

Julian Kunkel

Rethinking the Meaning of Storage in the AI Era

Storage: Just another HPC Component or Part of Application-Driven Design?

About Me – Prof. Dr. Julian Kunkel

- Deputy Head of GWDG
  High Performance Computing Division
- Professor at the Institute of Computer Science
- Coordinator of the AI Service Centre KISSKI
- Founding member of the IO500 benchmarking

KISSKI
KI-Servicezentrum für sensible und kritische Infrastrukturen

GWDG
Gesellschaft für wissenschaftliche Datenverarbeitung mbH Göttingen

NHR-NORD@GÖTTINGEN

HammerHAI

Challenge + Pain Point: Complexity and Usability of Storage

Available options



User perspective



Outline

1 About us

2 Storage

3 Conclusion

(Storage) Requirements for HPC

- **Goal:** Enable highly concurrent, parallel workflows
- **Storage mapped to use cases** (not just capacity/performance)
  - *Scratch:* High-throughput, ephemeral compute islands
  - *Archive:* Long-term preservation of scientific results
  - *Project space:* Collaborative data sharing within teams
  - *Environment:* Reproducible, task-specific configurations
  - *Data pools:* Shared datasets across users (e.g., public benchmarks)
- **Enablers:** Abstraction: Metadata-driven, policy-based management
- **Key dimensions:** Persistency, performance, availability, usability

Requirements for AI

- **AI research $\neq$ AI production**
  - *Training:* Scaled HPC-like workload, random-read intensive storage
  - *Inference:* Low-latency model loading; often paired with RAG/DBs or KV-cache
- **GPU cost drives workflow design**
  - Expensive compute $\rightarrow$ interactive, iterative development is essential
  - Storage must be *seamless, persistent, and workflow-integrated*
- **External collaboration**
  - Easy data ingest/egress $\rightarrow$ S3-compatible interfaces
  - Cloud-native workflows $\rightarrow$ object storage as bridge
- **Exploration & prototyping**
  - High-performance NFS for rapid iteration
- **Aspects:** Usability, availability, persistency, security, performance
- **Need:** High-level platforms that abstract storage complexity for developers

HPC and AI Coexistence

- **Shared foundations**
  - Asynchronous batch execution, fast scratch, small-scale exploratory jobs
  - *But:* Divergent IT proficiency → require: self-service, containers, usability
- **Divergent workflows**
  - Production AI & end-user services require fault tolerance, and availability
- **Common challenges**
  - Balancing usability vs. availability vs. peak performance
  - Reproducible science workflows across environments
  - Frameworks that simplify data access (FAIR, data spaces, federations)
  - Observability: performance monitoring and issue diagnosis

Personal Opinion: HPC Future - In 5 Years

- Storage access, API access, application design is 99% managed by AI
- End users will not care about storage anymore!
- Actually, users don’t care about HPC vs. cloud vs. anything else
  - Users **rarely** access the HPC system via CLI
- Researchers will have **Integrated Science Environments (ISE)**
  - Fundamental change toward AI agent based access
  - Focus is the research application, interfaces are no obstacle any more
  - Interaction with HPC via ChatBot
- Only support staff AND application architect will care about storage
- Storage capabilities and integrated engines are the building blocks

Opportunities and Challenges

- **Paradigm shift: From component to enabler**
  - *Use-case-driven design:* Start with the workflow, then integrate “storage“
  - *Storage as invisible fabric:* Integrated into dev/prod ecosystems, not bolted-on
  - *Cloud + HPC convergence:* Compute continuum with unified data access
  - *Urgent computing:* High availability for time-sensitive AI/HPC tasks
- **Technical enablers**
  - AI-native formats + semantic catalogs
  - Integrated microservices (versioning, lineage, policy enforcement)

Outline

1 About us

2 Storage

3 Conclusion

Conclusions

- Storage is not the goal; it is the enabler
  Users want outcomes, not infrastructure
- GPU economics demand seamless, interactive workflows
  Storage may disappear into the application layer.
- Shift from HPC building blocks to application-driven design
- Usability, availability, and performance must coexist
- To integrated services - offloading of functionality and appliances
- Toward Agent-Based **Integrated Science Environment**
- **Let's build the AI ecosystems together.**
  An "AI operating system" where storage is integral, not optional