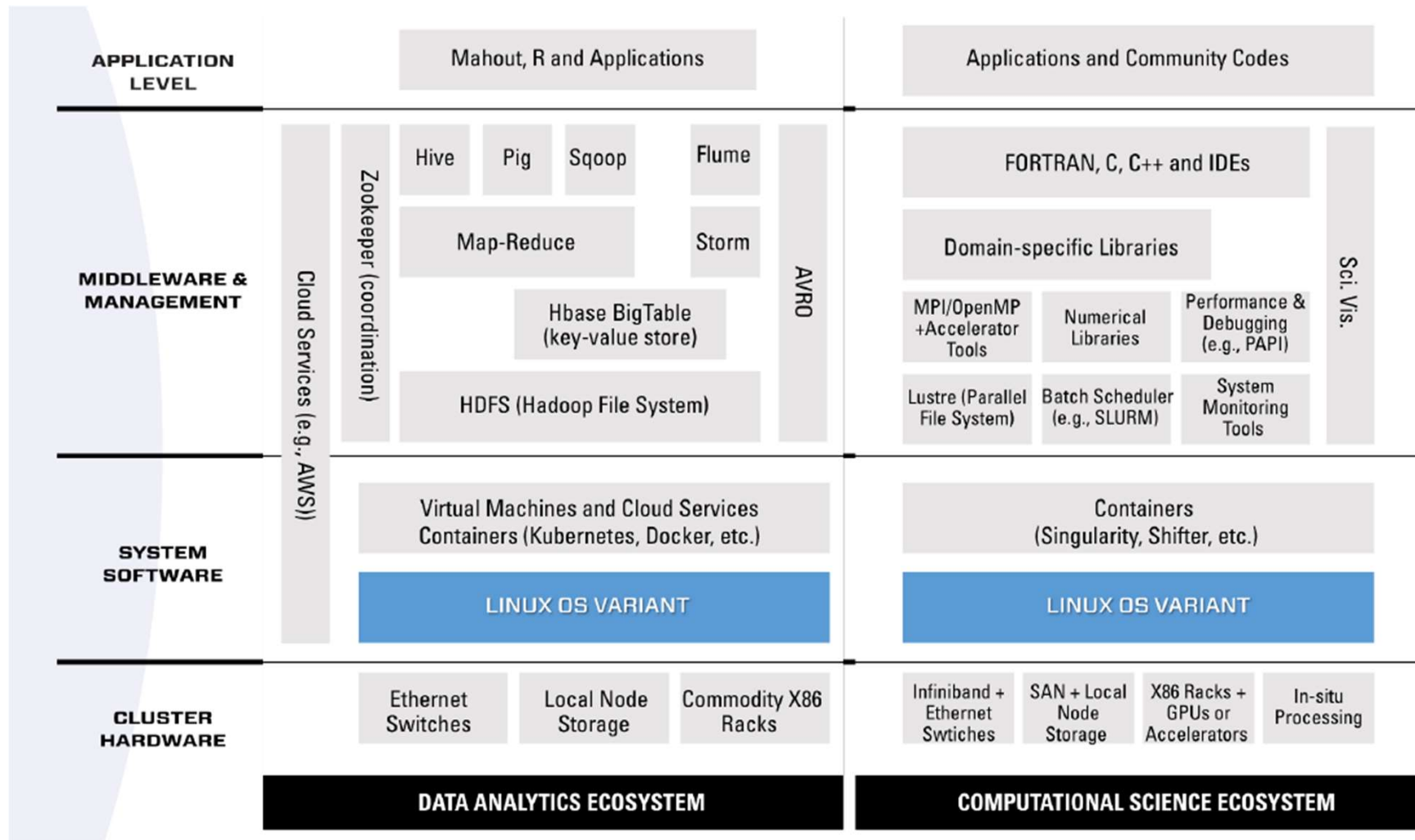
HPC and Big Data, A marriage of convenience?



María S. Pérez
mperez@fi.upm.es

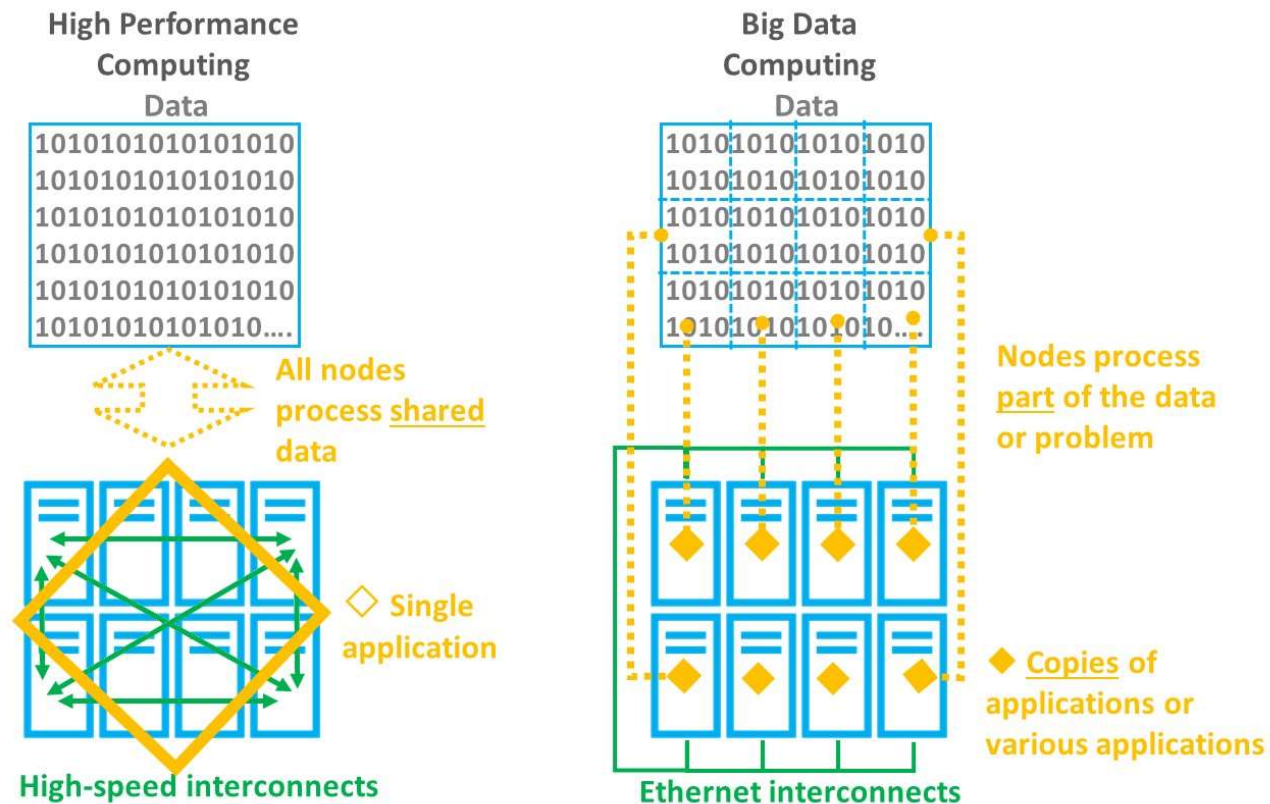
- **Introduction**
- Context
- General convergence HPC-Big Data problem
- Convergence HPC-Big Data at storage level
- Study and evaluation
- Conclusions

Divergence DA-CS



Source: *Big Data and Extreme-Scale Computing*, BDEC

Divergence at architecture level



Source: "Creating synergies across HPC & Big Data platforms", BDVA-ETP4HPC White Paper

Historical differences BD-HPC

	Typical workload	Design principles
Big Data	Data-intensive applications Most time is used for I/O and data management	Optimized for cost Less priority for performance
HPC	Compute-intensive applications Most time is used for computing	Optimized for performance Less priority for cost



[European Commission](#) > [Strategy](#) > [Digital Single Market](#) > [Policies](#) >

Digital Single Market

POLICY

The European High-Performance Computing Joint Undertaking - EuroHPC

PAGE CONTENTS

Useful links

The European High-Performance Computing Joint Undertaking (EuroHPC JU) will pool European resources to develop top-of-the-range exascale supercomputers for processing big data, based on competitive European technology.

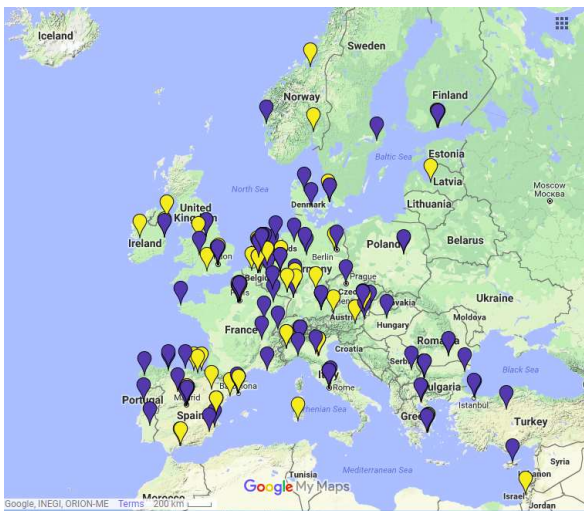
About HPC

[Policies](#) [+](#)

[Blog posts](#)

[News](#)

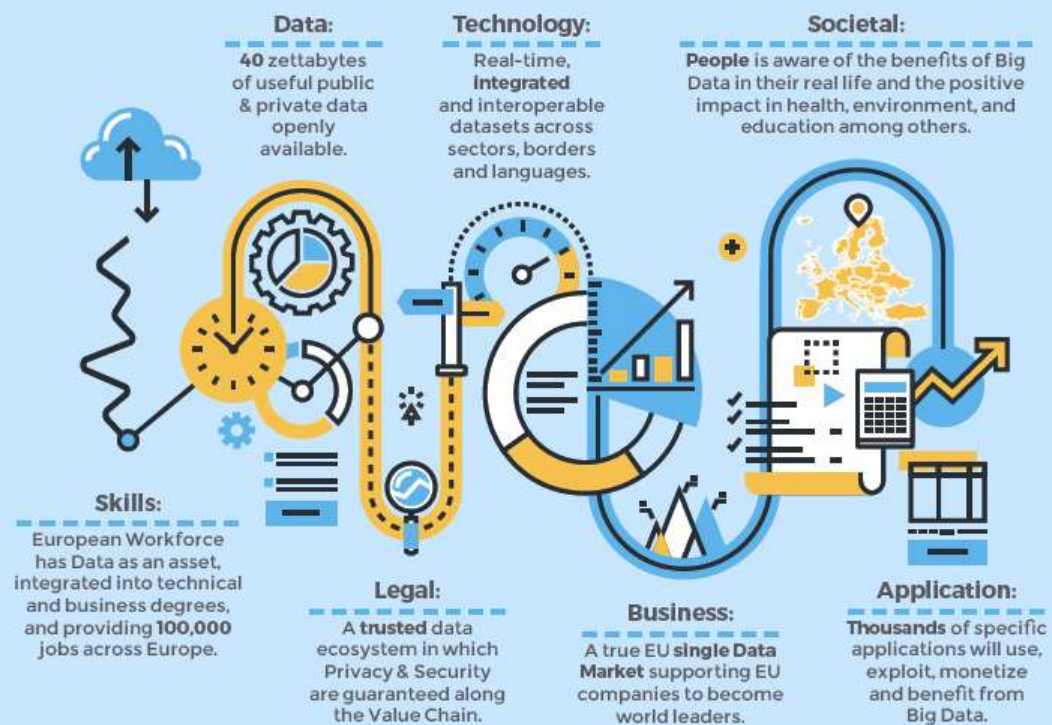
- Introduction
- **Context**
- General convergence HPC-Big Data problem
- Convergence HPC-Big Data at storage level
- Study and evaluation
- Conclusions



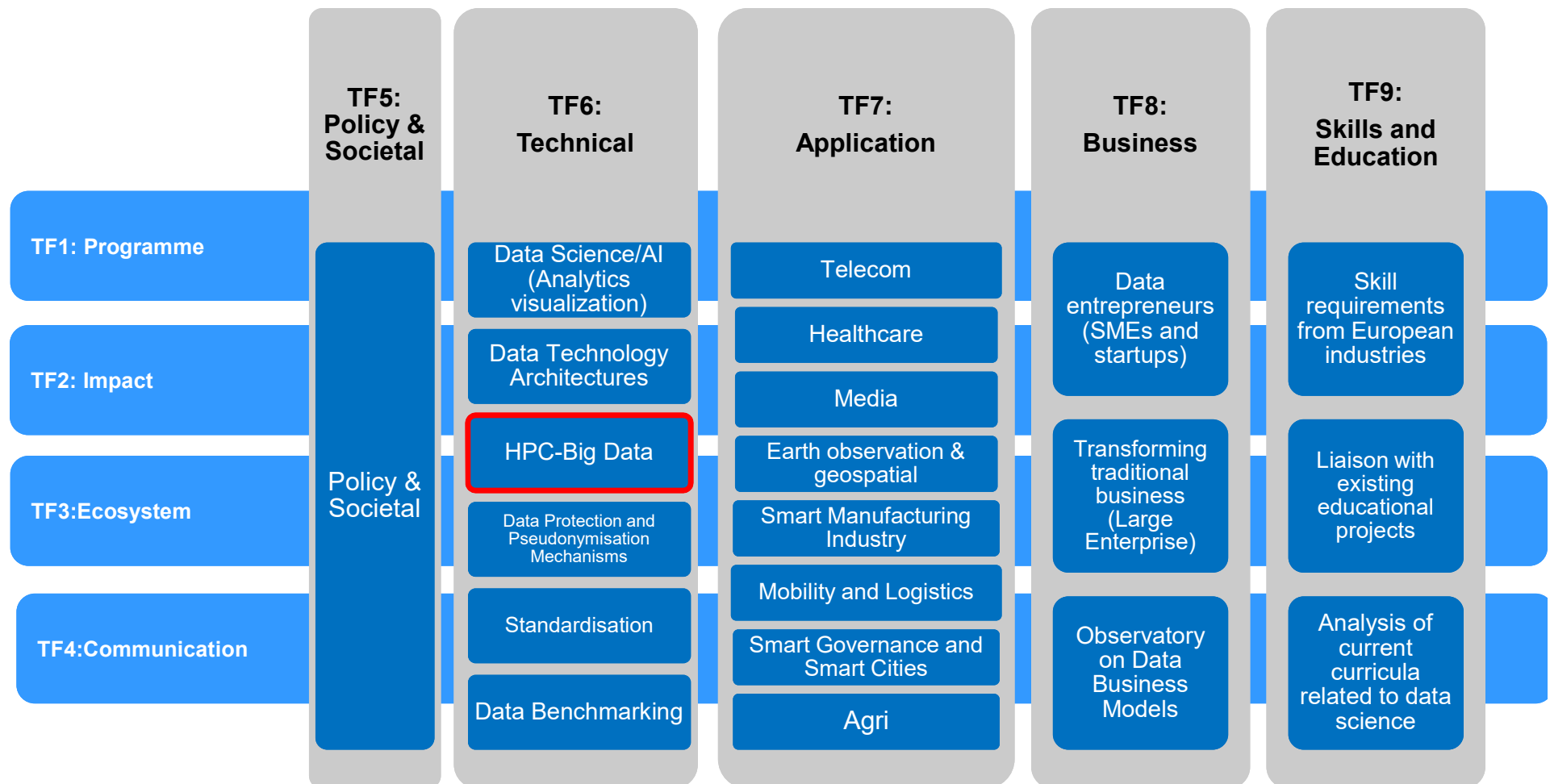
BDVA (~200) members include large industries, SMEs, research organisations and data users and providers to support the development and deployment of the EU Big Data Value Public-Private Partnership with the European Commission

BDVA focuses its activities on updating the multi-annual roadmap and on providing regular advice to enable the European Commission to prepare, draft and adopt the periodic Work Programmes, as well as on delivering Data Innovation Recommendations, developing Big Data Value Ecosystem, guiding Standards, and, facilitating Know-how exchange.

Big Data Value Vision for 2020



BDVA Task Forces



EuroHPC Joint Undertaking

<https://eurohpc-ju.europa.eu>

EuroHPC – Activities



EuroHPC
Joint Undertaking

Infrastructure & Operations	R&I, Applications & Skills	JU Admin/Running costs
HPC Ecosystem		
~270	min 180	10
~290	~186	10
560	392	20
0	~420 (in kind)	2

■ Infrastructure + Operations

Acquisition of 2 pre-exascale machines and several (tbd) mid-range machines

■ Applications & Skills + R&I

R&I, exascale technologies and systems (incl. low-power processor); applications

■ JU Admin/running costs

■ JU Operation: 2019 to 2026

486 m€	EC
486 m€	Participating States
972 m€	Total
422 m€	Private Members

<https://eurohpc-ju.europa.eu/documents/EuroHPC-Work-Plan-2019.pdf>

CABAHLA-CM: Convergence BD-HPC: From sensors to applications

- Project funded by the Community of Madrid, grants for R&D activities between research groups in technology and biomedicine (2019-2022)
- 4 groups:
 - ArTeCS, Universidad Complutense de Madrid
 - ARCOS, Universidad Carlos III de Madrid
 - SciTrack, Centro de Investigaciones Energéticas, Medioambientales y Tecnológicas
 - OEG, Universidad Politécnica de Madrid
- Goal: Improve the integration of HPC and Big Data paradigms
 - Computing and data intensive platform
 - Two use cases: capturing and modeling sensor data for the prediction of solar radiation with high spatio temporal resolution and processing massive data in brain's medical images

- Introduction
- Context
- **General convergence HPC-Big Data problem**
- Convergence HPC-Big Data at storage level
- Study and evaluation
- Conclusions

Lower <-----> higher

Data Intensive

Traditional Big Data



Data-intensive workloads

[Example] Inferring new insights from big data-sets e.g. pattern recognition across suppliers, consumers, etc for data-driven insights and innovation

Extreme Data Analytics



Compute- and Data intensive workloads:

[Example] Reshaping healthcare through advanced analytics and artificial intelligence – leading to predictive and personalized medicine

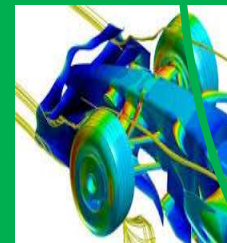
Enterprise IT



'Regular' workloads

[Example] Running the enterprise – HR, Legal, Payroll, finance, etc.

HPC



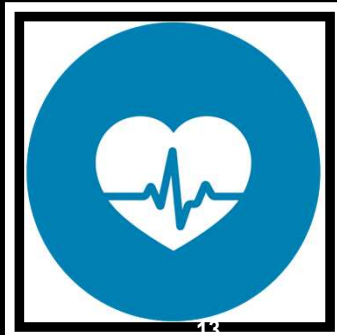
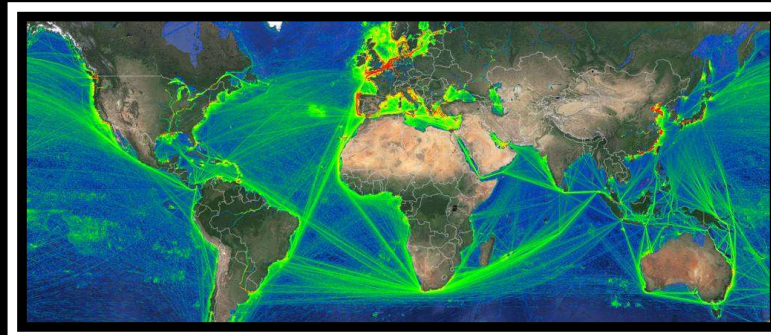
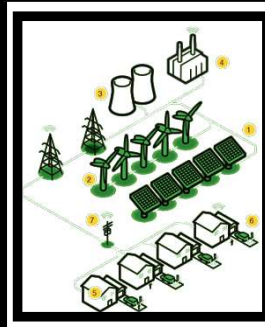
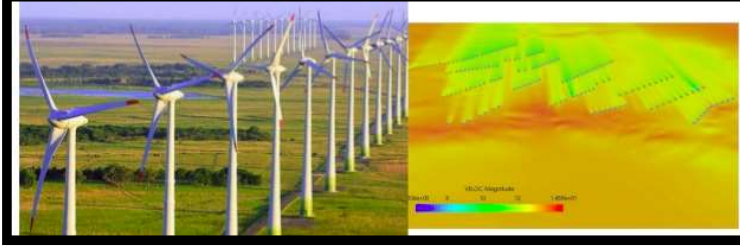
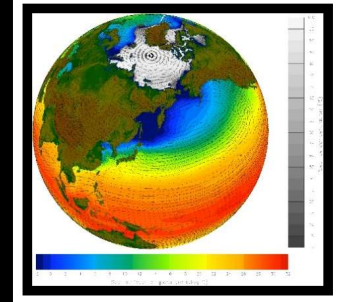
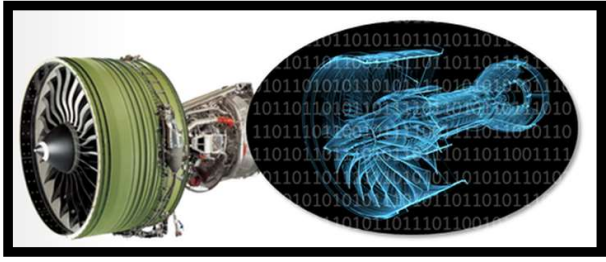
Compute-intensive workloads

[Example] Modelling and simulating focusing on interaction amongst parts of a system and the system as a whole e.g. product design

Compute Intensive
Lower <-----> higher

Source: Subgroup HPC-BD BDVA

Applications/Use cases



Source: Subgroup HPC-BD BDVA

HPC, Big Data and Deep Learning

	Supercomputing (SC)	Deep Learning (DL)	Big Data (BD)
Apps...	Boundary Interaction Services In-house, commercial & OSS applications [e.g. Paraview], Remote desktop [e.g. Virtual Network Computing (VNC)], Secure Sockets Layer [e.g. SSL certificates]	Framework-dependent applications [e.g. NLP, voice, image], Web mechanisms [e.g. Google & Amazon Web Services], Secure Sockets Layer [e.g. SSL certificates]	Framework-dependent applications [e.g. 2/3/4-D], Secure Sockets Layer [e.g. SSL certificates]
Middleware & MGMT	Processing Services Domain specific frameworks [e.g. PETSc], Batch processing of large tightly coordinated parallel jobs [100s – 10000s of processes communicating frequently with each other]	DNN training & inference frameworks [e.g. Caffe, Tensorflow, Theano, Torch], DNN numerical libraries [e.g. dense LA], DNNs, statistics, diagnostics [e.g. ?]	Machine Learning (traditional) [e.g. Mahout, Scikit-learn], Analytics / Statistics [e.g. Python, ROOT, R, Matlab, SAS, SPSS, Sci-Py], Iterative [e.g. Apache Hama], Interactive [e.g. Dremel, Drill, Tez, Impala, Shark, Presto, BlinkDB], Batch / Map-Reduce [e.g. MapReduce, YARN, Sqoop, Spark], Real-time/streaming [e.g. Flint, YARN, Druid, Pinot, Storm, Samza, Spark streaming]
	Model / Information Management Services Data Storage: Parallel File Systems [e.g. Lustre, GPFS, BeeGFS, PanFS, PVFS], I/O libraries [e.g. HDF5, PnetCDF, ADIOS]	Data Storage [e.g. HDFS, Hbase, Amazon S3, GlusterFS, Cassandra, MongoDB, Hana, Vora]	Serialization [e.g. Avro], Meta Data [e.g. HCatalog], Data Ingestion & Integration [e.g. Flume, Sqoop, Apache Nifi, Elastic Logstash, Kafka, Talend, Pentaho], Data Storage [e.g. HDFS, Hbase, Amazon S3, GlusterFS, Cassandra, MongoDB, Hana, Vora], Cluster Mgmt [e.g. YARN, MESO]
	Communication Services Messaging & Coordination [e.g. MPI/PGAS, direct fabric access], Threading [e.g. OpenMP, task-based models]	Messaging & Coordination [e.g. Machine Learning Scaling Library (MLSL)]	Messaging [e.g. Apache Kafka (streaming)]
	Workflow / Task Services Conventional compiled languages [e.g. C/C++/Fortran], Scripting languages [e.g. Python, Julia,]	Scripting languages [e.g. Python]	Workflow & Scheduling [e.g. Oozie], Scripting languages [e.g. Keras, Mocha, Pig, Hive, JAQL, Python, Java, Scala]
System SW	System Management & Security Services Domain numerical libraries [e.g. PETSc, ScaLAPACK, BLAS, FFTW], Performance & debugging [e.g. DDT, Vtune, Vampir], Accelerator APIs [e.g. CUDA, OpenCL, OpenACC], Data Protection [e.g. System AAA, OS/PFS file access control], Batch scheduling [e.g. SLURM], Cluster management [e.g. OpenHPC], Container Virtualization [e.g. Docker], Operating System [e.g. Linux OS Variant]	Batching for training [built into DL frameworks], Reduced precision [e.g. Inference engines], Load distribution layer [e.g. Round robin/load balancing for inference], Accelerator APIs [e.g. CUDA, OpenCL], LA numerical libraries [e.g. BLAS, LAPACK, ...], Virtualisation [e.g. Dockers, Kubernetes, VMware, Xen, KVM, HyperX], Operating System [e.g. Linux (RedHat, Ubuntu, etc.), (Windows?)]	Distributed Coordination [e.g. ZooKeeper, Chubby, Paxos], Provisioning, Managing & Monitoring [e.g. Ambari, Whirr, BigTop, Chukwa], SVM systems [e.g. Google Sofia, libSVM, svm-py, ...], Hardware Optimization Libraries [e.g. cuDNN, MKL-DNN, etc.]), Virtualisation [e.g. Dockers, Kubernetes, VMware, Xen, KVM, HyperX], Operating System [e.g. Linux (RedHat, Ubuntu, etc.), windows]
Hardware	Infrastructure Local storage [e.g. Storage & I/O nodes, NAS], Servers [e.g. CPU & Memory] [Gen Purpose CPU nodes, GPUs, FPGAs], Network [e.g. Infiniband & OPA fabrics]	Local storage [e.g. Local storage or NAS/SAN], Servers [e.g. CPU & Memory] [Gen Purpose CPU + GPU/FPGA, TPU], Network [e.g. Ethernet?]	Local storage [e.g. Direct attached Storage], Servers [e.g. CPU & Memory, [Gen Purpose CPU hyper-convergent nodes], Network [e.g. Ethernet fabrics]

Source: “Creating synergies across HPC & Big Data platforms”, BDVA-ETP4HPC White Paper

- Introduction
- Context
- General convergence HPC-Big Data problem
- **Convergence HPC-Big Data at storage level**
- Study and evaluation
- Conclusions

HPC at storage level

HPC application

POSIX file system

- Random reads and writes to file
- Folder/file hierarchies
- Permissions
- Atomic file rename
- Multi-user protection

POSIX

Random reads and writes to file
Folder/file hierarchies
Permissions
Atomic file rename
Multi-user protection

POSIX

Random reads and writes to file

POSIX

Random reads and writes to file Objects

HPC at storage level



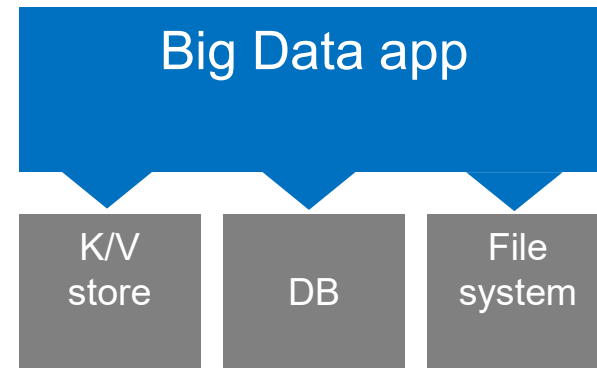
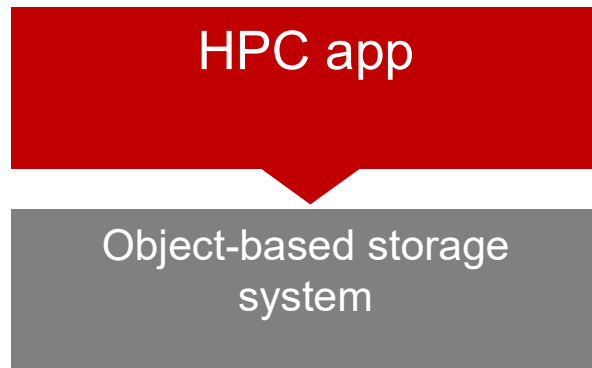
HPC and Big Data

HPC app

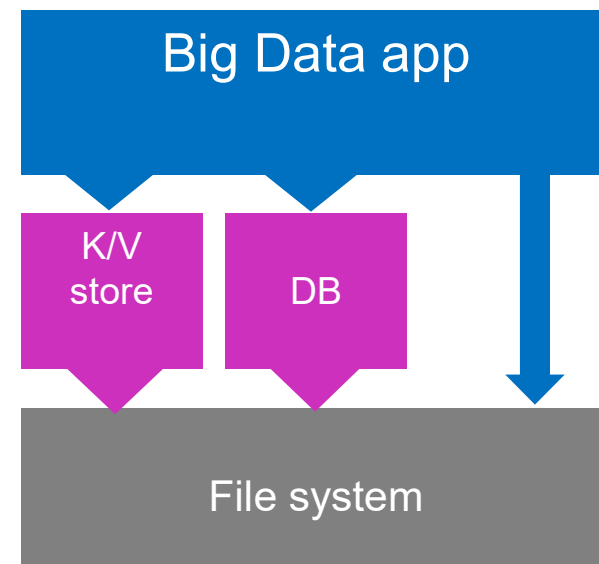
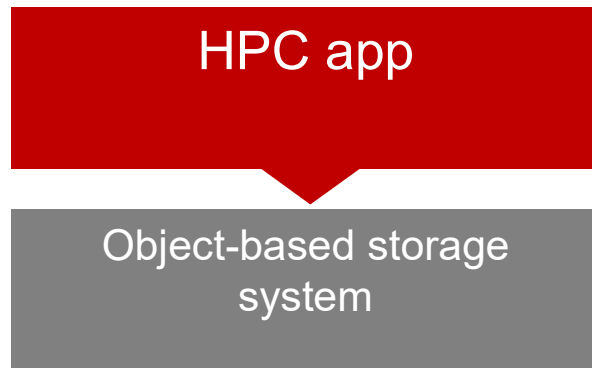
Object-based storage
system

Big Data app

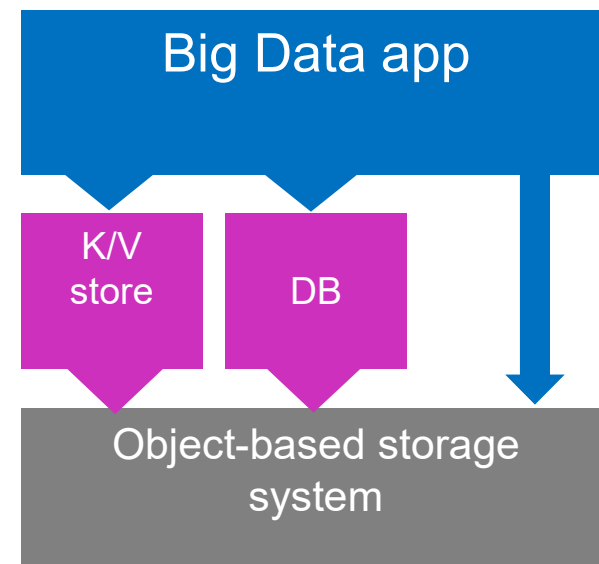
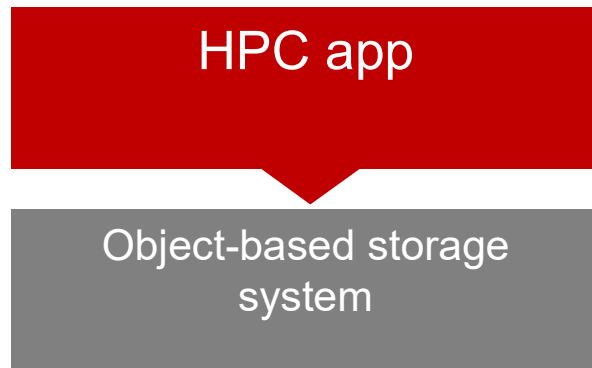
HPC and Big Data



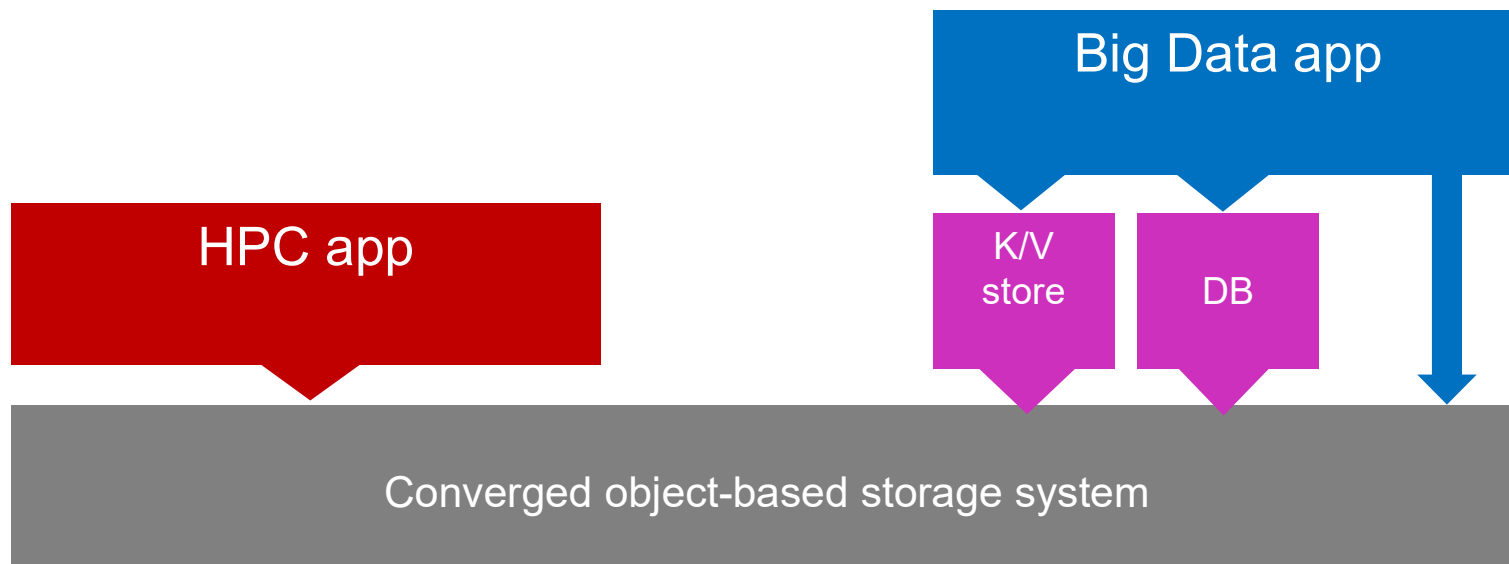
HPC and Big Data



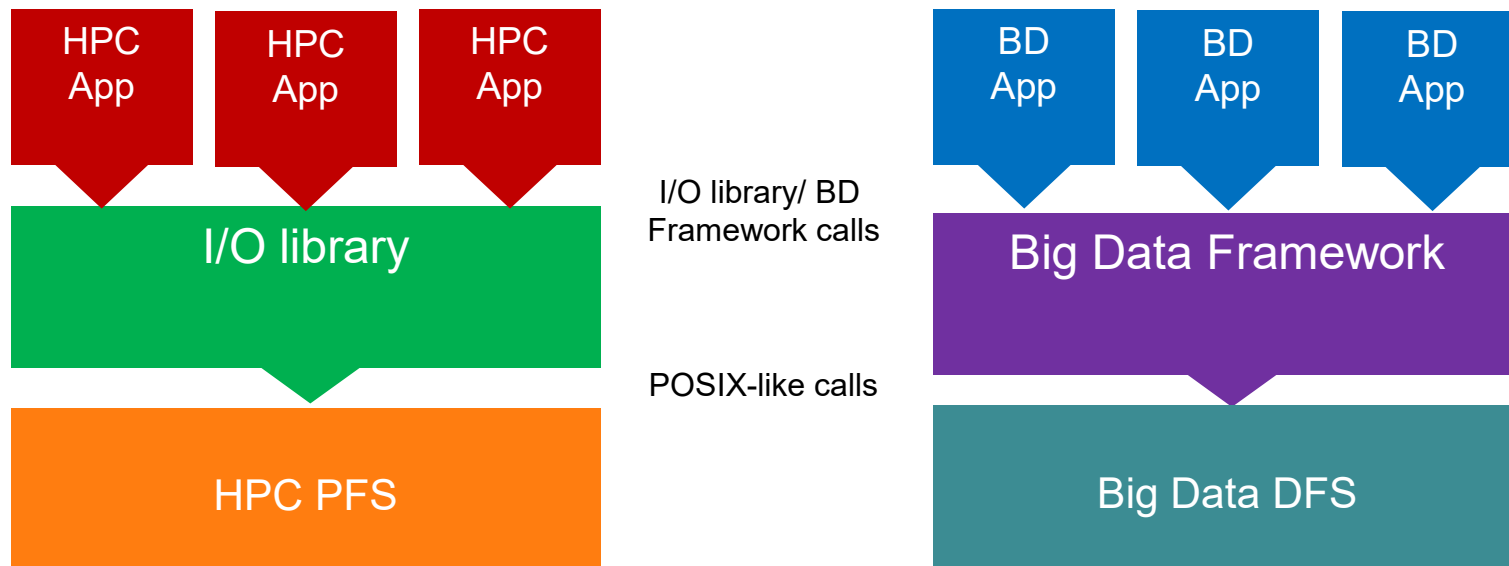
HPC and Big Data



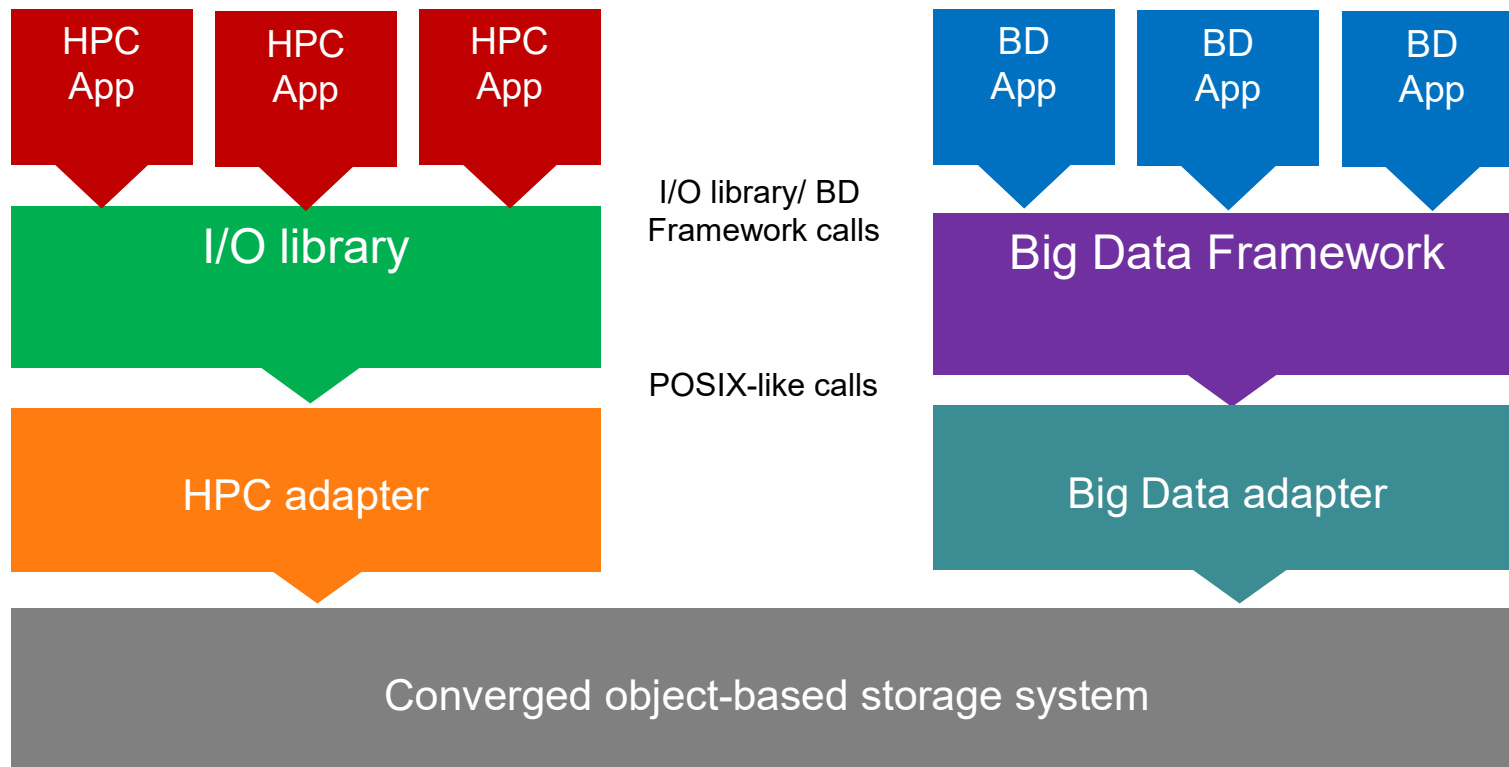
HPC and Big Data



Actual storage stack



Actual storage stack



- Introduction
- Context
- General convergence HPC-Big Data problem
- Convergence HPC-Big Data at storage level
- **Study and evaluation**
- Conclusions

Object-oriented primitives

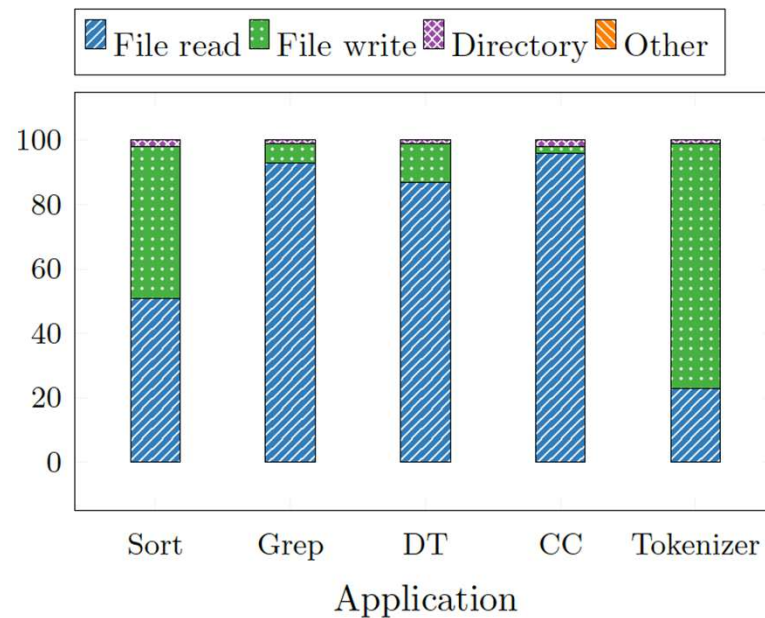
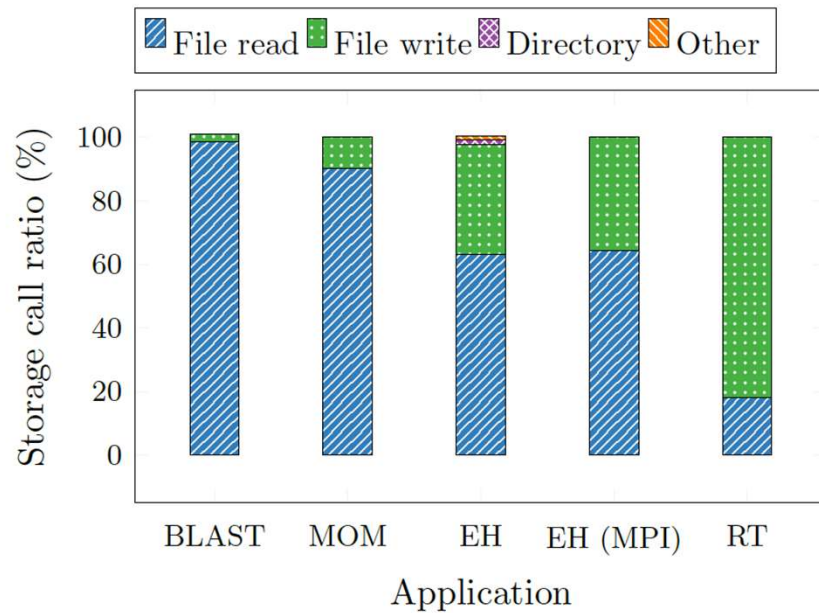
- Object access: random object read, object size
- Object manipulation: random object write, truncate
- Object administration: create object, delete object
- These operations are similar to those permitted by the POSIX-IO API on a single file
- Directory-level operations do not have their object-based storage counterpart (flat nature of these kinds of systems):
 - Low number of them
 - Emulated using the scan operation (far from optimized, but compensated by the gains permitted by using a flat namespace and simpler semantics)

Representative set of HPC/BD Apps

Plataform	Application	Usage	Total reads	Total writes	R/W ratio	Profile
HPC/MPI	mpiBLAST	Protein docking	27.7 GB	12.8 MB	$2.2 \cdot 10^3$	Read-intensive
	MOM	Oceanic model	19.5 GB	3.2 GB	6.09	Read-intensive
	ECOHAM	Sediment propagation	67.4 GB	71.2 GB	0.94	Balanced
	Ray Tracing	Video processing	0.4 GB	9.7 GB	$4.1 \cdot 10^{-2}$	Write-intensive
Cloud/Spark	Sort	Text processing	5.8 GB	5.8 GB	1.00	Balanced
	Connected Component	Graph processing	13.1 GB	71.2 MB	$1.8 \cdot 10^2$	Read-intensive
	Grep	Text processing	55.8 GB	863.8 MB	66.14	Read-intensive
	Decision Tree	Machine Learning	59.1 GB	4.7 GB	12.57	Read-intensive
	Tokenizer	Text processing	55.8 GB	235.7 GB	0.23	Write-intensive

Pierre Matri, Yevhen Alforov, Álvaro Brandón, María S. Pérez et al. Mission possible: Unify HPC and Big Data stacks towards application-defined blobs at the storage layer. Future Generation Computer Systems, In press.

Ops distribution

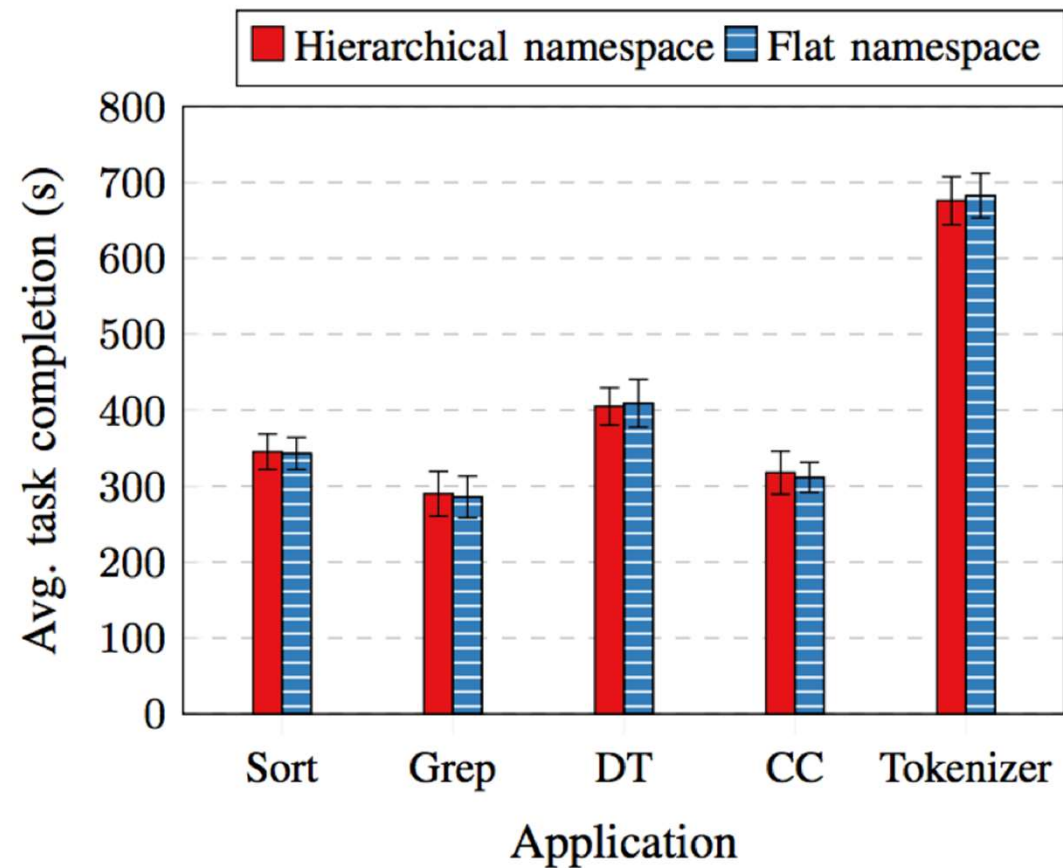
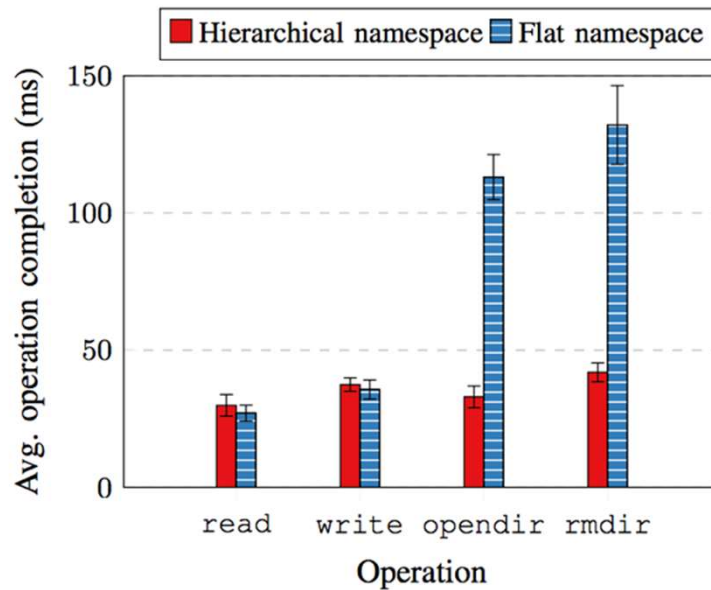


Directory operations (BD App)

Operation	Action	Oper. count
mkdir	Create directory	43
rmdir	Remove directory	43
opendir (Input data directory)	Open/List directory	5
opendir (other directories)	Open/List directory	0

Original operation	Rewritten operation
create(/foo/bar)	create(/foo__bar)
open(/foo/bar)	open(/foo__bar)
read(fd)	read(bd)
write(fd)	write(bd)
mkdir(/foo)	-----
opendir(/foo)	scan(/), return all files matching foo__*
rmdir(/foo)	scan(/), remove all files matching foo__*

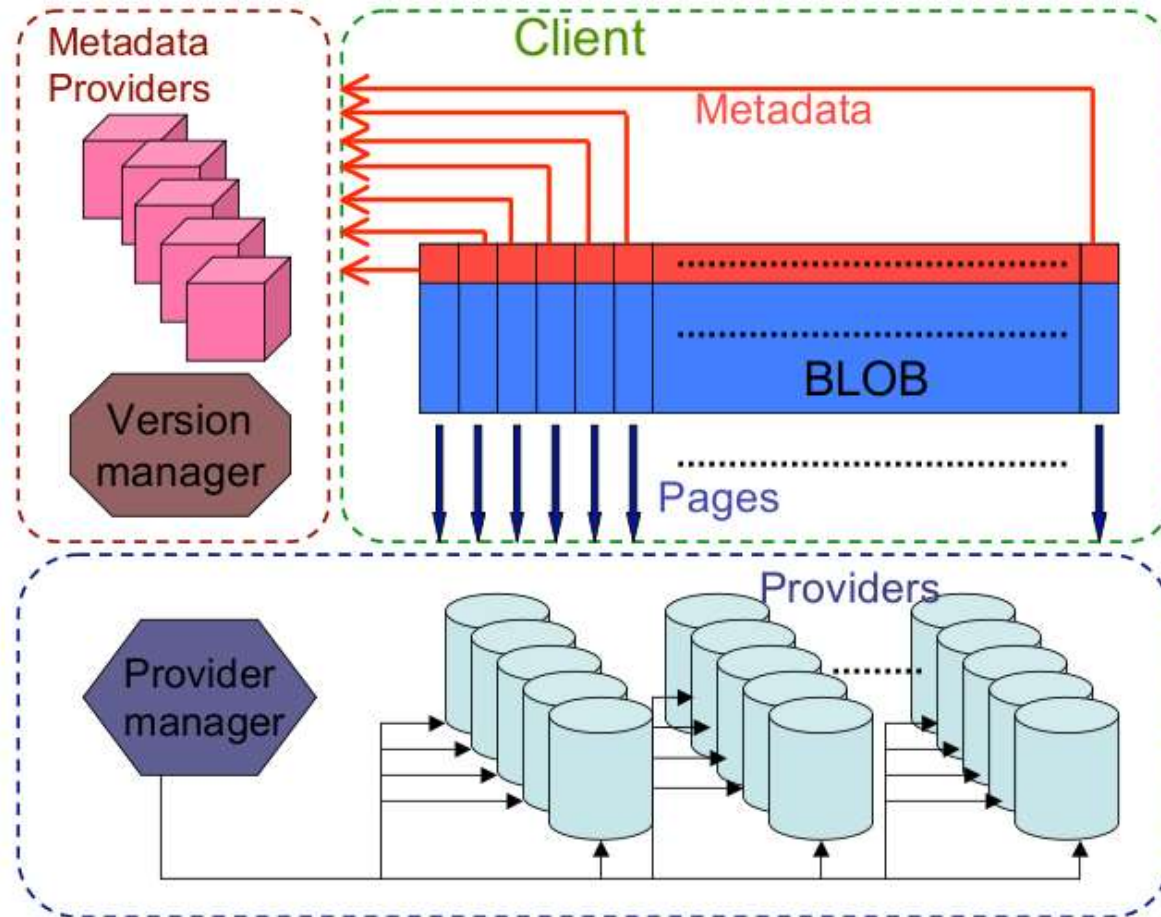
Influence of directory operations



BlobSeer/RADOS vs Lustre (HPC) and HDFS/Ceph (BD)

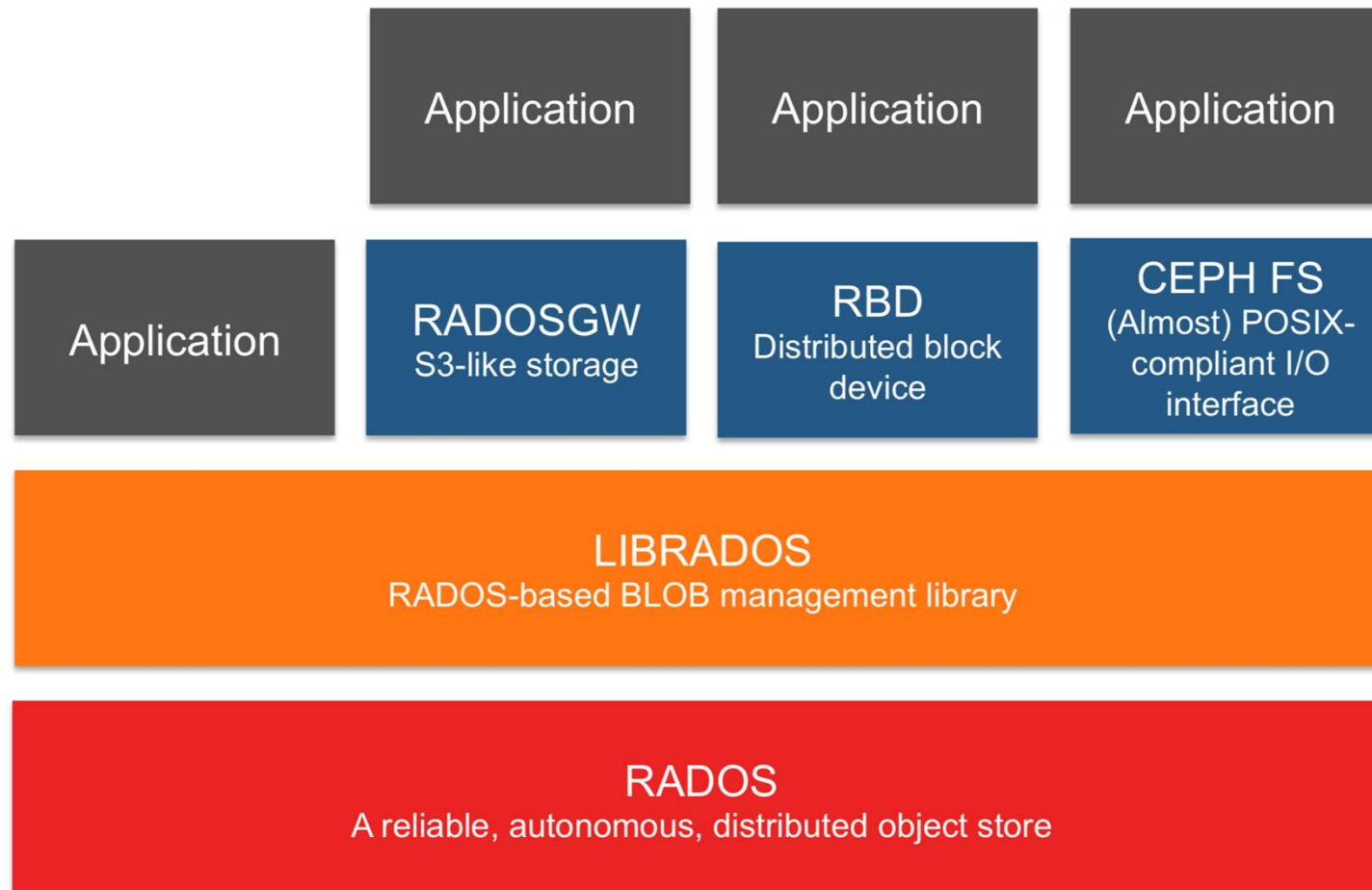
- Grid'5000 experimental testbed distributed over 11 sites in France and Luxembourg (parapluie cluster, Rennes)
- Each node: 2 x 12-core 1.7 Ghz 6164 HE, 48 GB of RAM and 250 GB HDD.
- HPC Apps: Lustre 2.9.0 and MPICH 3.2 [67], on a 32-node cluster (InfiniBand)
- BD Apps: Spark 2.1.0, Hadoop / HDFS 2.7.3 and Ceph Kraken, on a 32-node cluster (Gigabit Ethernet)

BlobSeer

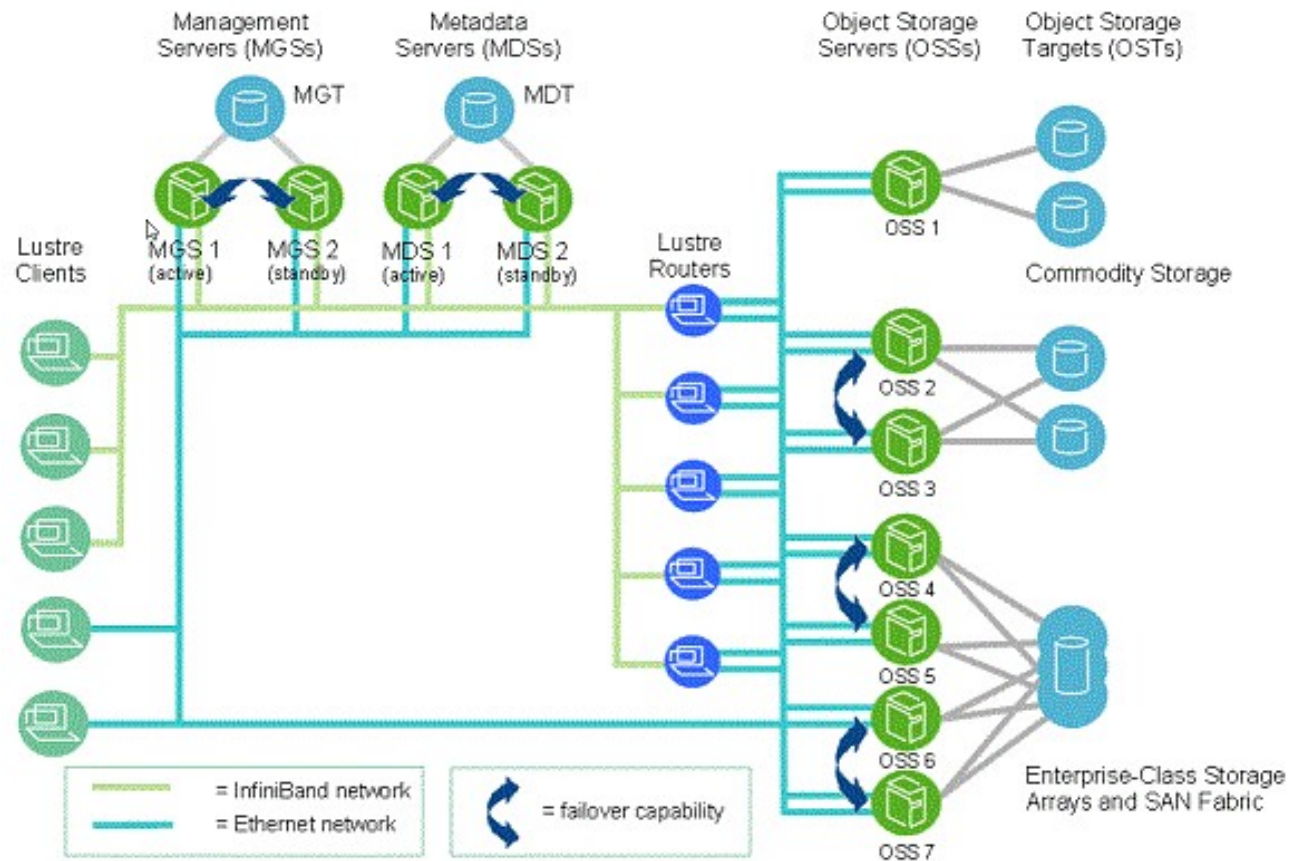


Bogdan Nicolae; Gabriel Antoniu; Luc Bougé; Diana Moise; Alexandra Carpen-Amarie. 2011. BlobSeer: Next-generation data management for large scale infrastructures. J. Parallel Distrib. Comput. 71, 2 (February 2011), 169-184.

RADOS/Ceph



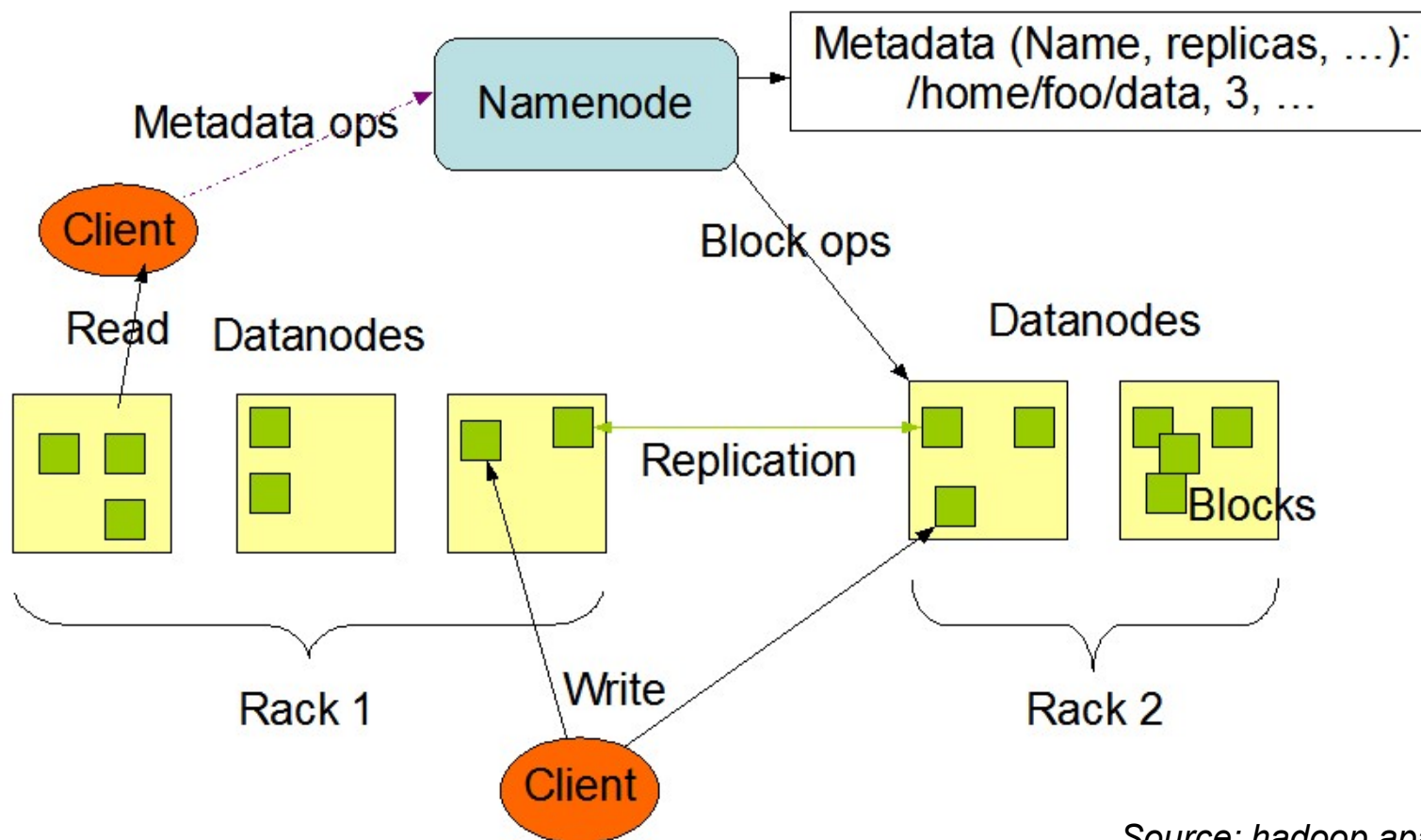
Lustre



Source: lustre.org

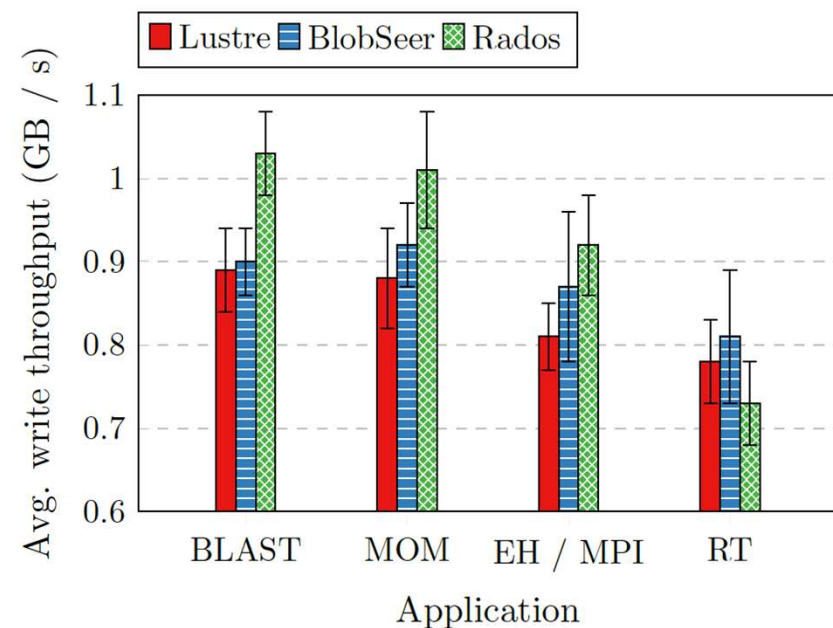
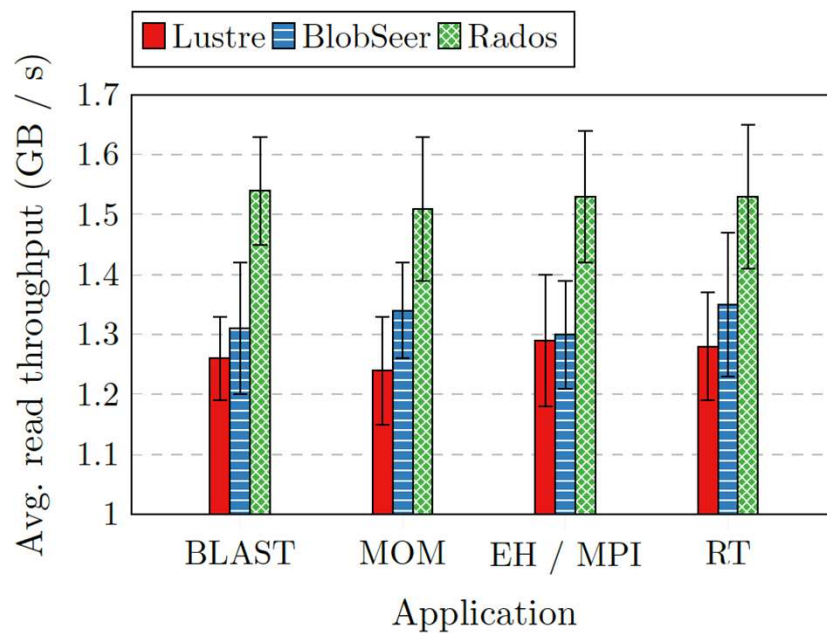
HDFS

HDFS Architecture

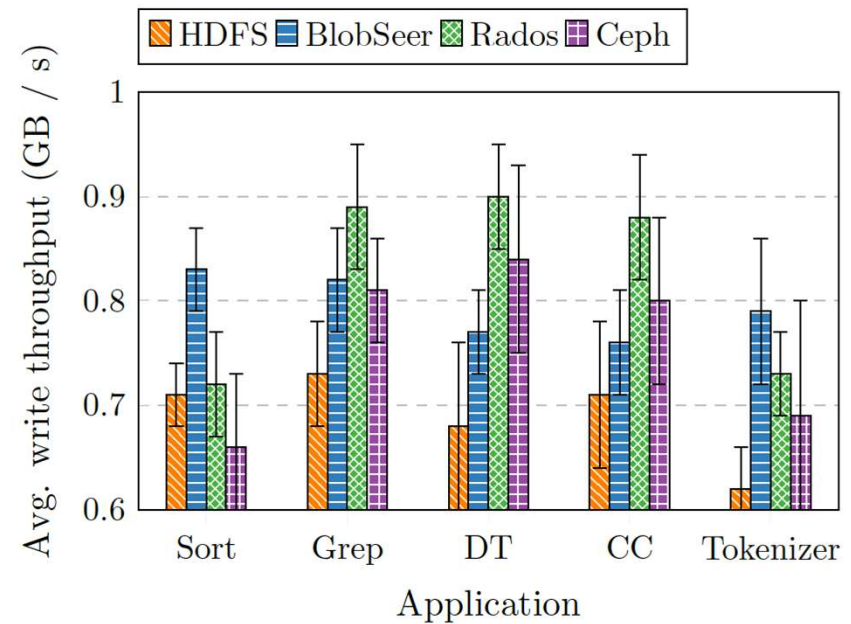
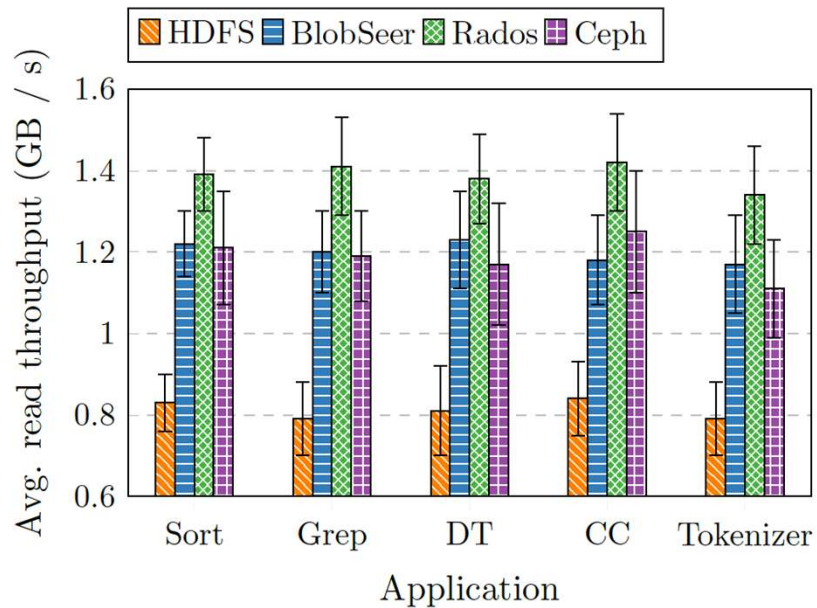


Source: hadoop.apache.org

HPC Apps



BD Apps



Study insights

- The **convergence at storage level is possible by using object-based storage systems**, achieving an improvement in the performance for both platforms (HPC and Cloud)
- By using objects, it is possible to achieve a **maximum improvement of 32%**
 - Mainly because of the **flat namespace**
 - Rados: direct reads and simple and decentralized metadata management (high performance for read operations)
 - BlobSeer: multi-version concurrency control supports high-performance write operations with highly concurrent workloads (high performance for write operations)
- **Issues of both systems:**
 - Although the **Rados** performance is excellent when the write contention is low, its **lock-based concurrency control** limits the performance of highly concurrent use cases.
 - The multi-version concurrency control in BlobSeer provides good performance at write level, but the **distributed metadata in BlobSeer** provides a significant read latency

Týr

- Can we achieve both systems benefits?
- Apart from combining the best features of Rados and BlobSeer, there is a significant set of use cases with stricter consistency semantics
 - Indexing and data aggregation (E.g., ALICE CERN LHC experiment)
 - Distributed shared logs (E.g., Computational steering + in-situ visualization)

Týr

One problem...

A scientific monitoring service, monitoring the
ALICE CERN LHC experiment:

- Ingests events at a rate of up to 10⁹ events per second
- Produces more than 10⁹ data points per second

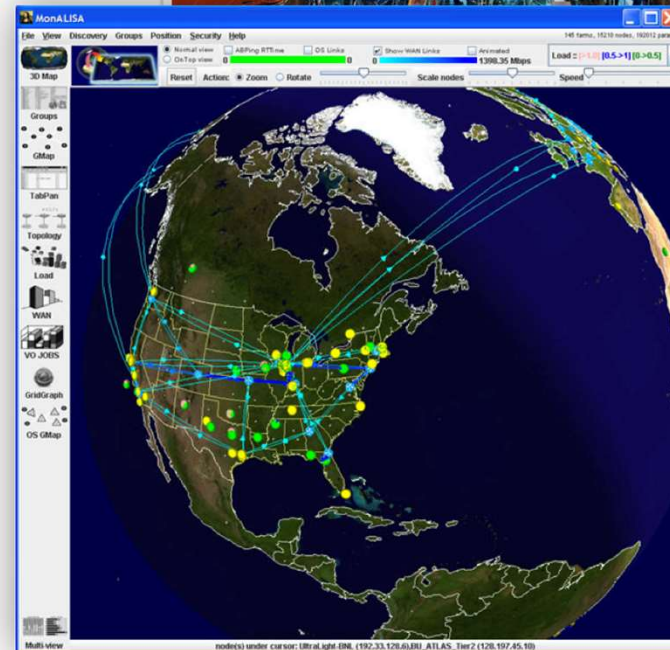
Computes 35.000+ events per second

Current load

týr
a (converged) blob store with built-in transactions

Visualization support

Leads
to stability



Pierre Matri; Alexandru Costan; Gabriel Antoniu; Jesús Montes; María S. Pérez. "Týr: Blob Storage Systems Meet Built-In Transactions". SC '16 Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. Article n. 49, Best student paper award finalist

Týr

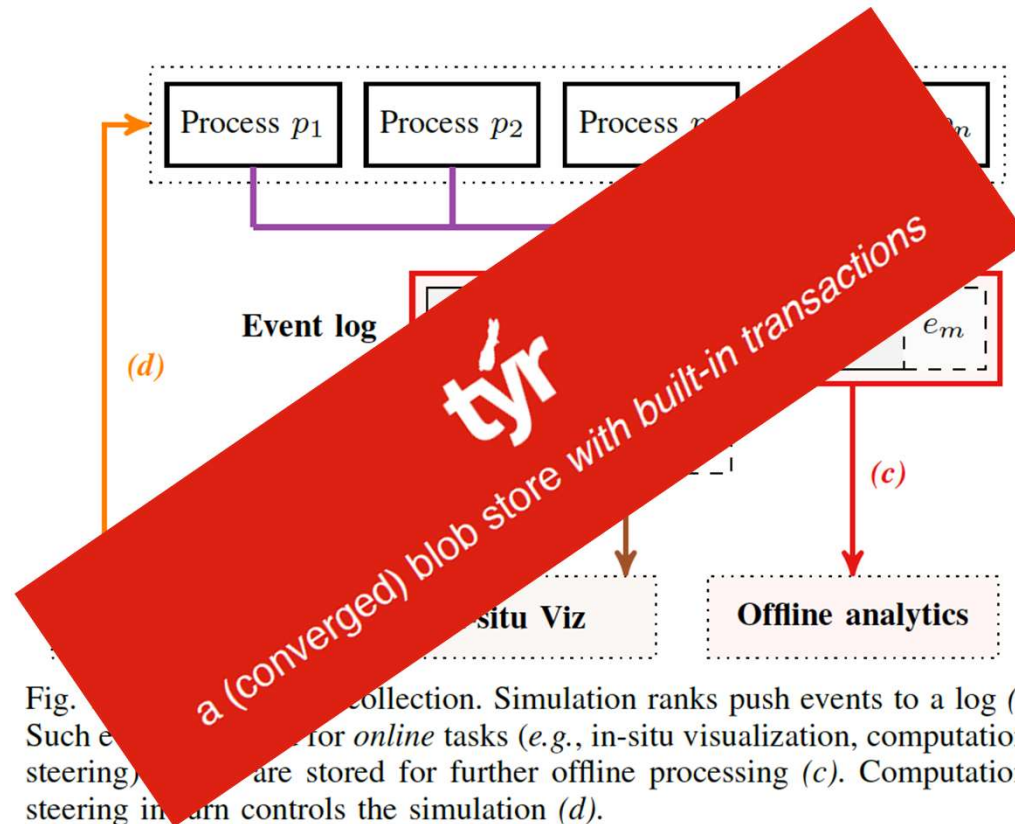


Fig. 1. Týr architecture. Simulation ranks push events to a log (a). Such events are then processed for *online* tasks (e.g., in-situ visualization, computational steering) or stored for further *offline* processing (c). Computational steering in turn controls the simulation (d).

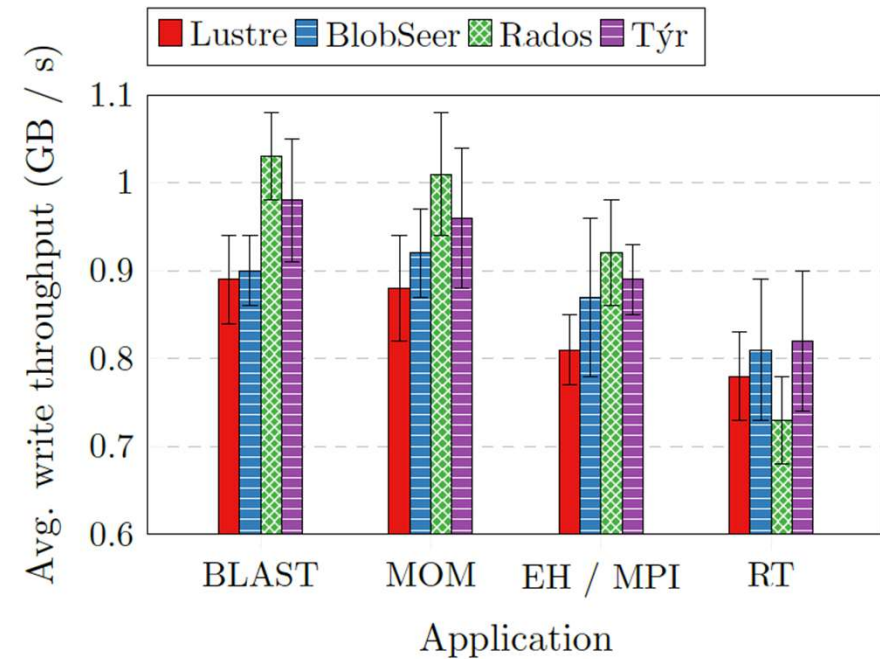
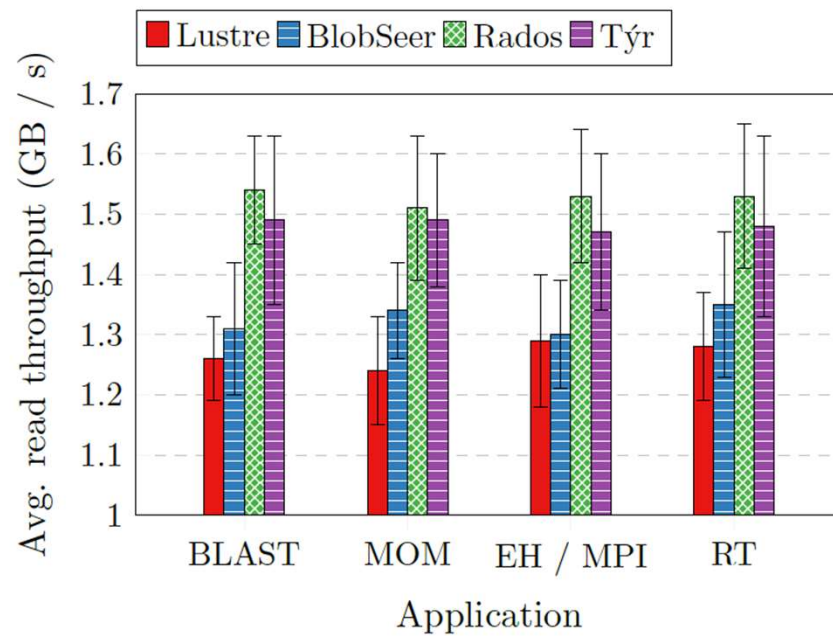
Pierre Matri; Philip Carns; Robert Ross; Alexandru Costan; María S. Pérez; Gabriel Antoniu; "SLoG: A large-scale Logging Middleware for HPC and Big Data convergence". ICDCS'2018. pp. 1507-1512, 2018.

Týr design

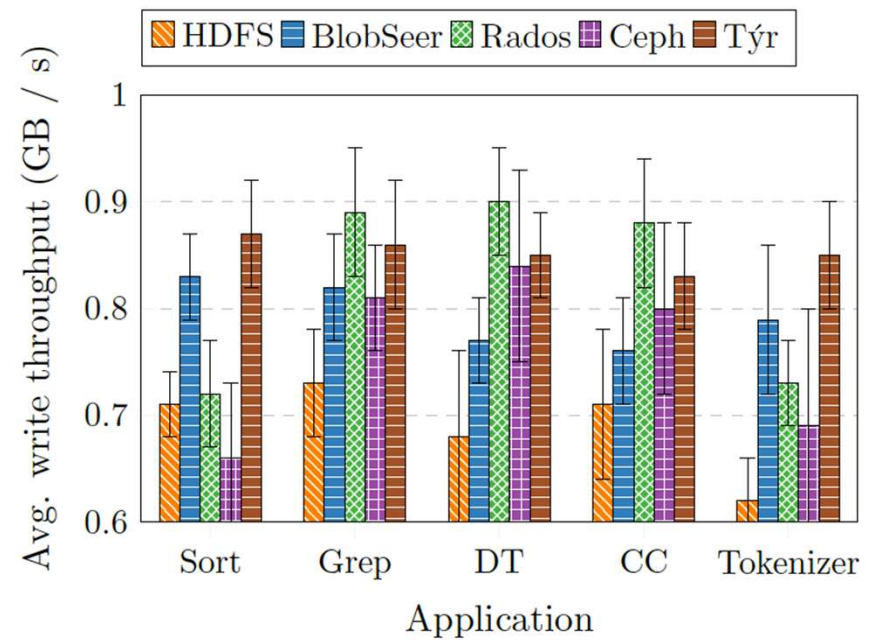
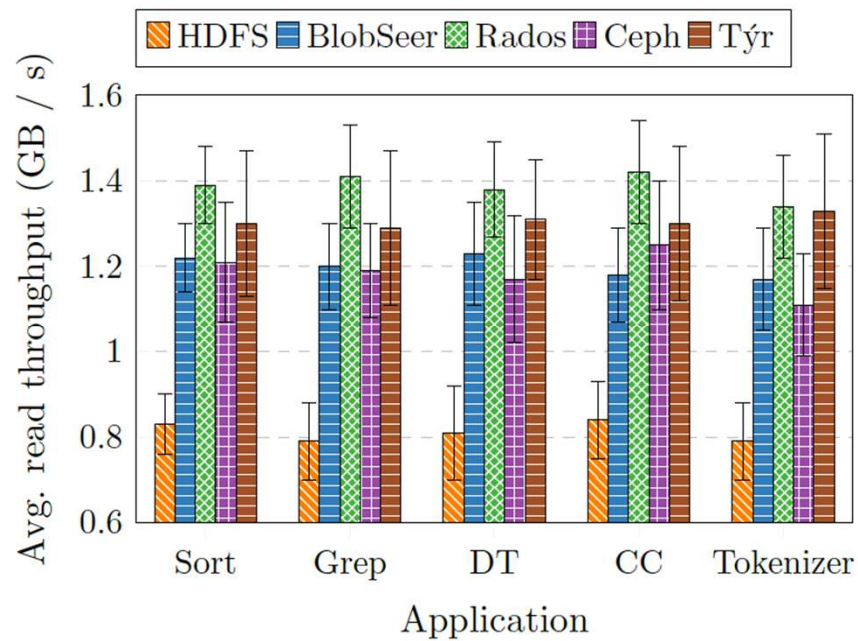
- Predictable data distribution
 - Combining *data striping* and *consistent hashing* techniques
 - Avoiding the use of a centralized metadata server
- Transparent multi-version concurrency control
 - *Versioning* at *chunk* level
- Lightweight ACID transactions
 - By using the transactional protocol Warp*
 - By using chains of the servers involved in transactions and dependency graphs
- Atomic transformation operations
 - Efficient *read-modify-write* ops
 - Particularly interesting for simple transformation operations (arithmetic, bit-wise)
 - The client does not send the new data to be written, but the modification, avoiding *two-round trips*
- Software prototype with around 25,000 Rust and GNU C code lines

* R. Escrivá, B. Wong and E. Sirer. Warp: Lightweight multi-key transactions for key-value stores. arXiv preprint arXiv:1509.07815, 2015.

HPC Apps



BD Apps

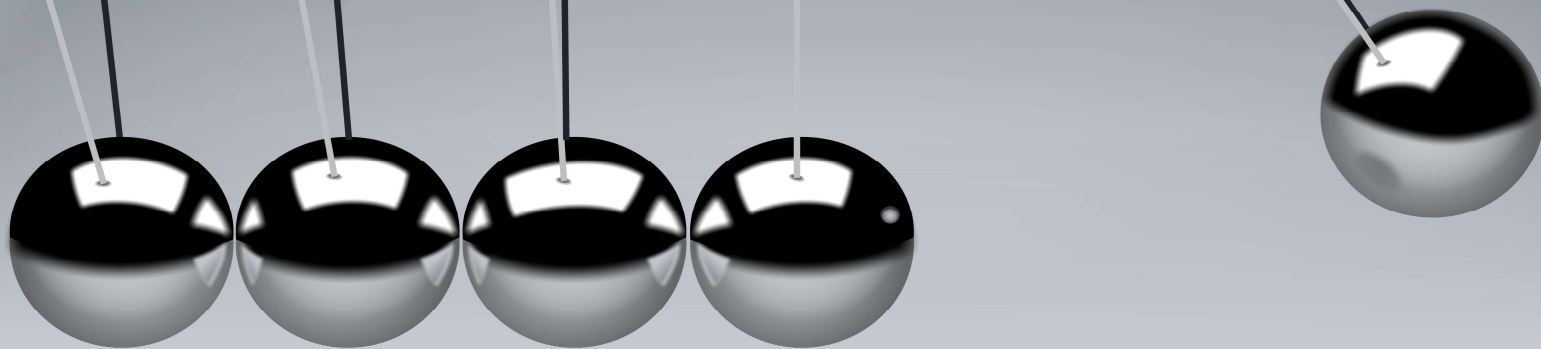


- Introduction
- Context
- General convergence HPC-Big Data problem
- Convergence HPC-Big Data at storage level
- Study and evaluation
- **Conclusions**

Conclusions

- **Týr outperforms BlobSeer and traditional file systems**, for both HPC and BDA applications
 - Non blocking writes, by means of multiversion concurrency control
 - Direct writes by using *consistent hashing*
- **Týr has some limitations in comparison with Rados**
 - **Except for write-intensive applications**, due to the efficiency of the multiversion concurrency control
 - **As a result of stronger consistency guarantees** (transactions)
- This is a **first step for the convergence between HPC and BDA** (at storage level)

HPC and Big Data, A marriage of convenience?



María S. Pérez
mperez@fi.upm.es