Dynamic Federations

Global Access to HTTP/WebDAV Storage Elements

Fabrizio Furano (CERN), Patrick Fuhrmann (DESY), Adrien Devresse (CERN), Alexandre Beche (CERN), Alejandro Alvarez (CERN), Martin Hellmich (CERN), Oliver Keeble (CERN), Ricardo Brito Da Rocha (CERN)

Currently data lives on islands of storage

Catalogues are the maps, FTS/gridFTP are the delivery companies, experiment frameworks populate the island

Jobs are directed to places where the needed data is (or should be) ......

Almost all data lives on more than one island

Common assumptions:

perfect storage (unlikely to impossible) perfect experiment workflow and catalogues (unlikely)

Strict data locality has limitations

e.g. a single missing file can derail the whole job or series of jobs

Make different storage clusters be seen as one

Make global file-based data access seamless

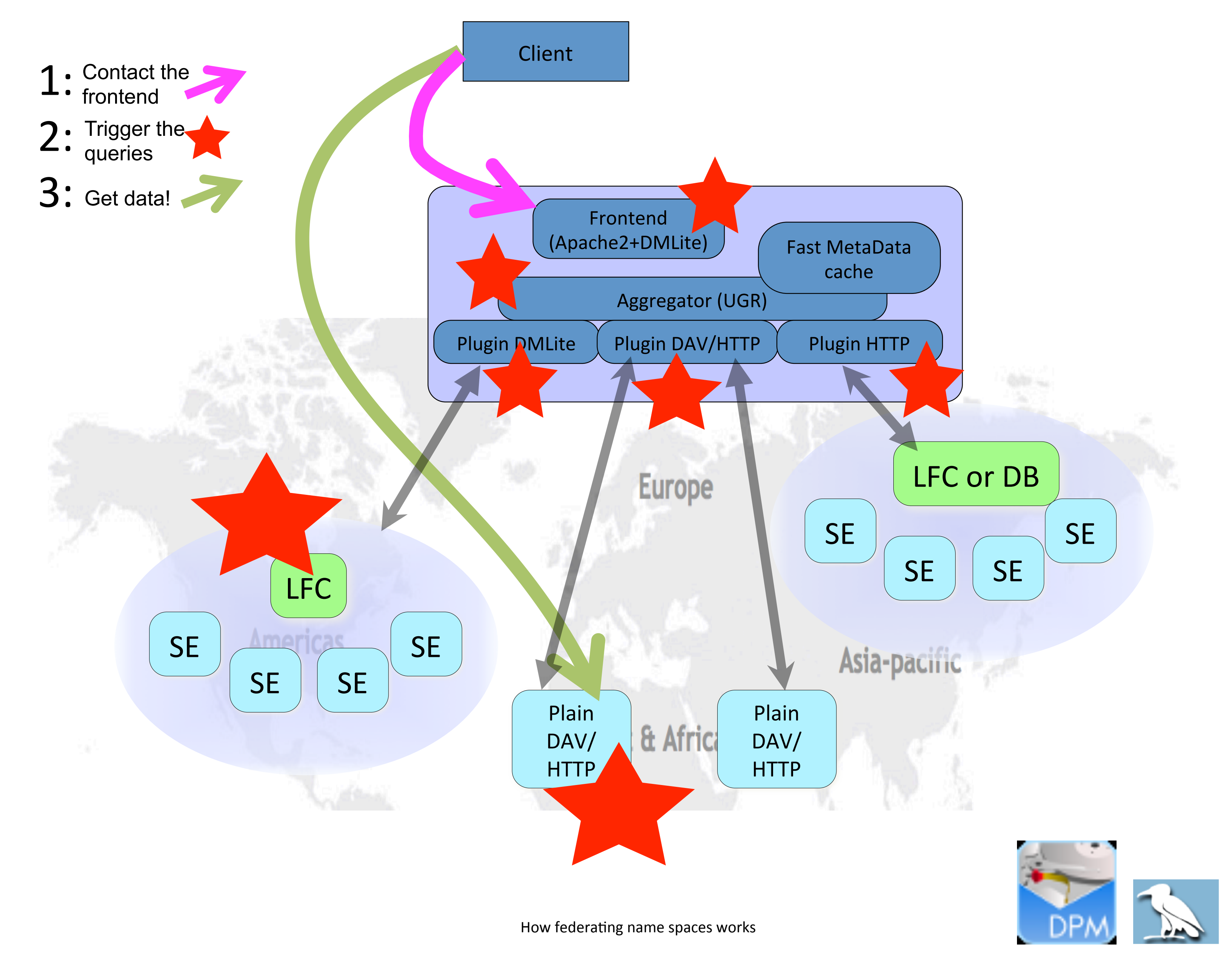
Fully support metadata browsing

No strange APIs, everything looks natural

Use dynamic systems that are easy to setup/maintain:

No complex metadata persistency

No DB babysitting (and lives well with the experiment's metadata repositories)



How federating name spaces works

Federations can help

- **Failover between islands**
- **Simplicity for personal interactive access, anything is anywhere**
- **'diskless' sites where data is nearby**
- **Storage sharing, eg 3 collaborating sites, each with 1/3 of required files**
- **Seamless integration of cloud storage**
- **Reduced data management for analysis, files fetched when needed**
- **Site 'overflow' (when jobs are waiting too long in a site queue, balance IO and CPUs)**

We federate (meta)data repositories that are 'compatible'

Name space (modulo simple prefixes)

Permissions (they don't contradict across sites)

Content (same key or filename means same file)

Dynamic, transparent metadata discover

looks like a unique, very fast file metadata system

properly presents the aggregated metadata views

redirects clients to the geographically closest endpoint

TODAY we can federate multiple, worldwide instances of:

dCache DAV/HTTP

DPM DAV/HTTP, LFC DAV/HTTP

Cloud DAV/HTTP

Native LFC and DPM databases (through DMLite used as a client)

Can be extended to other metadata sources

The system also can load a Geo plugin

Gives a geographical location to replicas and clients

Allows the core to choose the replica that is closer to the client

The one that's available uses GeoIP (free)* other implementations are possible

No central catalogue inconsistencies, by design

(central catalogue is not needed in a dynamic federation)

High Performance

Can be used by client tools that everybody knows

focus on HTTP/DAV, we can use it from a smartphone.

Base everything on open 'just works' technologies.

Use your browser to browse your data, anywhere, make the GRID jobs use the same

See poster "Web enabled data management with DPM&LFC"

* "This product includes GeoLite data created by MaxMind, available from http://maxmind.com/"

Technology

The core component is a plugin-based component called "Uniform Generic Redirector" (*Ugr*). It can plug into an Apache server thanks to the *DMLite* and *DAV-DMLite* modules (by IT-GT). Architecturally, Ugr acts as an information feeder for *DMLite*.

Ugr internally splits the queries into parallel tasks of information location

Composes on the fly the aggregated metadata views by managing these tasks and delaying clients the minimum amount of time that is necessary

Never stacks up latencies! No fixed delays.

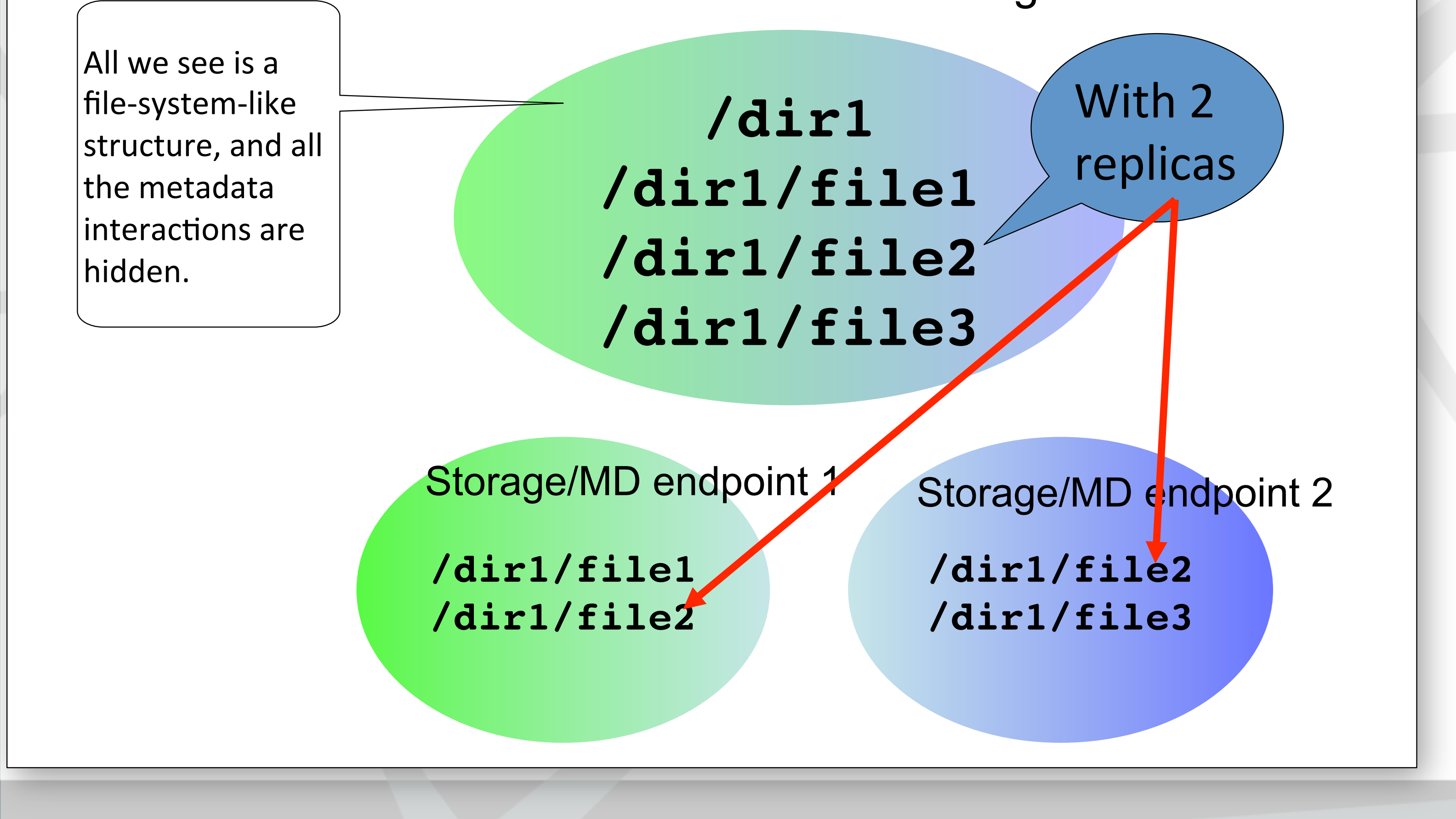
Able to handle file listings and metadata

Able to **redirect** clients to replicas

By construction, the internal workspace is a data structure that models a partial, volatile namespace

An LRU purging policy makes a fast, in-memory, 1st level cached namespace.

The basic idea behind a Storage Federation



Full parallelism, high performance

No limit to the number of outstanding clients/tasks

No global locks/serializations!

The endpoints are treated in a completely independent way

Thread pools, prod/consumer queues used extensively (e.g. to stat N items in M endpoints while X clients wait for some items)

Aggressive metadata caching

A relaxed, hash-based, in-memory partial name space

Juggles info in order to always contain what's needed

Stalls clients the minimum time that is necessary to juggle their information bits

Peak performance per CPU core: 0.5~1M stats/sec

High performance DAV client implementation (DAVIX)

Loaded by the core as a "location" plugin

Uses libneon w/ sessions caching

Compound list/stat operations

Seamless storage federations of:

Official Storage Elements, LFCs, catalogues…

Cached data (i.e. SQUID-like things, not registered in any catalogue)

HTTP/DAV-based servers

Cloud storage services

HTTP-enabled XROOTD/EOS clusters, sharing the data.

Local SE as a preference, give the freedom to point to an **efficient** and **reliable** global federation

Optimize redirections based on on-the-fly client-data proximity

Limit complexity: read only

Usually writes happen to well-known, close islands

Detailed performance measurements are on the way.

CERN IT Department
CH-1211 Geneva 23
Switzerland
www.cern.ch/it

For more information: www.eu-emi.eu