Data Access on a Data Lake Straw Model

(discussion of the document)

Frank, Markus, Ilija, Stéphane and Xavier

WLCG DOMA General Meeting - 23 Jan 2019

Disclaimer

The mandate of the DOMA ACCESS working group is data access, caching and latency hiding for all kind of workloads

The current discussion pertains only to “analysis use case”, or more general, a use case that implies substantial reuse of data by many users

The “production data workflows” (official raw/derived data) managed by higher level infrastructure (Data Lakes) will be discussed at some other time

As the “analysis data” will be extracted from the “production data layer” we propose a strawman model of a Data Lake based infrastructure to explore how data usage and access can be optimized for analysis

Motivation (1/2)

HL-LHC will be a multi-Exabyte challenge where the envisaged Storage and Compute needs will be a factor 10 above what can be met by the evolution of current technology within a flat budget

The WLCG community needs to evolve current computing models in order to introduce changes in the way we use and manage storage. Focusing on resource optimization to improve performance and efficiency and simplifying operations

Motivation (2/2)

Some points in favor of the Data Lake model as an evolution of the current infrastructure:

- Storage costs reductions: global redundancy instead of local redundancy, dynamic adaptation of QoS, economy of scale
- Conceptual separation of compute and storage services allows higher specialisation, leading to improvements in reliability and operations
- Computational oriented sites can provide increased resources with less effort
    - The effort for storage operations will be drastically reduced and sites can re-orient manpower and budget to provide *deeper* services to their users: support for enduser analysis, training in advanced analysis techniques, support for machine learning infrastructure, etc.

Strawman Model: Global Description (1/2)

- The world is divided in **Data Lakes**
  - Areas with a radius of ie. 20-30 ms in network latency to keep reasonably efficient direct access between sites
- A group of **Data and Compute Centers** (DCC) define a Data Lake
- A Data Lake must have one or more **Archive Centers** (AC) associated
- A Data Lake may have 3 or more DCCs providing large scale disk storage
- A given center may be providing **both:** DCC and AC functionality
- Each DCC can store Data Lake files **without** need for local redundancy
  - It is up to the site hosting the DCC to select the best QoS for their needs

Strawman Model: Global Description (2/2)

- A **Data Lake** hosts a distributed working set of analysis data
    - This comprises an experiment’s full set of mini/nano-AODs (or equivalents) of a given campaign/period
    - Based on CMS’ forecasts O(50) and O(1) PB are required respectively for mini/nano AODs
- If **not** affordable, the analysis working set is distributed across more than one lake
    - The experiment’s workload management system will allocate jobs within the matching Data Lake
    - Popular datasets may be hosted in more than one Data Lake
    - To allow for non-local redundancy at least 2 copies have to be available globally
- **Caches** are used to reduce the impact of latency and reduce network load

Storage Building Blocks (1/2): Naming Conventions

Data Lake: set of sites, associated by proximity, providing online and nearline storage services to an identified set of user communities. Capable to carry out independently well defined tasks

Replica: managed space where one copy of the working set is guaranteed to be on disk

Staging Area: located at the edge of DCCs, data is moved and removed there by data management systems. This is storage that is used to give unimpeded access to workloads that are active or will become active very shortly ie. full reprocessing campaigns or gateways to HPC/IaaS

Storage Building Blocks (2/2):Naming Conventions

cache+ (proxy-cache with failover functionality): a self-managed transient storage layer at the edge of a site, having **streaming** ability and providing **read-ahead** functionality

- The content of **cache+** is only known by **cache+** while for users and data management services (ie. Rucio) the cache is fully transparent
- The role of **cache+** is to:
  - **Reduce wide area network bandwidth** by holding frequently used files in the cache
  - Ability to **read ahead to reduce the impact of latency** and peak bandwidth requirements for the first reading of the file

Client buffer: the ability of the application to manage the I/O in order to achieve an intelligent cache behavior (streaming, read-ahead) in terms of managing latency and bandwidth.

AC: Archive Center

An **Archive Center** is defined as tape or tape-equivalent-QoS enabled center able to provide long term archive system

A proportional **Staging area** is required to handle the data input/output from the Data Lake to the archive infrastructure

Main functionalities:

- Permanent data archive (raw data, custodial, long-term)
- Data recall ability for reprocessing campaigns

DCC: Data and Compute Centers

DCCs are sites providing large disk storage and computing resources

DCCs may be distributed. Shared disk space across sites but shown as one DCC indistinguishable for users and data management systems

A DCC may have multiple Compute Centers assigned to it.

- With the current workloads it has been shown that moderate latencies (up to 10ms) can be handled with minimal impact on the CPU efficiency (latency hiding by the application or Xcache)

Network bandwidth at the DCC must be proportional to the processing power across all assigned Compute Centers plus the processing power of the DCC itself

CC: Compute Centers (1/2)

Compute Centers are sites providing computing resources

In a CCC (**Compute Center with Cache+**) the data is accessed through a latency hiding cache where <u>all</u> data flow through this cache (proxy behavior)

In a CCDA (**Computing Center with Direct Access**) data is accessed directly from the DCC relying on the client side latency hiding capabilities

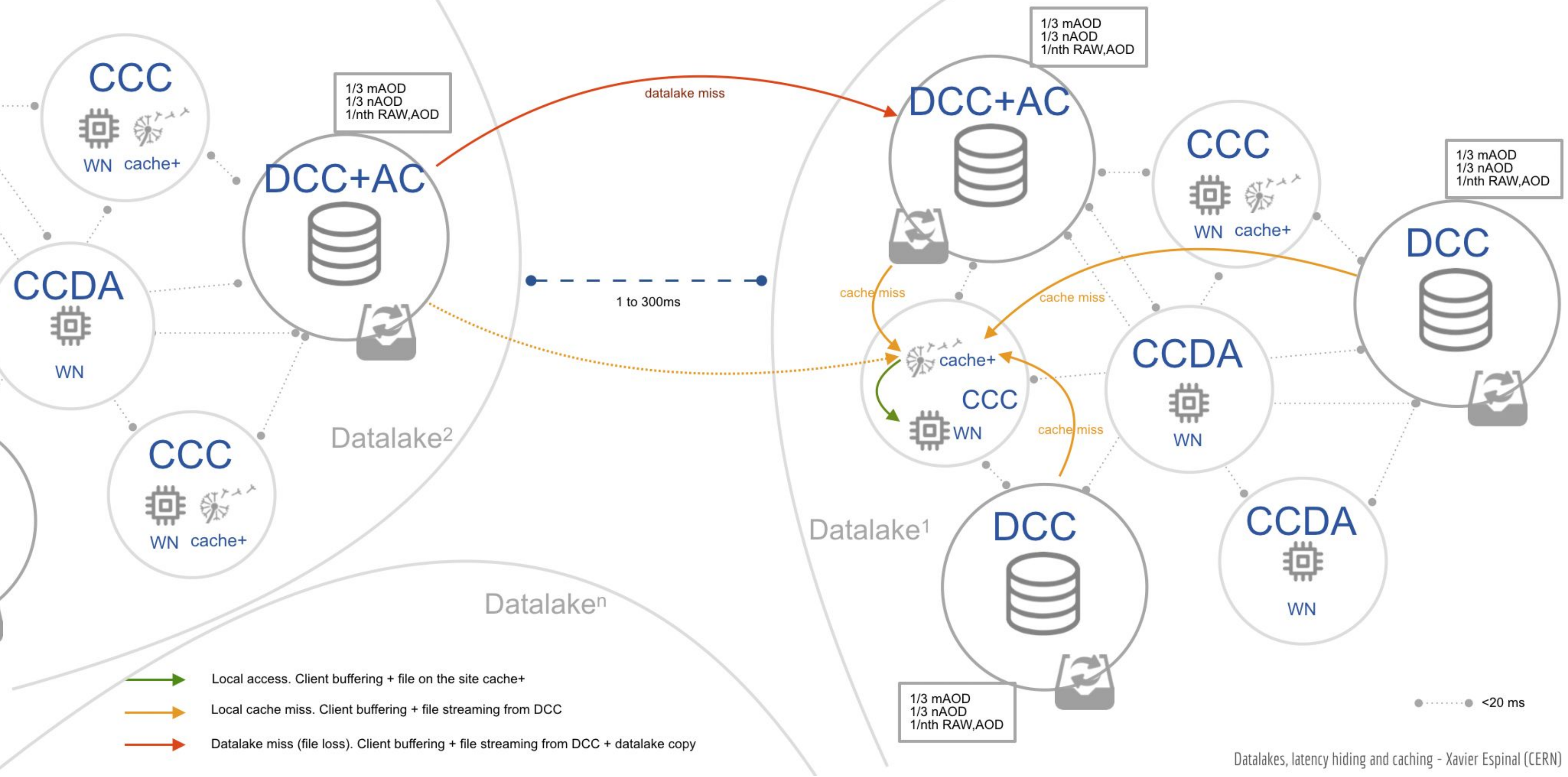
CC: Compute Centers (2/2)

cache+ characteristics at the CCCs:

- ***cache+*** can be located close to the WN (need for dedicated nodes) or can be distributed across the worker nodes of the Compute Center (no need for dedicated nodes)
- ***cache***+ has a some additional capabilities to the current Xcache:
    - It can map files via an experiment’s local/trivial file catalogue
    - It can map via contacting the experiment’s replica catalogue (Rucio etc.)
    - It has a failover mechanism to retrieve files that are missing in its data lake.
        - This mechanism is aware of the surrounding Data Lakes and knows the distance in units of latency and bandwidth (this information can be provided statically by configuration)
- ***cache+*** has no external state and acts as a proxy
- If not available the jobs continue to get data directly and latency hiding is then up to the client

File access orchestration: WN to Cache to Staging Area to Datalake

- Disk failures estimation: **only 1%** of data will be fetched outside the local datalake
- **Efficiently hide** latencies with **buffering** at the client side and **access through cache+**



🟩 Local access. Client buffering + file on the site cache+

🟧 Local cache miss. Client buffering + file streaming from DCC

🟥 Datalake miss (file loss). Client buffering + file streaming from DCC + datalake copy

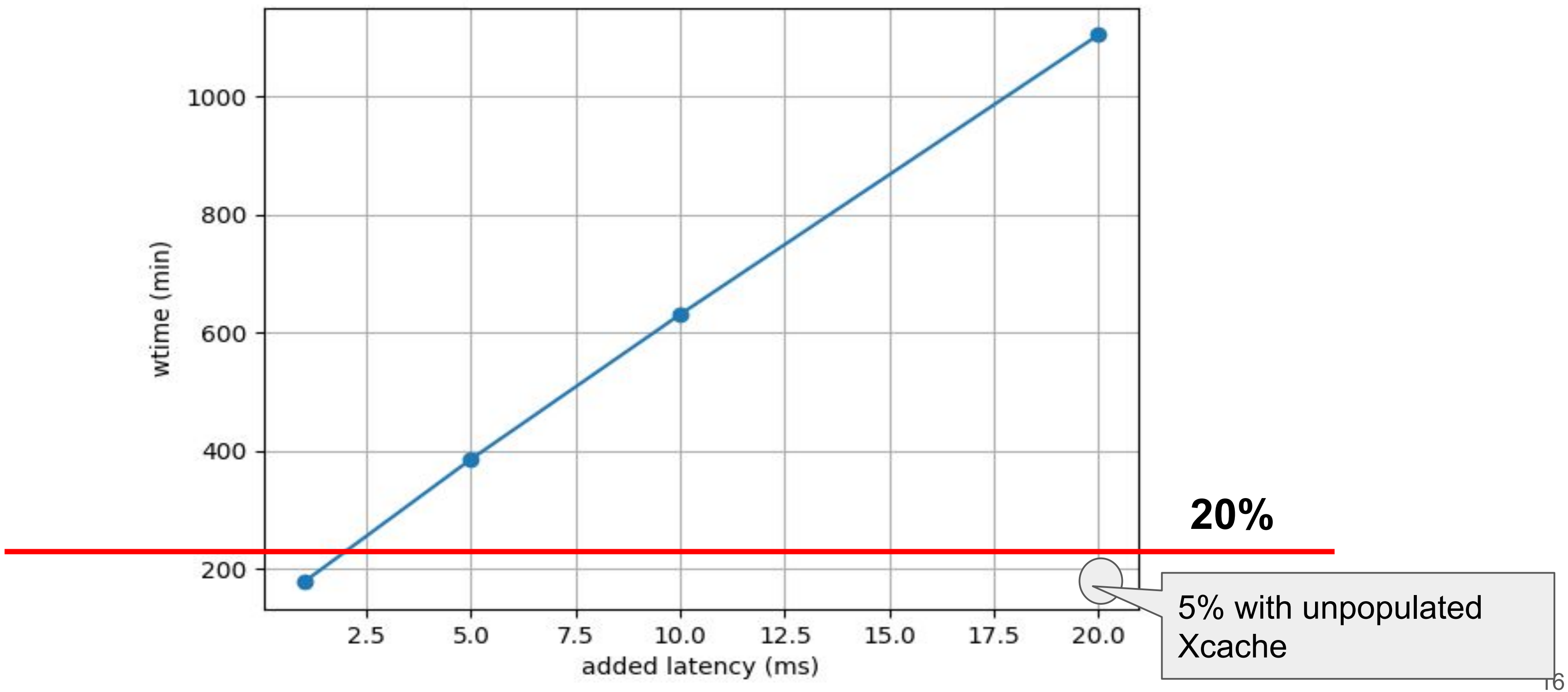
	AC Archive Centers	DCC Data and Compute Centers	CC Compute Centers	AF Analysis Facility	EF Ephemeral facilities (HPC, IaaS)
Archive	☆☆☆	☆			
Disk Storage	☆	☆☆☆	☆	☆	
Staging Area	☆☆☆	☆☆	☆	☆	
Cache			☆☆☆	☆	☆☆
CPU	☆	☆☆☆	☆☆☆	☆☆	☆☆

☆ Optional ☆☆ Available ☆☆☆ Large

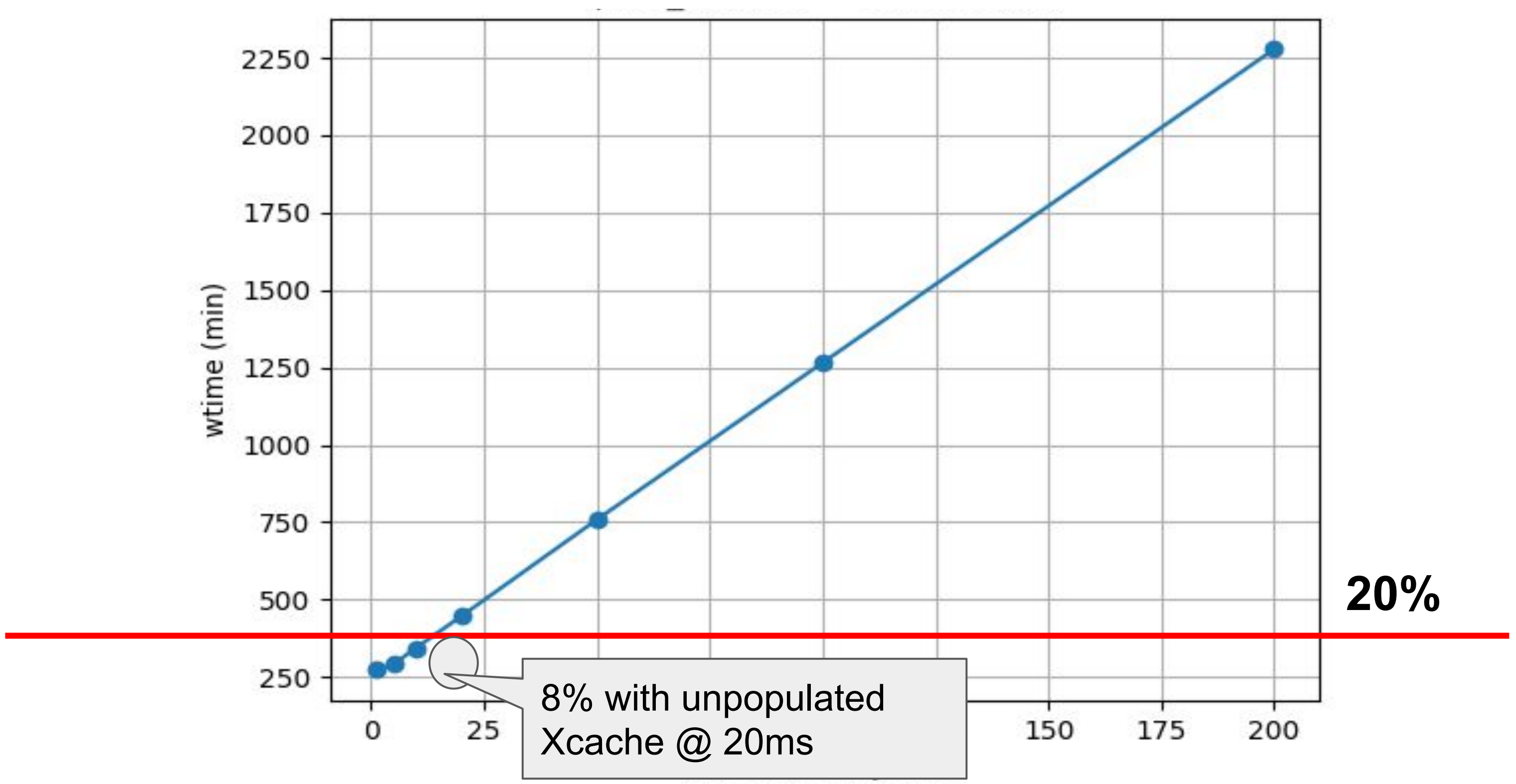
Some examples of dependency on latency

- Different LHC workloads
- With/without client side latency hiding
- Some measurements with Xcache

I/O intensive workload, no client side latency hiding

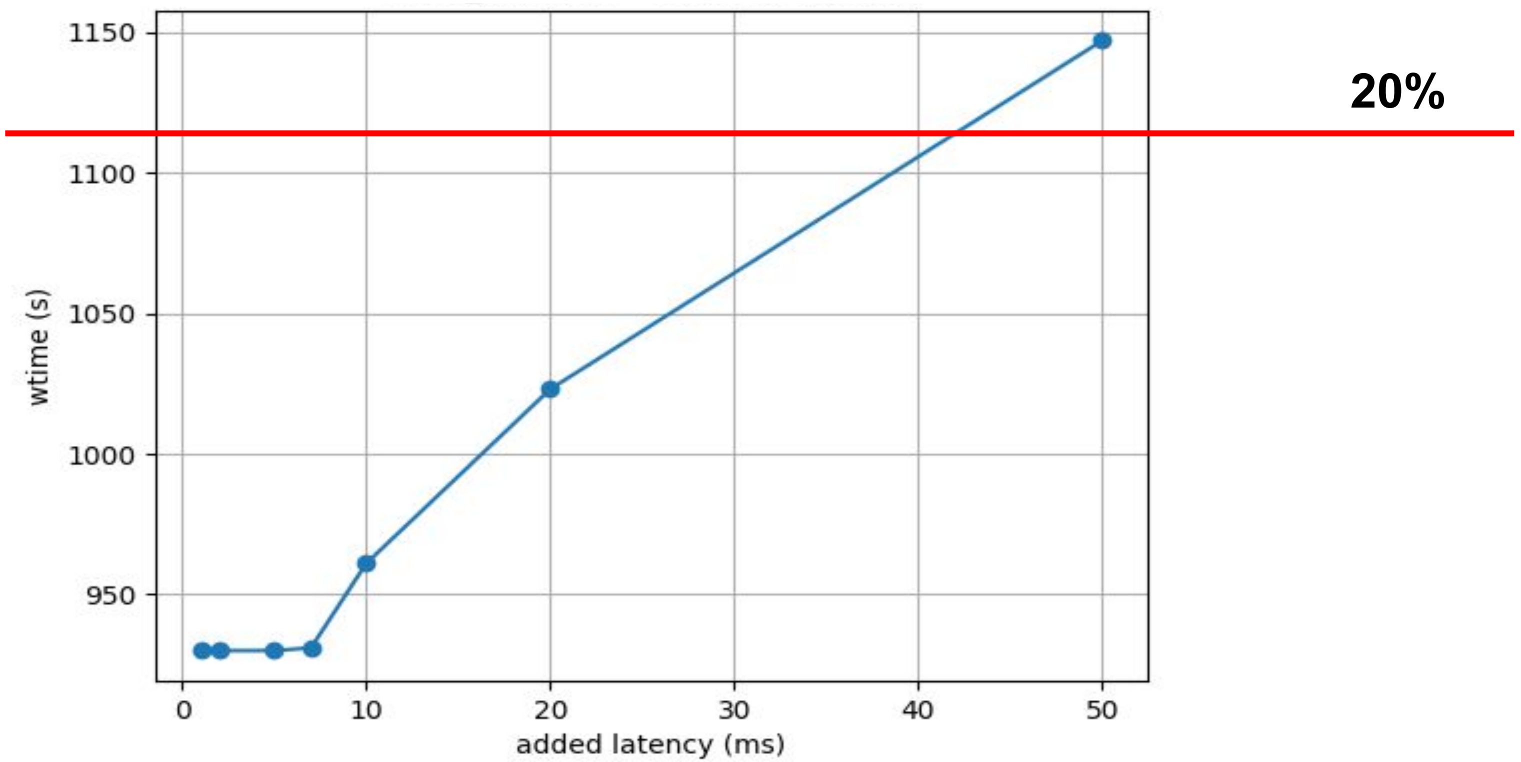


Workload with moderate I/O, no client side hiding



8% with unpopulated Xcache @ 20ms

Moderate I/O, Client side latency hiding



Disk failures, data loss and data recovery (1/5)

The collection of all files that a Data Lake holds is specified to the Data Lake by the Replica Manager (RM, ie. Rucio) and changes dynamically under its control

The Data Lake is responsible to have all the assigned data on disk

When a file is lost at the Data Lake there must be a recovery mechanism defined, e.g. the lake may ask the RM for the missing file

The RM may control placement of all files across all DCCs in the Data Lake, and be informed by the DCC when a file is lost. The RM thus will initiate recovery

Disk failures, data loss and data recovery (2/5)

After disk failure detection need to create the list of affected files

Considering a 100PB system, 1PB/year will be *impacted* on average (assuming 1%/year of disk failure rate*)

- On a daily basis this translates to 2.75 TB (~2000 files/namespace queries in addition to the normal operations)

The list has to run through the mapping: fileName → Data Set → Data Lakes → Site

This list has to be given to a RM system to copy the lost data to the local storage system from the remote site

** CERN measured in 2017/18 a failure rate of 0.86 %/year for the disks in EOS. A rate of 1% /a can be used as a conservative approximation.*

Disk failures, data loss and data recovery (3/5)

The number of sites that could potentially contribute to the data recovery process is high

10TB failed disk at one DCC will be re-constructed in few hours with no special bandwidth requirements:

- Assuming 10 Data lakes and 3 DCCs per Data Lake
- Assuming ⅓ of the sites will hold part of the data that was present at the broken disk
- Assuming 0.1GB/s of recovery throughput per DCC,
- => Data is recovered in ~2.5h. This recovery will occur every 3.6 days (on average)

More detailed numbers and calculations on this: [diskfailures] [serverfailures]

Disk failures, data loss and data recovery (4/5)

Reasons for missing/unavailable data in a system without internal replication

- Disk failures, statistically affecting individual disks (1%/year)
- Disk servers being offline due to maintenance (upgrades, reboots, etc.)
  - Depending on operations practice
  - Since a disk server affects many disk this can be significant

Disk failures, data loss and data recovery (5/5)

Running workloads will be impacted during the restoration of the disk as clients might try to access some some of the affected files

This is proportional to the restoration time and the fraction of files that are accessed on the failed disk/hour

Following with our example with the 100 PB and a failed disk of 10 TB the miss rate during reconstruction of the disk is 0.53 ‰

- This corresponds to an average file access miss rate of 0.036‰, which means 1 access in 27780 will request a file currently not present, in worst case the file will be back in few hours

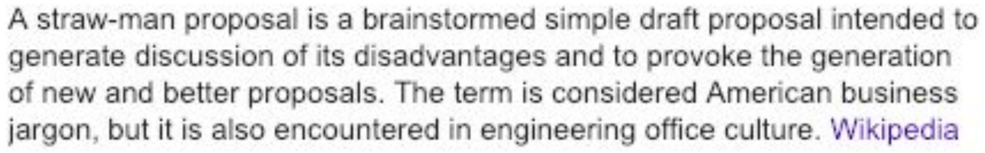
This has to be compared to the normal job failure rates of the experiments. From Panda log file analysis we know that for production jobs is O(10%)

Components for data recovery and failover: what is missing?

- Storage Systems
  - All systems can produce on demand lists of files that have been on a broken disk
  - This process has to be automated (useful anyway, but not trivial)
  - Lists have to be split by experiment and brought into a standard format (not tough)
  - List have to be sent to the experiments' data management systems (not tough)
- Data Management Services
  - Standard format file lists have to be received (not tough)
  - Converted to internal format for data replication (not tough)
  - These sets of files have to be replicated by the standard process
    - Taking into account the specific loss (might need some thinking)
  - Potentially handling replica location requests (authorised) from cache+ systems (not tough)
- Extensions to Xcache
  - Resolving for files not present in the expected location alternative locations
    - Either by mapping, or by query to the experiment's data management system
    - Failing over

Conclusions

A straw man model has been presented. This enables to start discussions with something to refer to. Get involved.


A straw-man proposal is a brainstormed simple draft proposal intended to generate discussion of its disadvantages and to provoke the generation of new and better proposals. The term is considered American business jargon, but it is also encountered in engineering office culture. Wikipedia

We are trying to address several items (probably too many but the overlaps are huge)

- Optimize data storage and data access for analysis workflows
- Optimize storage resources focusing on performance, efficiency and operations simplification
- A starting point for an evolution of the current site topology towards a global Data Lake infrastructure