HPC-Systeme
HPC und Storage

Prof. Dr. Volker Gülzow

Dr. Yves Kemp

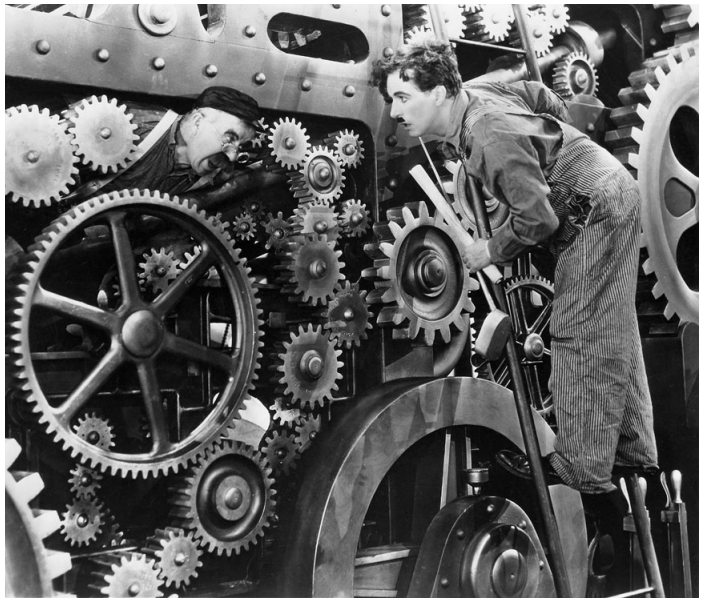
SS 2017

Storage und Storage-Systeme für HPC

- Wenn man „Computing“ in HPC eng auslegt, dann betrifft dies nur das “Rechnen“
- Häufig wird Storage und Storage-Nutzung in HPC Vorlesungen (und auch Planungen...) stiefmütterlich behandelt
- Für den erfolgreichen Aufbau eines kompletten HPC-Systems ist allerdings auch vernünftiger Storage notwendig
- Für die erfolgreiche Nutzung von HPC-Systemen sind einige Kenntnisse über Storage-Systeme und Storage-Nutzung wichtig

Zwei Sichten auf Storage:

Nutzer-Sicht auf Storage
Schwerpunkt der Vorlesung und Übungnächste Woche



Admin-Sicht auf Storage
Schwerpunkt dieser Vorlesung

http://selfstorageindavis.com

http://cinemalacrum.blogspot.de/2011/08/top-ten-thursday-silent-films.html

Storage != Data

- Mit „Data“ und „Data Access“ wird im HPC Vorlesungen häufig Daten und Datenzugrif im/auf Arbeitsspeicher oder L1/2/3 Caches der CPUs gemeint.
- Die Festplatte im Server eines HPC-Workernode spielt typscherweise eine untergeordnete Rolle

- Unter Storage verstehe ich in der Vorlesung grosser, zentraler Storage

Anwendungsfall von Storage im HPC

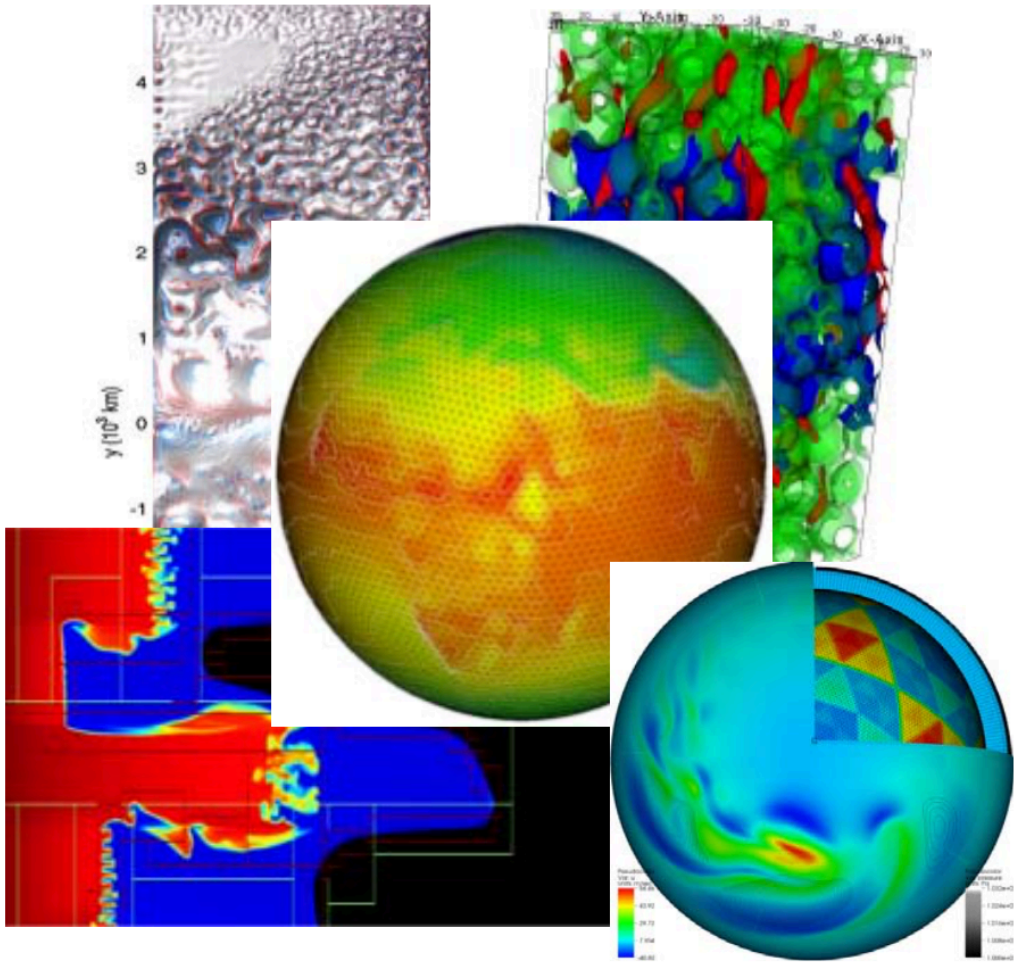
- Input-Daten für Simulation (oder Analyse)
- Lagerung von Output-Daten einer Simulation (oder Analyse)
- Projekt-Daten für Zwischenschritte
- Nutzer-“$HOME“

Anwendungsfall von Storage im HPC

- Checkpointing: Snapshot der aktuellen Berechnung auf Storage zu schreiben, um im Fall eines teilweisen oder ganzen Ausfalls vom letzten Snapshot aus weiterzurechnen

Storage & IO

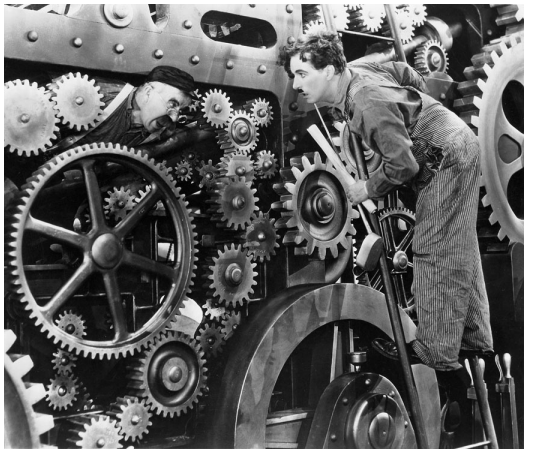
- In der wissenschaftlichen Community steigen die Anforderungen an IO kontinuierlich
- IO ist ein Bottleneck für viele Nutzer
- Erforderlicher Storage-Platz muss mit System-Speicher (RAM) skalieren
- Paralleles IO ist für grosse Nutzer unerlässlich
- Typische grosse HPC Cluster haben mehre 10 PB Storage



Diese und nächste Folien mit Material von Richard Gerber (NERSC)
https://www.olcf.ornl.gov/wp-content/uploads/2013/05/OLCF-Data-Intro-IO-Gerber-FINAL.pdf

Zwei Sichten auf Storage:

Nutzer:
Daten? Ja, Abbildung meines Systems im Code: Grid Zellen, Teilchen, ...



Admin:
Daten? Ja, das Zeugs was von einem Worker-Node auf einen Storage-Server Transferiert wird und dort auf Speichermedien liegt

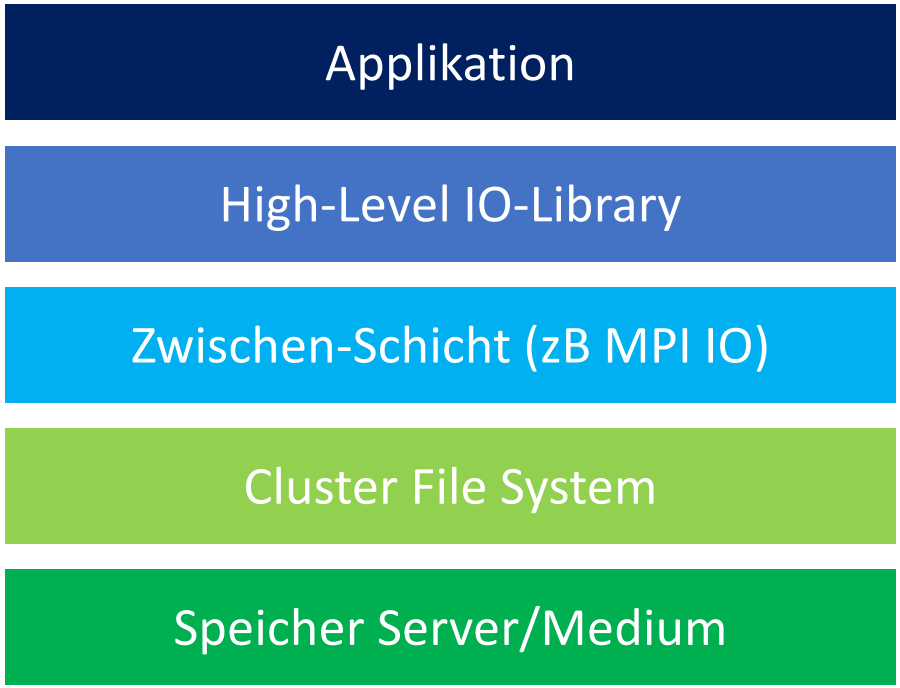
http://selfstorageindavis.com

http://cinemalacrum.blogspot.de/2011/08/top-ten-thursday-silent-films.html

Relationship: It‘s complicated

- Nutzer müssen ihre IO Muster verstehen - und ggf anpassen – um gute Performance zu erreichen
- Admins brauchen möglichst detailliertes Wissen über die Anforderungen der Nutzer um Storage-Systeme zu designen und zu tunen
- Manches kann durch spezialisierte IO-Libraries abgefedert werden

IO Hierachie



Welche Speicher Medien?

- Sehr schnell: Solid-State
  - FLASH / 3D-Xpoint (neue Technologie von Intel & Micron) / ...
  - Sehr schnell, sehr teuer, vergleichsweise wenig Kapazität
- Schnelle aber kleine rotierende Platten
  - SAS, typischerweise 2.5“ mit 10kRPM oder 15kRPM
  - 600 Gbyte – 1.8 TB aktuell typische Kapzitäten
- Nearline-SAS Platten
  - NL-SAS, 3.5“ mit 7.2kRPM
  - 6-10 Tbyte aktuell typische Kapazitäten
- Bänder
  - Langsam, gross, nur streaming IO, unhandlich, günstig (je nach Rechnung)

Welche Speicher Medien?

- Sehr schnell: Solid-State
  - In Storage-Systemen zB als Burst-Buffer, Cache oder Meta-Daten-Speicher benutzt
- Schnelle aber kleine rotierende Platten
  - Nutzung nimmt ab zugunsten von Solid-State
- Nearline-SAS Platten
  - Stellen typischerweise den Grossteil des Plattenplatzes
- Bänder
  - Eventuell Archiv oder Backup. Typischerweise nicht direkt aus dem Filesystem adressierbar

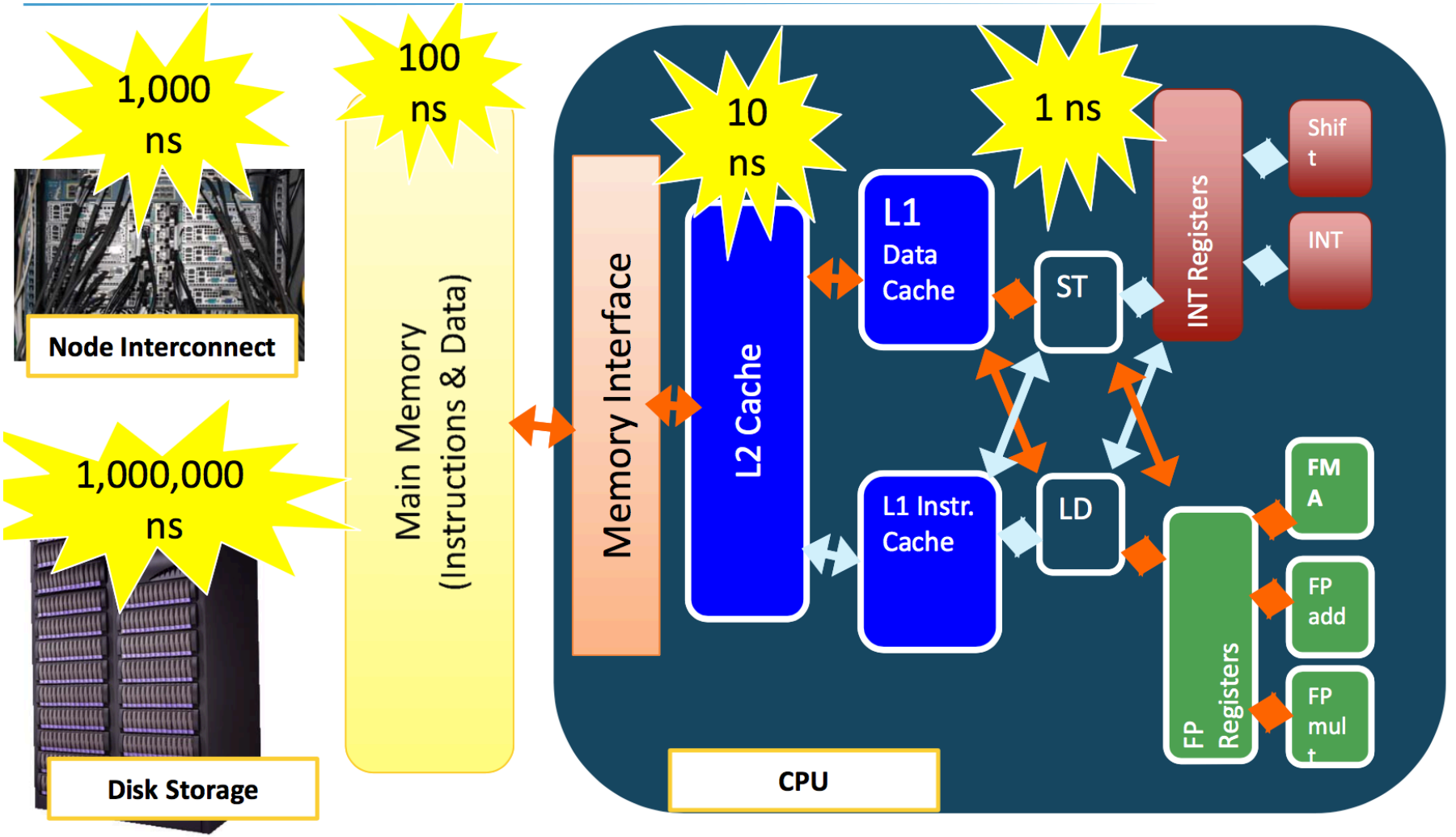
Eine Festplatte macht noch kein Storage-System

- <milmmädchenrechnung>
  - Um 10 Pbyte zu speichern braucht man aktuell 1000 x 10 Tbyte Platten
  - NL-SAS Platte hat MTBF von 1.2 Mio Stunden ... 1.2 Mio h / 1000 / 24h = Ein Fehler alle 50 Tage
  - Oder: NL-SAS Platte hat Bit Error Rate von 1/10^15 bits ~= 110 TByte. 10 Pbyte einmal vollschreiben: ~100 Bit Fehler.
- Man braucht also
  - Fehlerkorrekturen und Redundanzen
  - Etwas was 1000 Einzelschicksale in wenige, grössere Blöcke gruppiert
- (In Wirklichkeit deutlich mehr „Einzelschicksale“)

RAID und Erasure Coding

- RAID: Redundant Array of In[expensive|dependent] Disks
  - Heute Typischerweise RAID-6: n+2 Parity
  - Realisiert typischerweise über Hardware-RAID Controller
- Erasure Coding
  - RAID-6 eine spezielle Form von Erasure Coding (ggf in Hardware)
  - Aktuell viel Entwicklung um Erasure Coding in die Filesysteme zu bekommen – ohne HW-Controller!
  - Einige kommerzielle Produkte setzten dies um, OpenSource Varianten noch in den Kinderschuhen

Latenzen: Zeit zum Lesen des ersten Byte



Richard Gerber

Bandbreiten

- Wie schnell können Daten gestreamt werden von/zu Platte?
- N*10 bis N*100 GByte/s sind heute typisch für grosse Systeme (N*10 PB)
  - Aggregiert! Also „viele WNs reden mit vielen Servern“
  - Single-Stream ist begrenzt von Netzwerk-interface und ggf Speichermedium
- Hängt allerdings von der Applikation ab:
  - Maximale Bandbreite ist nur abrufbar wenn grosse Blöcke gelesen werden, und wenn dies im Streaming-Modus passiert!

Latenzen und Bandbreiten-Optimierung

- Schreiben: Buffering
  - Schneller Speicher sammelt Daten und schreibt sie dann (ggf. streaming modus) auf langsameren Speicher
- Lesen: Caching
  - Ein ganzer Block wird vom langsamen in den schnellen Speicher gelesen, auch wenn nur ein Teil des Blocks angefragt wurde. Der Block bleibt erstmal im Cache
- Unterschiedliche Orte: In der Nähe der CPU, im Worker-Node, dedizierter IO-Node, spezialisierte Storage-Server, zusätzliche Platte in Storage Server, im RAID-Controller oder auf der Platte selber

Local / Global aus Sicht des Worker-Node

- Local Storage: Die Festplatte (on-board)
  - Altbekannt: Für OS und /tmp. Uninteressant
  - Aber: Eventuell spezieller IO-Node mit schnellem Cache (SSD). „Local Read-Only Cache“ (LROC im IBM-GPFS Jargon)
- Cluster-Lokaler Storage:
  - Network Attached Cluster File System
  - Nur im dedizierten Cluster Interconnect (schnell) verfügbar, typischerweise nur für ein HPC System
- Globaler Storage:
  - Im Campus Netzwerk (oder sogar weltweit) verfügbar.
  - ZB Ausgangsort für Rohdaten oder permanenter Speicherort

Verschiedene Arten von Clustered Storage

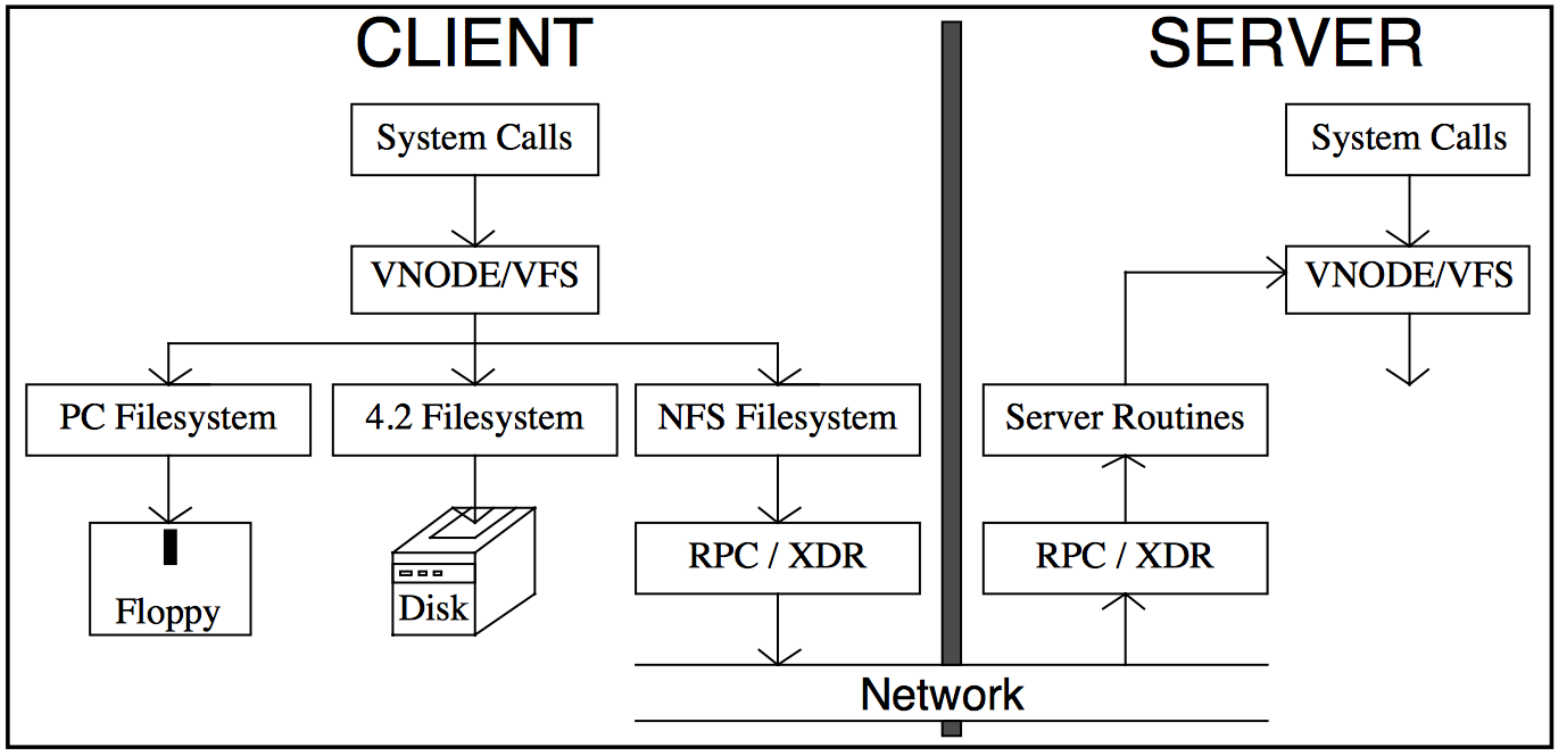
- Network attached storage
- Shared-disk FS
- Verteiltes FS

- https://en.wikipedia.org/wiki/Clustered_file_system

Network attached storage

- Typischerweise NFS (Network File System) Protokoll
  - CIFS/SMB oder Andere sind im UNIX-lastigen HPC Umfeld eigentlich nicht zu finden
- NFS ist in Linux quasi überall zu finden
  - Kernel-NFS Server, NFS Client Modul...
  - Mounten ist trivial: mount –t nfs server:/export/path /local/path
- Viele Hersteller bieten NFS Appliances an

NFS Client <-> Server Kommunikation

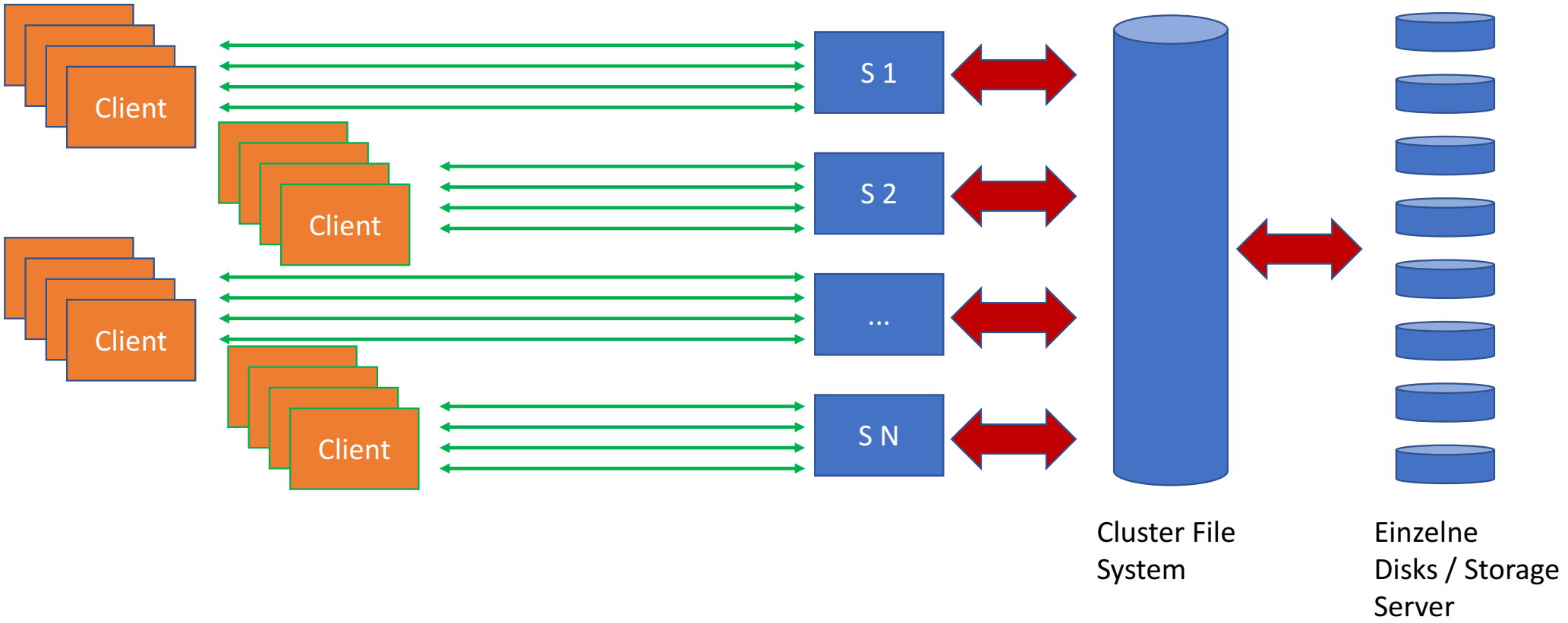


http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.25.7841 (Russel Sandberg) das Paper stammt von 1986 ☺

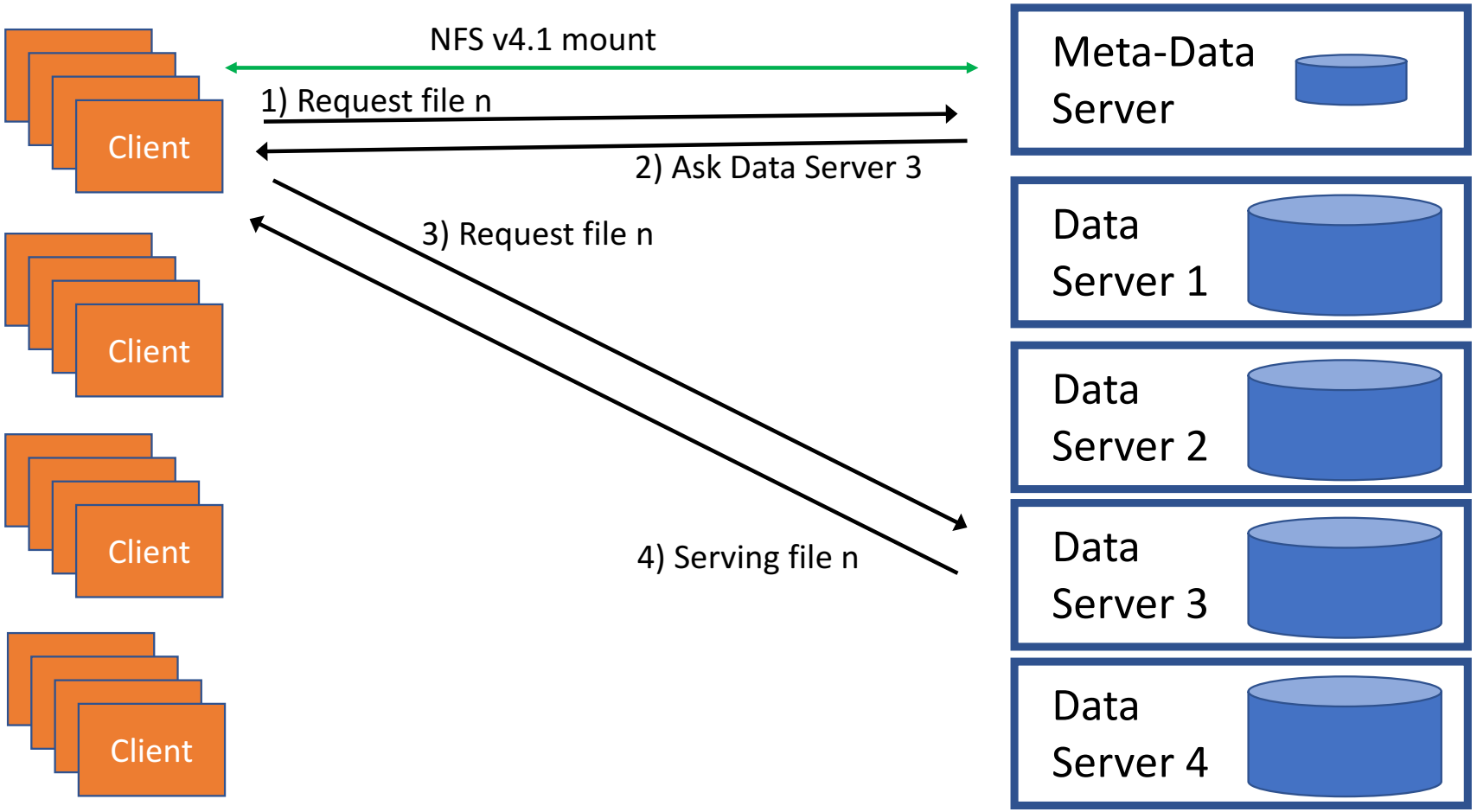
Skalierung? NFS v3 und NFS v4.0

- Die Kommunikation geht immer zwischen Client und Server
- Mehr Plattenplatz im Server
  - Bottleneck: Netzwerkanbinung Server
- Mehrere Server
  - Bandbreiten-Skalierung über #Server
  - Unterschiedliche Namespaces
    - /mnt/nfs/server1
    - …
    - /mnt/nfs/servern

Skalierung in NFS v3 und NFS v4.0, Beispiel



Skalierung mittels NFS v4.1 / pNFS



NFS v4.1/pNFS vergleichsweise neu

- Einige grosse Storage-Hersteller unterstützen NFS v4.1/pNFS
- Server haben noch nicht den Reifegrad von grossen NFS v3 und NFS v4.0 Appliances
- Client Code noch nicht so ausgereift wie v3 und v4.0
- Hersteller und Nutzer sind zögerlich

- DESY war mit NFS v4.1/pNFS Server in dCache Vorreiter
- DESY hat ca 1000 Client im Einsatz die NFS v4.1/pNFS mounten
  - Allerdings nicht im HPC Umfeld☺

NFS Security @ HPC

Security?

NFS Security

- OK, es gibt Kerberised NFS (mit NFS v4)
  - Macht nur keiner im RZ-Bereich (ich kenne keinen...)
- Wer darf mounten?
  - Server hat eine Liste von IP Adressen / Host-Namen von Clienten die mounten dürfen
  - HPC Cluster: IP Adressen und Host Namen sind sicher™, und können nicht gefaked werden
- Wer darf auf Daten eines Mounts zugreifen?
  - UID/GID basierende Security. Üblicherweise simple UNIX-ACLs (User/Group/Others)
  - HPC Cluster: UID/GID sind sicher™, und können nicht gefaked/missbraucht werden

„Echte“ Cluster File Systeme

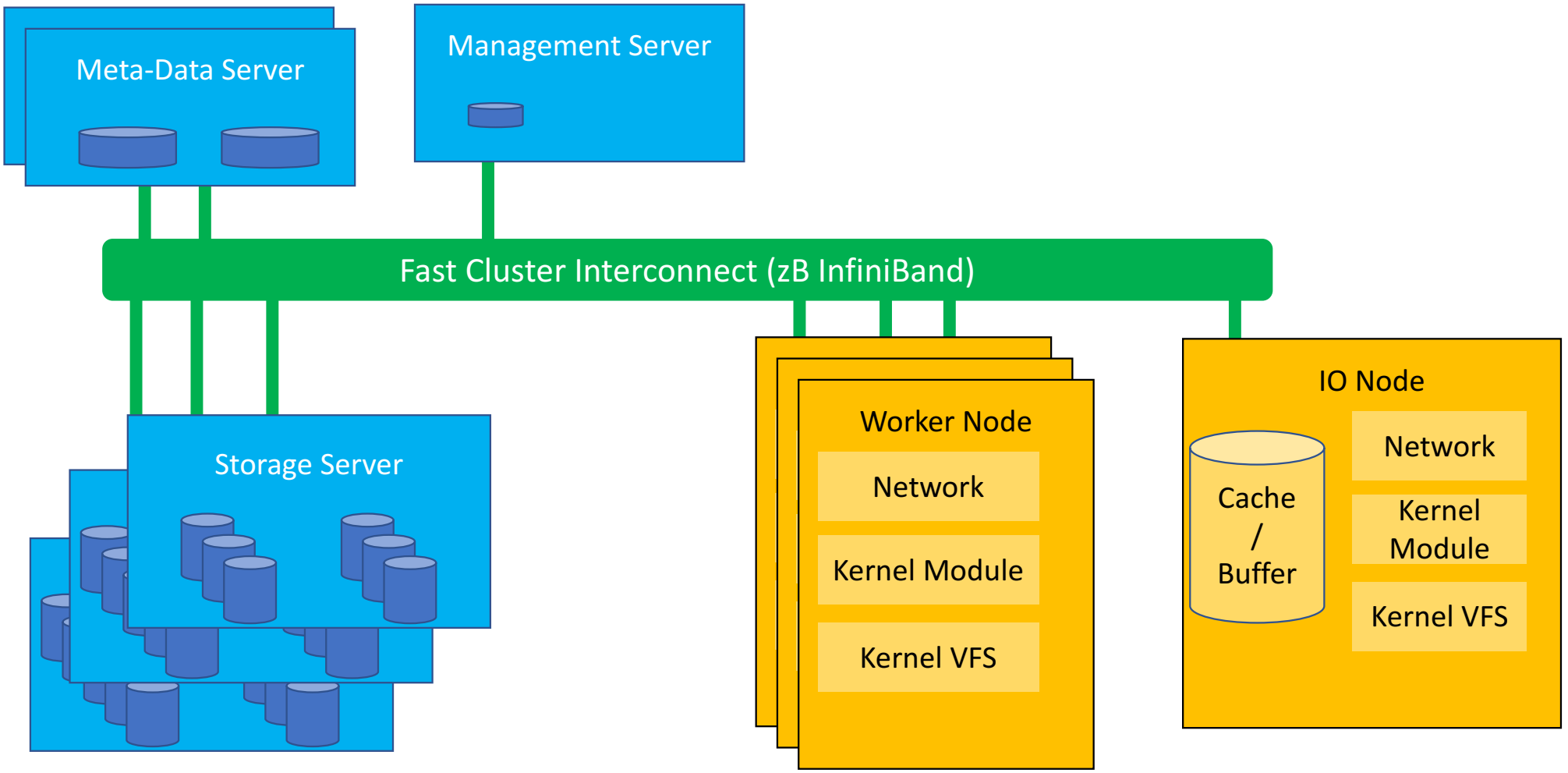
- Am DESY im Einsatz
- Lustre: HPC Storage am Standort DESY/Zeuthen
- BeeGFS: Günstiger Projekt-Space im HPC Cluster DESY/HH
- GPFS (Jetzt Spectrum Scale): Online-Datennahme der neueren DESY Experimente, schnelle Analyse

lustre®
File System

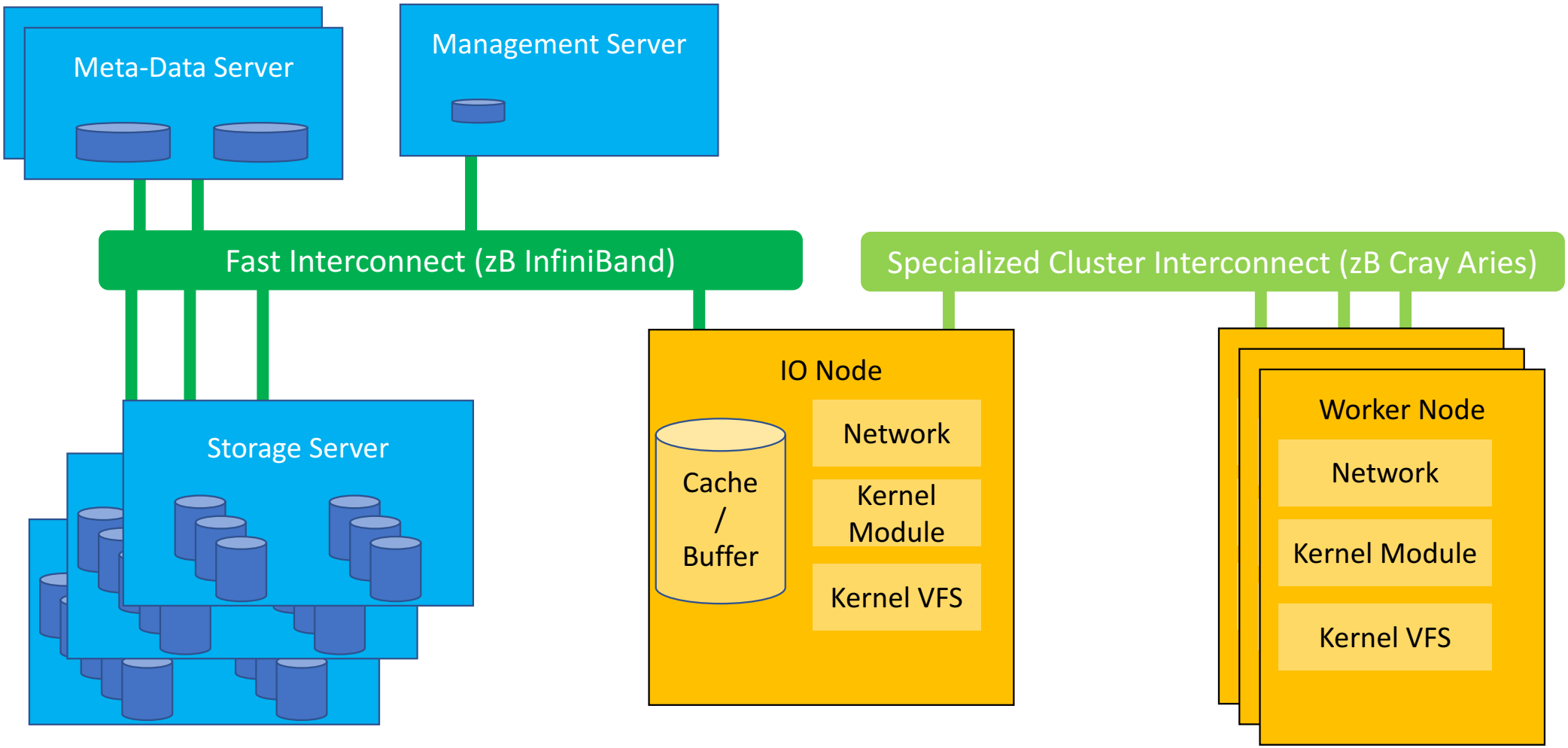
BeeGFS®

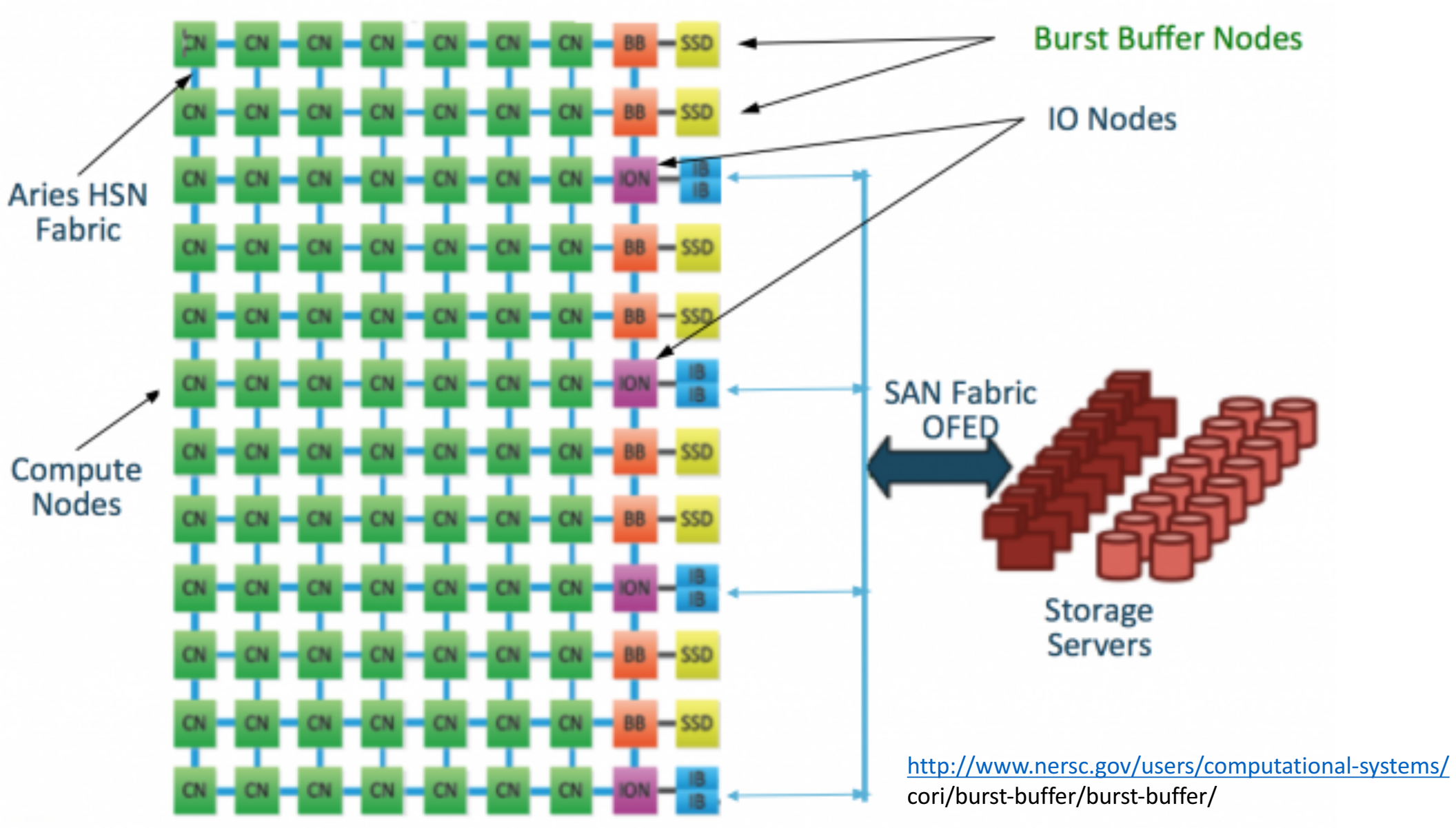
IBM®
GPFS

Generisches Schema von Cluster-File-System



Oder auch:


Meta-Data Server
Management Server
Fast Interconnect (zB InfiniBand)
Specialized Cluster Interconnect (zB Cray Aries)
Storage Server
IO Node
Cache / Buffer
Network
Kernel Module
Kernel VFS
Worker Node
Network
Kernel Module
Kernel VFS



http://www.nersc.gov/users/computational-systems/cori/burst-buffer/burst-buffer/

Burst Buffer

- Netzwerktopologisch sehr nahe an den Compute-Nodes
- Dadurch hohe Bandbreite, zB CORI @NERSC
  - „approximately 1.7 TB/second of peak I/O performance with 28M IOPs, and about 1.8PB of storage“
  - http://www.nersc.gov/users/computational-systems/cori/burst-buffer/burst-buffer/
- Wichtig zB für Snapshots / Checkpointing
  - Das (koordinierte) Schreiben der Snapshots auf die Burst Buffer geht sehr schnell
  - Wenn die Berechnung wieder angelaufen ist werden die Daten vom Burst Buffer auf den eigentlichen Storage geschrieben. Dies geschieht deutlich langsamer

Einige Produkte im Detail: BeeGFS

- High performance parallel filesystem developed 2007 from Competence Center for High-Performance Computing, Fraunhofer ITWM
  - Aim: Replace GPFS and Lustre by something easy to deploy, config and administer
  - Originally named FhGFS it was renamed in 2014 to allow a commercial spin off
- Development is driven by Fraunhofer, the company ThinkparQ offers support.
- The Software is free of charge
- License and costs
  - The client kernel module is under the GPL
  - Storage and Management Daemons, currently closed source, but guaranteed in the context of the DEEP-ER project to become open-source
- Commercial support offered: Annual license per storage target

BeeGFS Facts

- Distributed Object and Metadata
  - Aggregated throughput for objectdata (using striping)
  - Loadbalancing for Metadata **>** Linux (only) based
- Packed for for RHEL, Debian, Suse
  - Support x86_64 and XeonPHI (proof of concept for ARM)
- Server runs in userspace, and use supported filesystem of the OS
  - Object Store tested with xfs, ext4 and zfs
  - Metadata Stored on ext4 filessystem (use extended Attributes)
- Clients are kernel modules
  - Support all kernels from 2.6.16 to latest vanilla, no Kernel Patch
  - Automatic rebuild after kernel update
- Support native Infiniband/1GE/10GE/40GE
- BeeGFS is improving fast
  - Already there: Raid Level, HSM integration
  - Upcoming: HA, Data integrity, erasure coding

https://agenda.infn.it/getFile.py/access?sessionId=7&resId=0&materialId=0&confId=5516

GPFS Basics

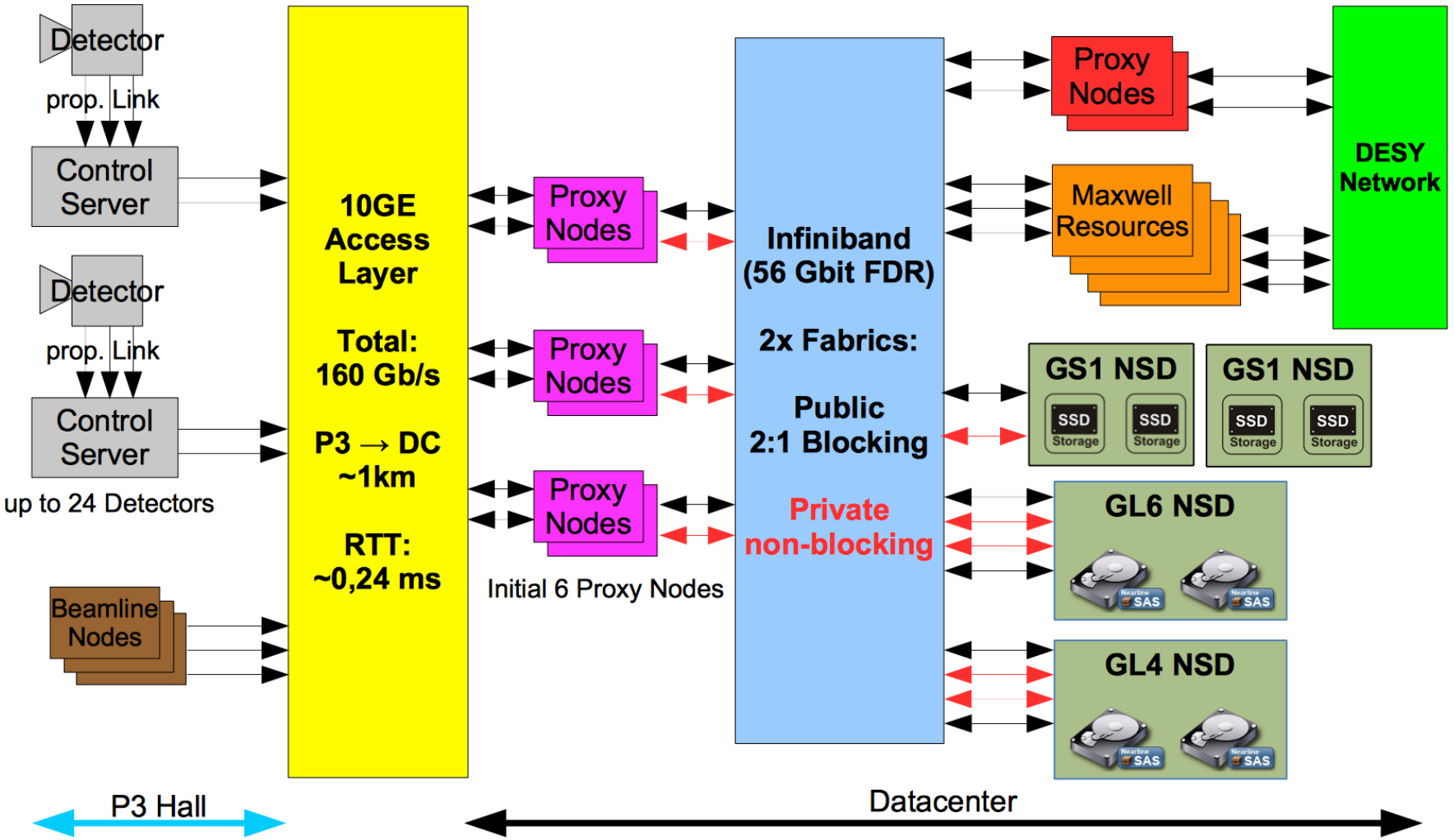
- General Parallel File System
  - mature IBM product generally available for more then 10 years (GPFS has been
- available on AIX since 1998 and Linux since 2001)
- ▪ Works on AIX, Linux and (surprise!) Windows
- ▪ Adaptable to many user environments by supporting a wide range of basic configurations and disk technologies
- ▪ Provides safe, high BW access using the POSIX I/O API
- ▪ Basic features: POSIX API, journaling
- ▪ Provides non-POSIX advanced features
  - ▪ e.g., DMAPI, data-shipping, multiple access hints (also used by MPI-IO)
  - ▪ ILM, integrated with tape, disaster recovery, SNMP, snapshots, robust NFS support
- ▪ Provides good performance for large volume, I/O intensive jobs
- ▪ Works best for large record, sequential access patterns, has optimizations
- for other patterns (e.g., strided, backward)
- ▪ Converting to GPFS does not require application code changes provided the code works in a POSIX compatible environment

https://agenda.infn.it/getFile.py/access?sessionId=7&resId=0&materialId=0&confId=5516

Was GPFS nicht ist:

- GPFS is not a client/server file system like NFS, CIFS (Samba) or AFS/ DFS with a single file server.
  - GPFS nodes can be an NFS or CIFS server, but GPFS treats them like any other application.
- GPFS is not a SAN file system with dedicated metadata server.
  - GPFS can run in a SAN file system like mode, but it does not have a dedicated metadata server.
- GPFS avoids the bottlenecks introduced by centralized file and/ or metadata servers.

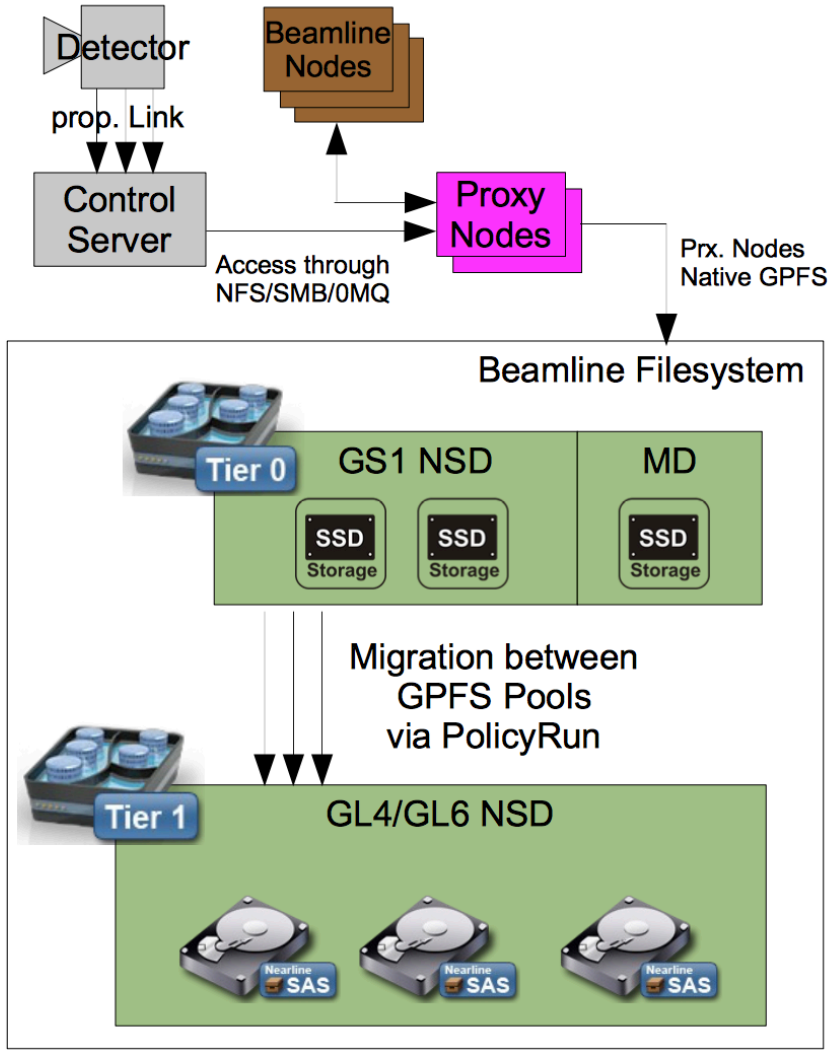
GPFS am DESY: Datennahme



Stefan Dietrich

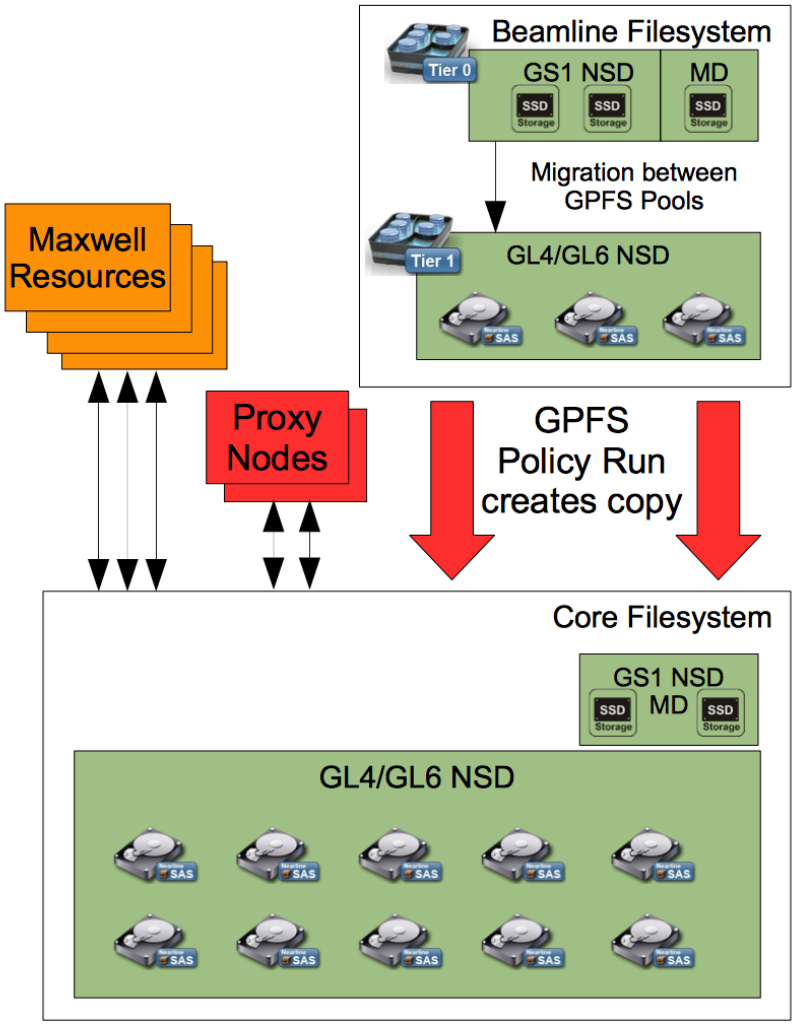
Beamline Filesystem

- "Wild-West" area for beamline
- Only host based authentication, no ACLs
- Access through NFSv3, SMB or ZeroMQ
- Optimized for performance
  - 1 MiB filesystem blocksize
  - Pre-optimized NFSv3: ~60 MB/s
  - NFSv3: ~600 MB/s
  - SMB: ~300-600 MB/s
- Tiered Storage
  - Tier 0: SSD burst buffer (< 10 TB)
  - Migration after short period of time
  - Tier 1: ~90 TB capacity



Core Filesystem

- "Clean world"
- Full user authentication
- NFSv4 ACLs
- Access through NFSv3, SMB or native GPFS
- GPFS Policy Runs copy data
  - Beamline → Core Filesystem
  - Single UID/GID
  - ACL inheritance gets active
  - TBD: raw data set to immutable
- 8 MiB filesystem blocksize
- 2 snapshots per day
- Fileset per beamtime



Zusammenfassung

- IO wichtige Komponente für das Funktionieren eines HPC Clusters
- IO & Storage wird häufig vernachlässigt ... In der Ausbildung, Nutzer-Planung (und manchmal auch Cluster-Planung)
- Storage ein weites Feld
- Heute haben Sie alles aus Admin-Sicht gelernt
- Nächstes Mal lernen Sie alles aus Nutzer-Sicht
- Und üben das fleissig

"A supercomputer is a device for turning compute-bound problems into I/O-bound problems." Ken Batcher

… One more thing: Prüfung

- Prüfung am 27.7. ab 14:00
- Mündliche Prüfung bei Herrn Prof. Gülzow, Beisitzer Y. Kemp
- Dauer ca 20 Minuten pro Prüfling
- Anmeldung und konkrete Termine werden rechtzeitig bekannt gegeben.
- Inhalt: Kurs und Übungen, Materialien siehe Folien