Dynamic Federations
Storage federations for HTTP and WebDAV

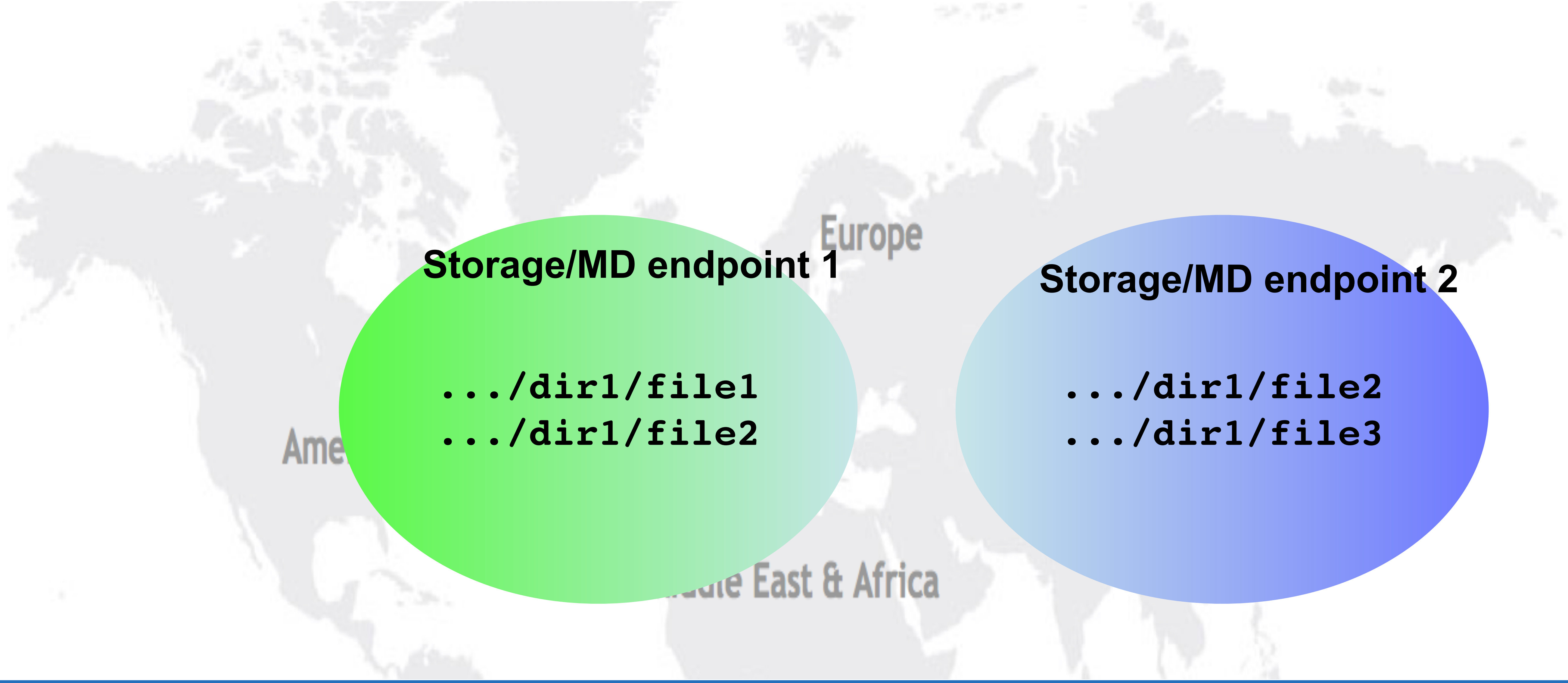
Fabrizio Furano (presenter)
Adrien Devresse
CERN IT-SDC

The problem

- Pick a number of generic HTTP/WebDAV storage endpoints, Grid or commercial “clouds”
- We want to see and use them as an unique seamless multipetabyte, high performance system

- HTTP supports redirecting clients to get to the data
- The challenging problems are:
  - “Where is File X ?”
  - “What’s the content of /myfolder, worldwide ?” Be quick to browse it!

- Our answer is:
  - Smart, efficient, seamless metadata discovery and caching
  - Flexible WebDAV, HTTP and HTML presentation
  - Flexibility of interfacing to various existing and future infrastructures



Storage/MD endpoint 1

```
.../dir1/file1
.../dir1/file2
```

Storage/MD endpoint 2

```
.../dir1/file2
.../dir1/file3
```

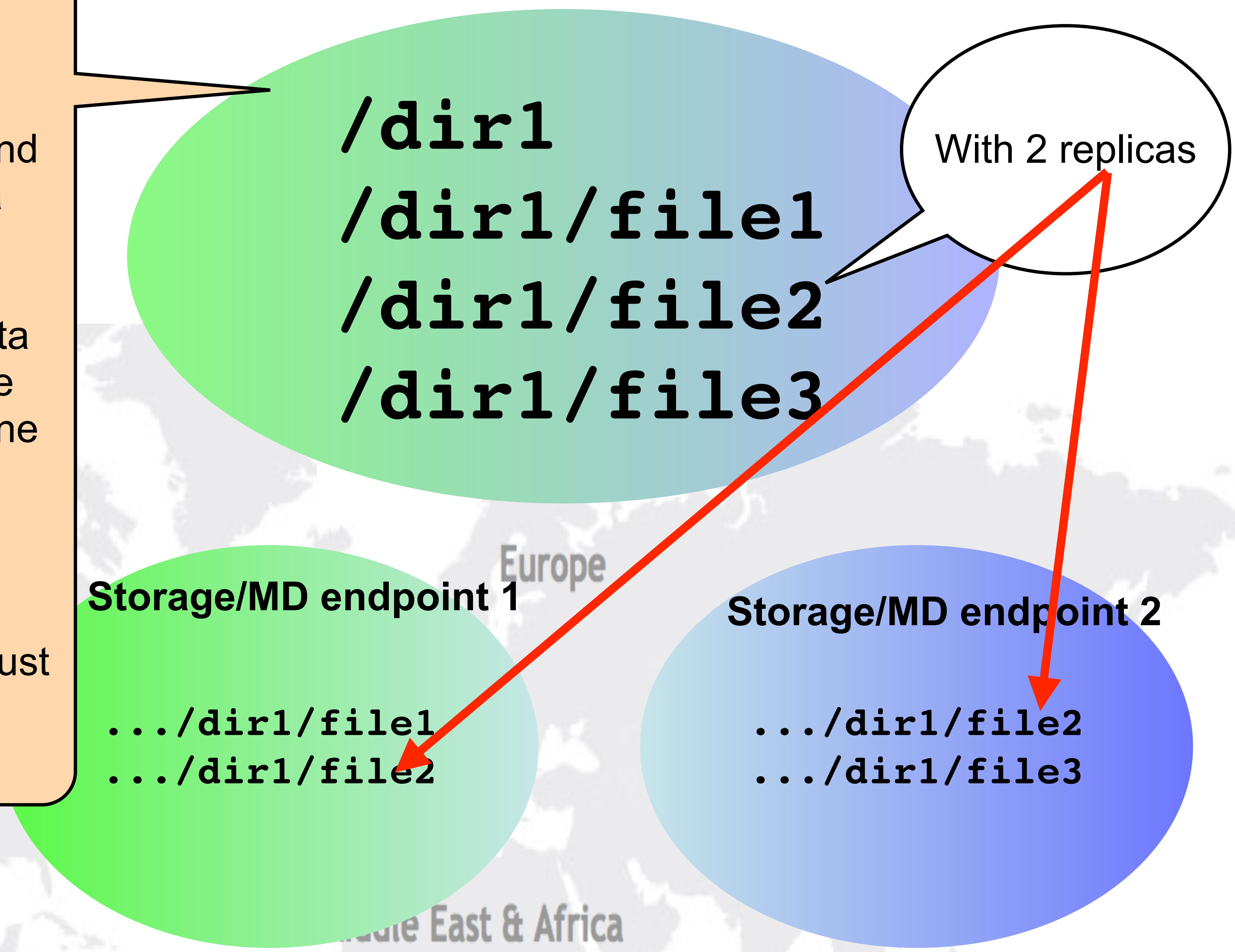
Aggregation

This is what we want to see as users

Sites remain independent and participate to a global view

All the metadata interactions are hidden and done on the fly

NO metadata persistency needed here, just efficiency and parallelism



/dir1
/dir1/file1
/dir1/file2
/dir1/file3

With 2 replicas

Storage/MD endpoint 1

.../dir1/file1
.../dir1/file2

Storage/MD endpoint 2

.../dir1/file2
.../dir1/file3

Why HTTP/DAV?

- It's there, whatever platform we consider
  - A very widely adopted technology

- We (humans) like browsers, they give an experience of simplicity

- Goes towards convergence
  - Users can use their devices to access their data easily, out of the box
  - Web applications development can meet Grid computing
  - Jobs and users just access data directly, in the same way
  - Can more easily be connected to commercial systems and apps

Dynamic Federations

- An interactively browsable system able to discover dynamically its **metadata** content and present it to the clients
- Supports replicas AND listings
- Browse and access a huge repository made of many sites without requiring a static index
  - No "registration", no maintenance of catalogues
- If catalogues are needed, can talk to more than one at the same time. Acts as a "Catalogue access accelerator"
- Redirect intelligently clients asking for replicas
  - Automatically detect and avoid sites that go offline
  - Can make client-dependent choices on the fly
- Accommodate algorithmic name translations
  - E.g. to correctly map on the fly existing SRM TURLS to HTTP Urls
- Accommodate client-geography-based redirection choices
- Dynamic partial namespace caching: fast and scalable

Dynamic Federations

- Opens to a multitude of use cases, by composing a worldwide system from macro building blocks speaking HTTP and/or WebDAV
  - Federate third party outsourced HTTP/DAV servers
  - Federate the content of fast changing things, like file caches
  - Federate them together with the information of some experiment’s DB
  - Clients are redirected to the replica closer to them
  - Redirect only to working endpoints
  - Accommodate whatever metadata sources, even two or more remote catalogues at the same time
  - Accommodate whatever other Cloud-like storage endpoint

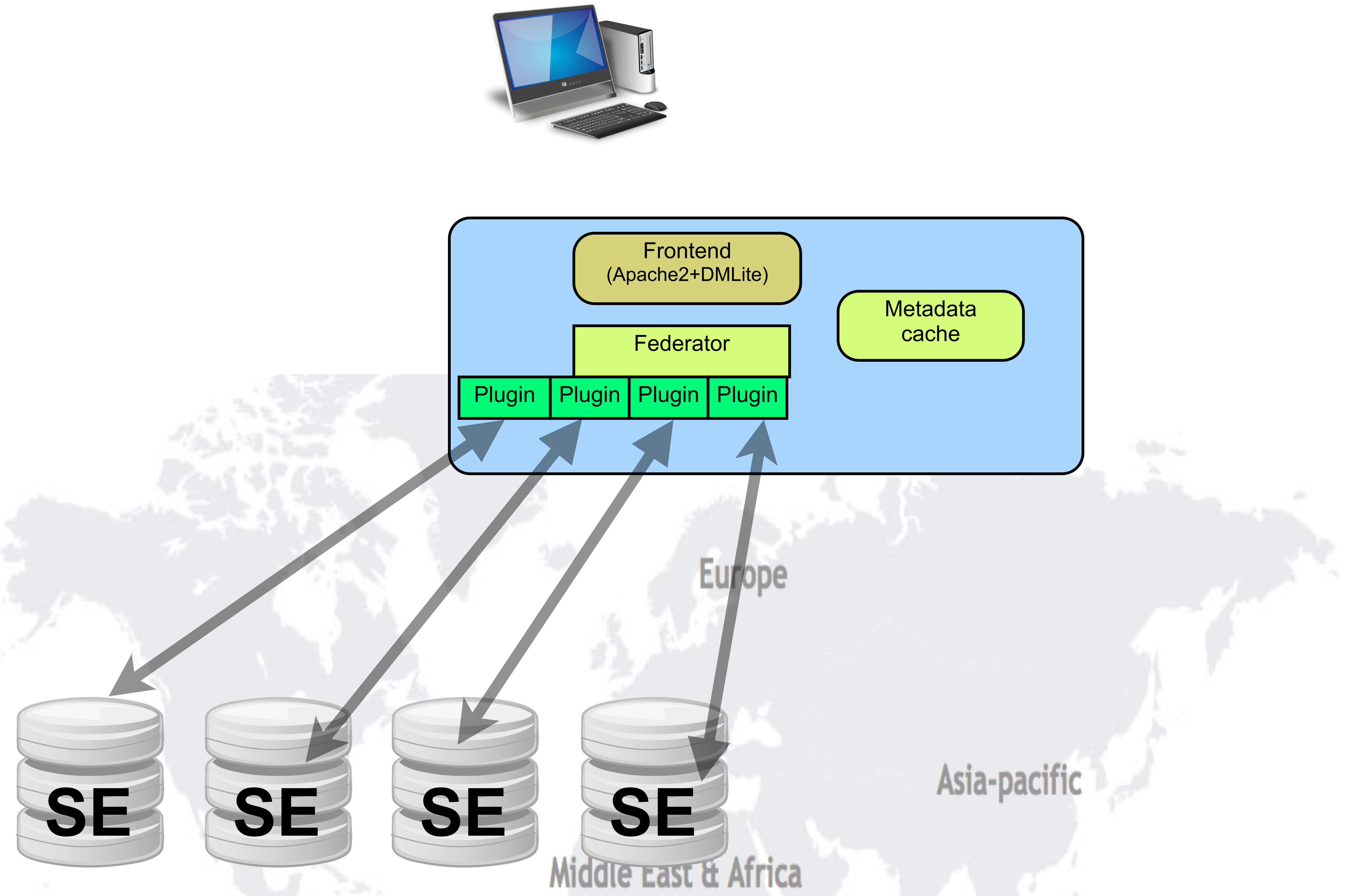
Some deployment examples

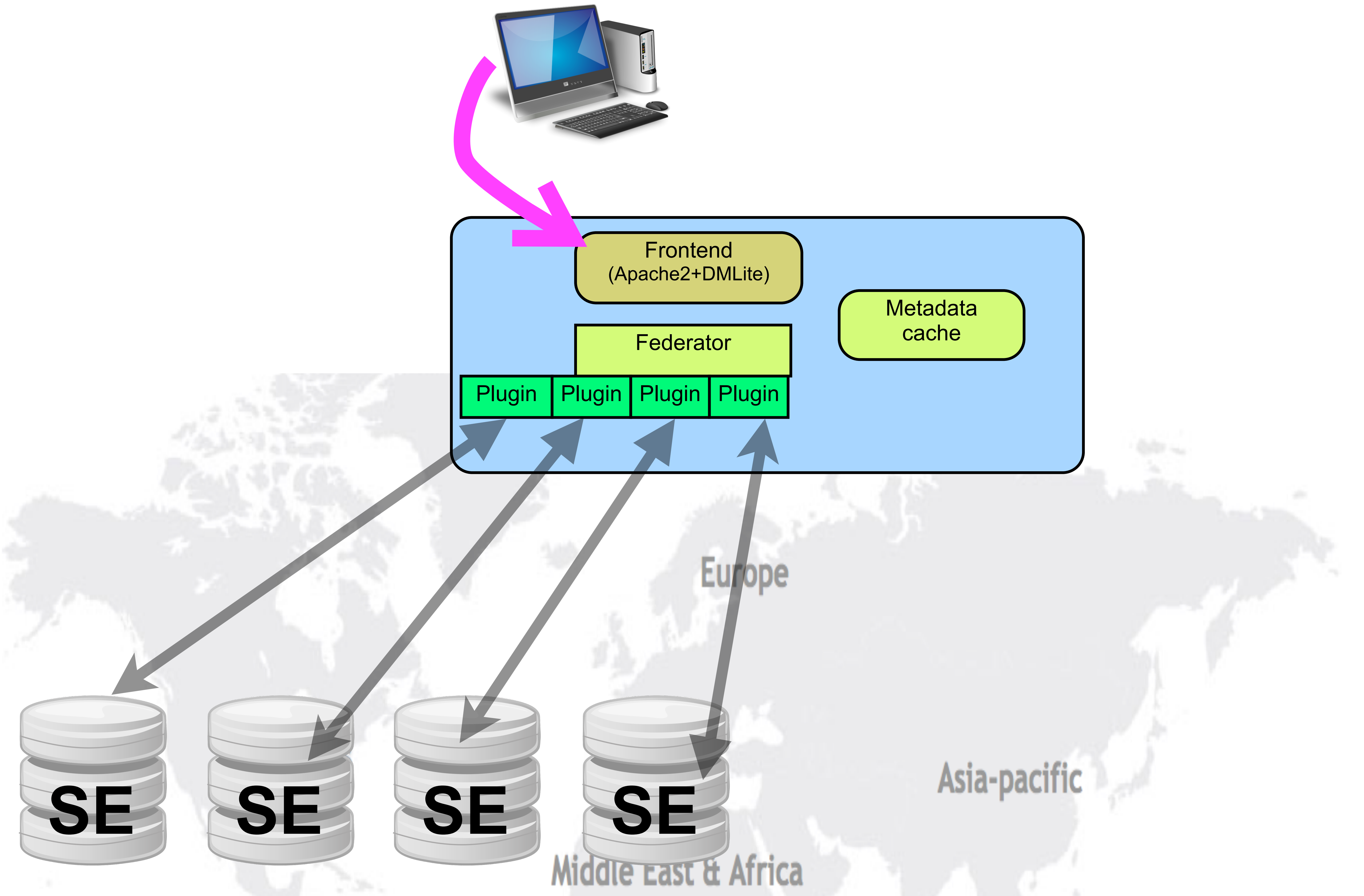
Example #1

- Aggregate multiple DAV servers into a federation
  - Similar to the xrootd federations
- Plus HTTP/DAV browsing and fast rendering of global file listings
  - User-friendly! No quirks, looks banal and comfortable.

- In this case the storage endpoints are considered as
  - Listing providers (for their own listings, if they support it)
  - Replica containers (for their own files)
  - The animation shows the replica location case

- Can be used internally to a site to aggregate instances of Xrootd with XrdHTTP and any other WebDAV endpoint
  - Set up Xrootd clusters that are efficiently browseable
  - See the presentation on XrdHTTP


Frontend
(Apache2+DMLite)
Metadata cache
Federator
Plugin
Plugin
Plugin
Plugin
Europe
SE
SE
SE
SE
Asia-pacific
Middle East & Africa



Frontend (Apache2+DMLite)

Metadata cache

Federator

Plugin

Plugin

Plugin

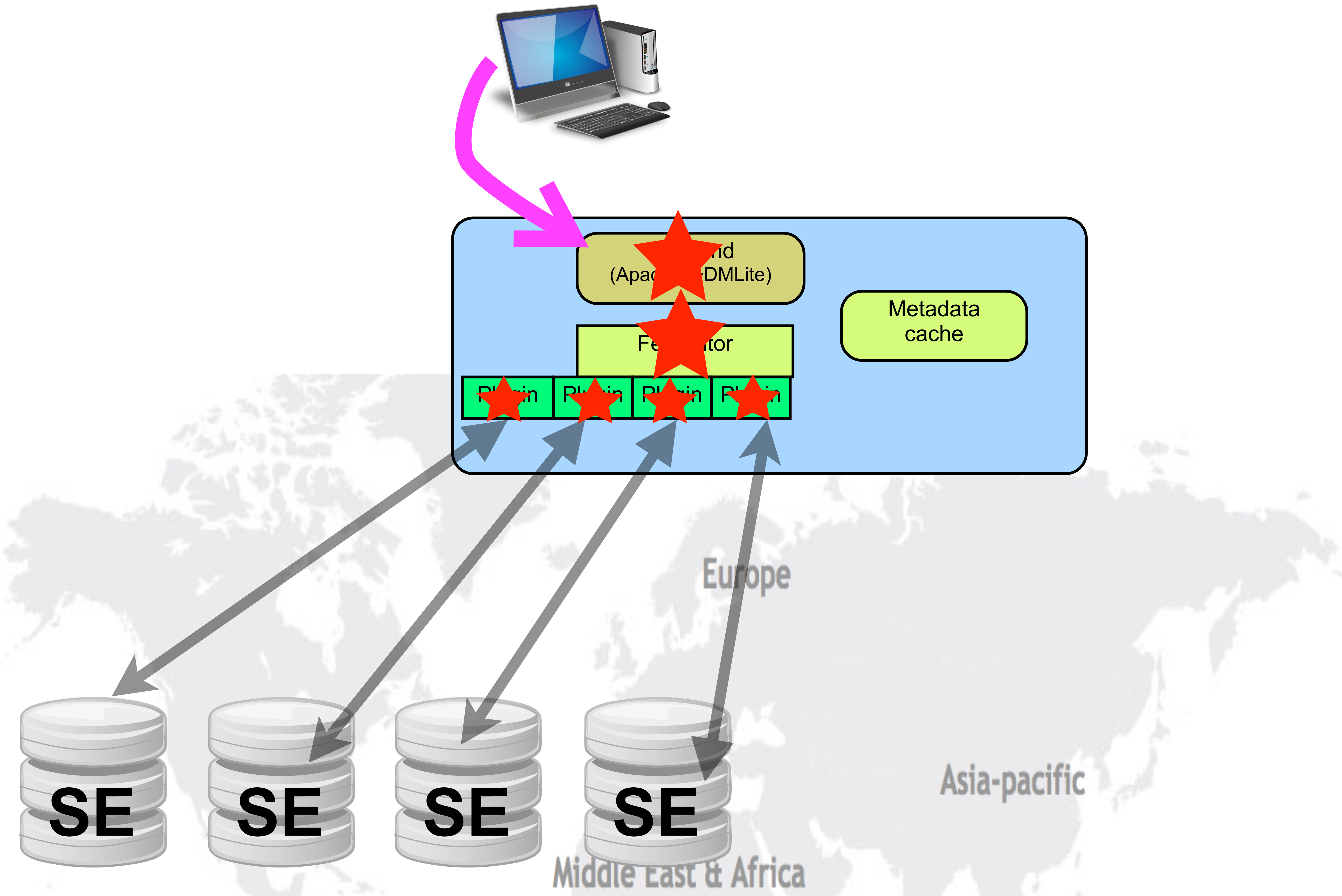
Plugin

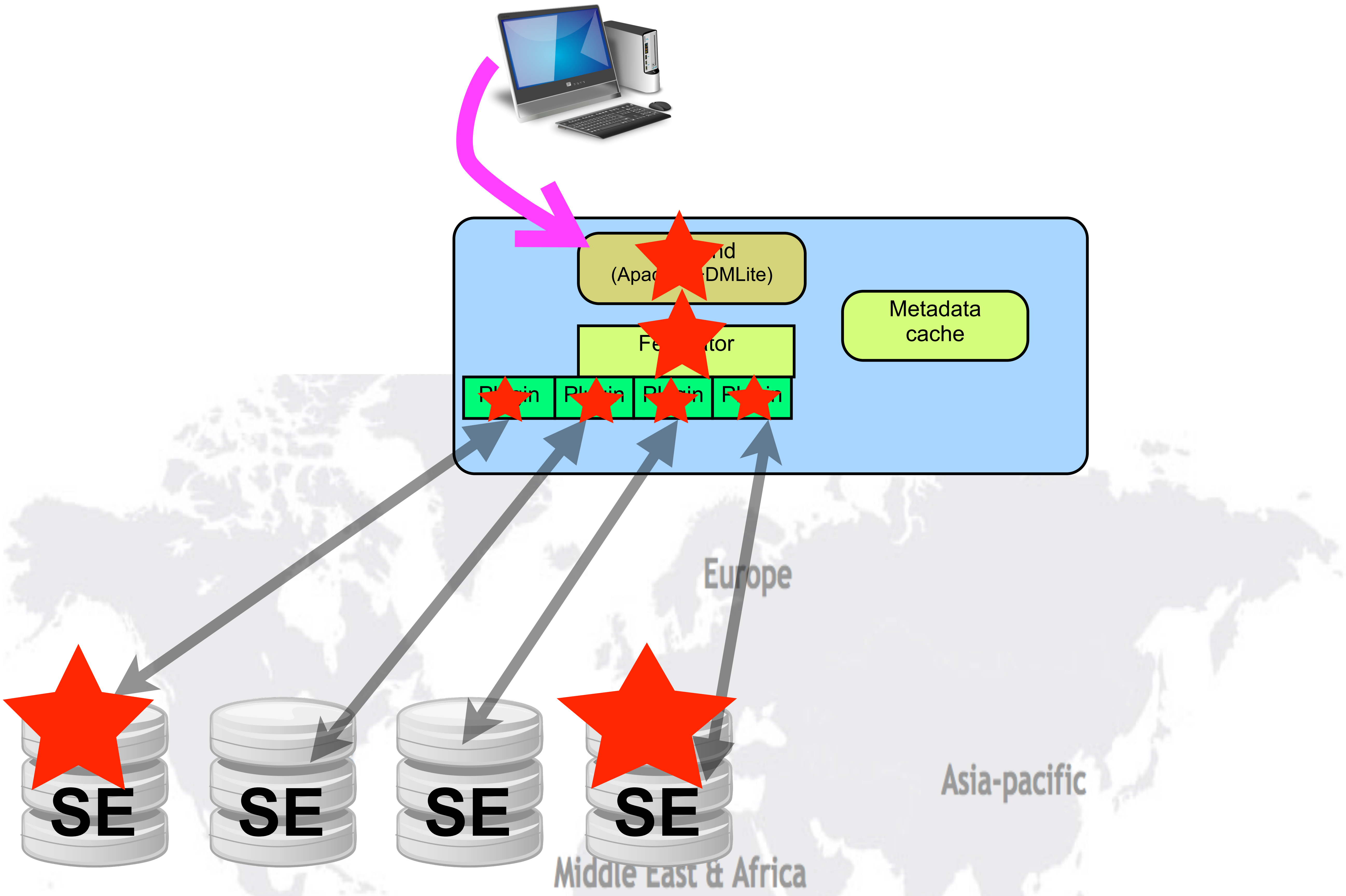
SE

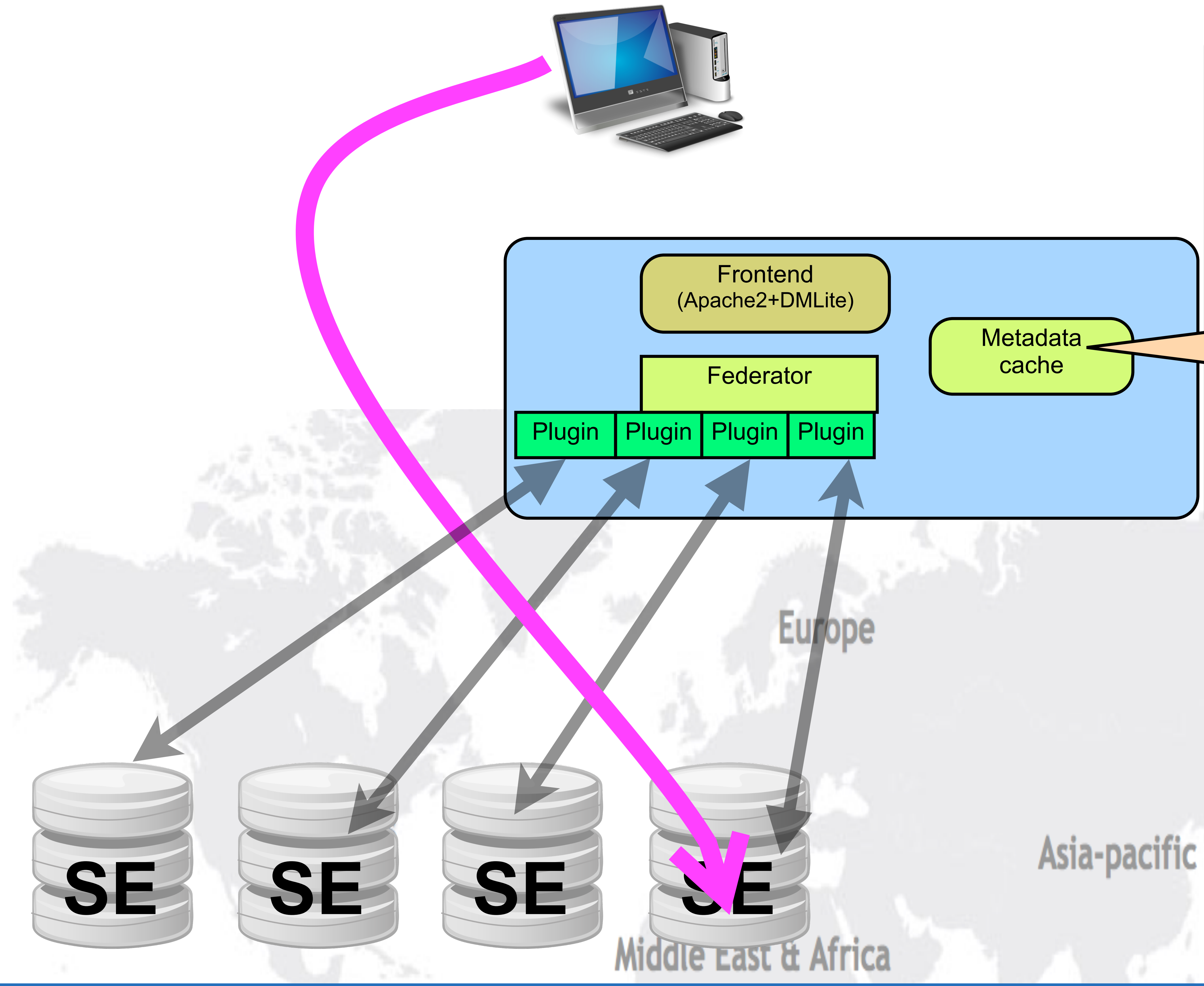
SE

SE

SE


nd
(Apac -DMLite)
Metadata cache
Fe tor
Pl in
Pl in
Pl in
P in
Europe
Asia-pacific
Middle East & Africa
SE
SE
SE
SE


nd
(Apac -DMLite)
Fe tor
Metadata cache
Pl in
Pl in
Pl in
Pl in
Europe
Asia-pacific
Middle East & Africa
SE
SE
SE
SE



The cache remembers what happened

The next **metadata** interactions will very likely be fed by the cache

The cache can be shared

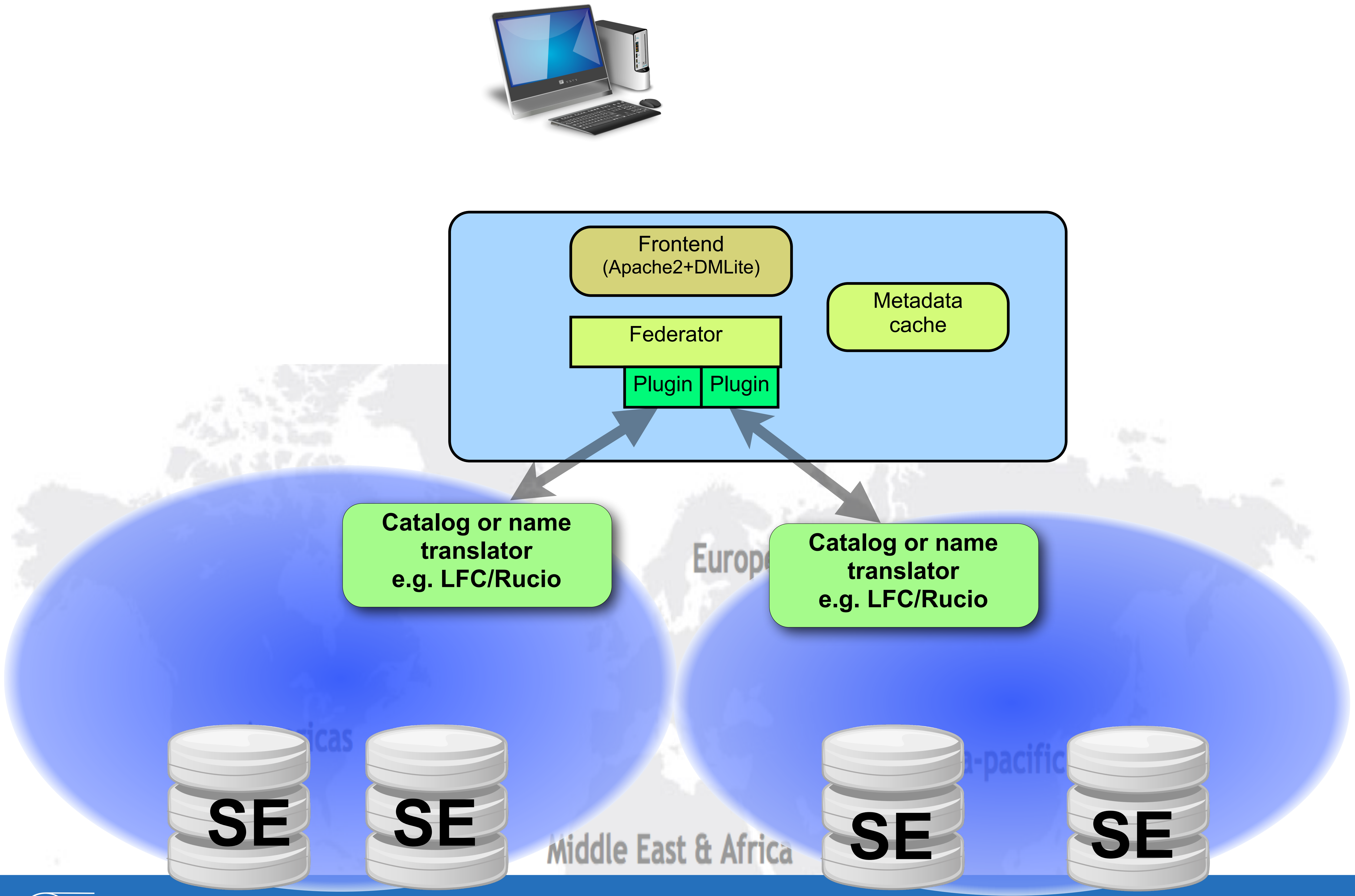
Example #2

- DAV metadata catalogues
  - E.g. LFC, Rucio or whatever else is similar

- In this case the catalogues are considered as
  - Listing providers (if they support it)
  - Replica locators and name translators
  - The animation shows the replica location case

- The SEs can be anything that supports HTTP data access
  - I federated my Dropbox with Patrick's DT cloud plus DPM and dCache

- The dynafed looks like a browseable catalogue that has the content of both
- Performance is faster than the fastest of the two.
- Maximum latency with cold cache is one network roundtrip to the most distant endpoint



Frontend (Apache2+DMLite)

Metadata cache

Federator

Plugin

Plugin

Catalog or name translator e.g. LFC/Rucio

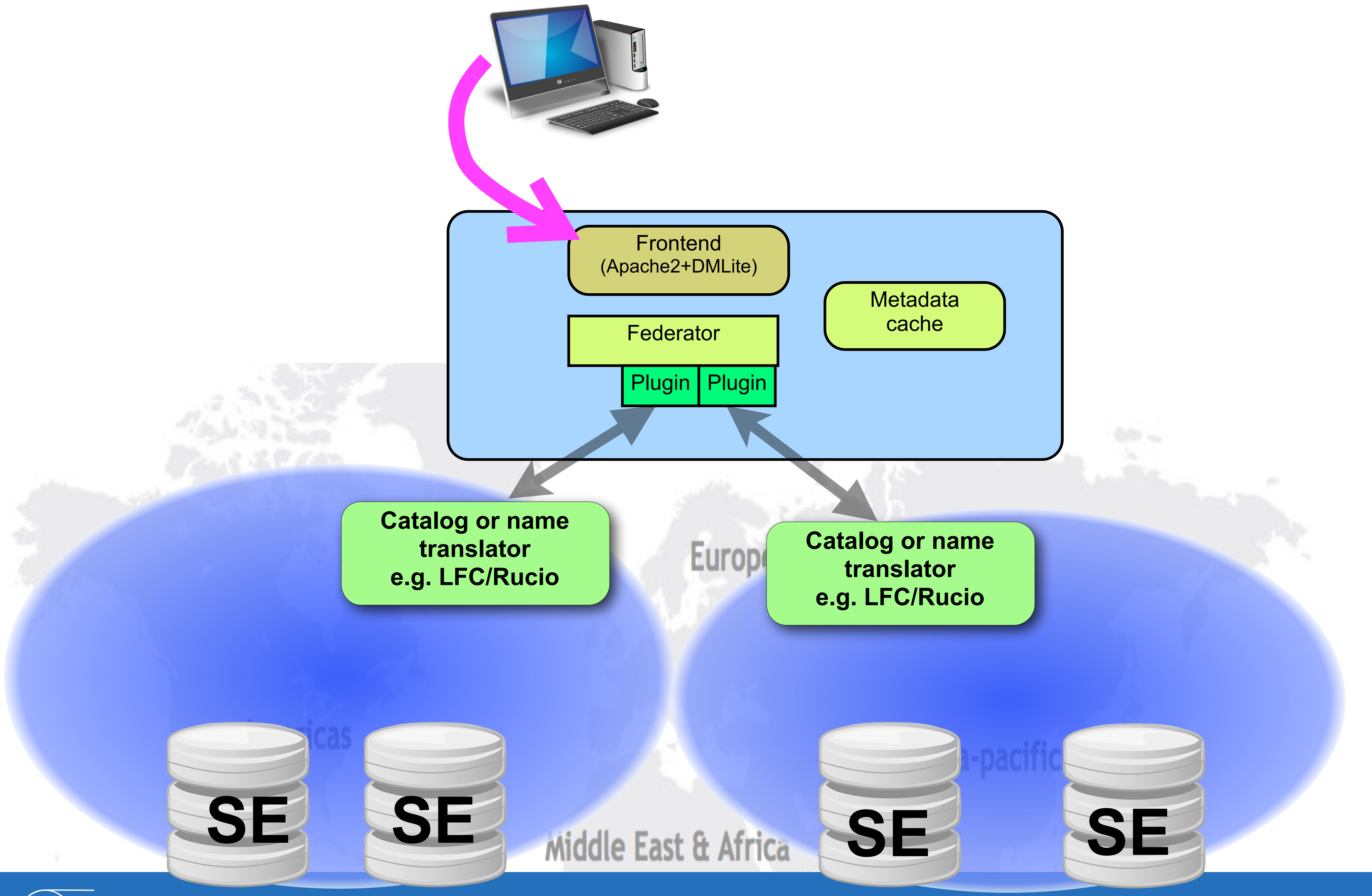
Catalog or name translator e.g. LFC/Rucio

SE

SE

SE

SE



Frontend (Apache2+DMLite)

Metadata cache

Federator

Plugin

Plugin

Catalog or name translator e.g. LFC/Rucio

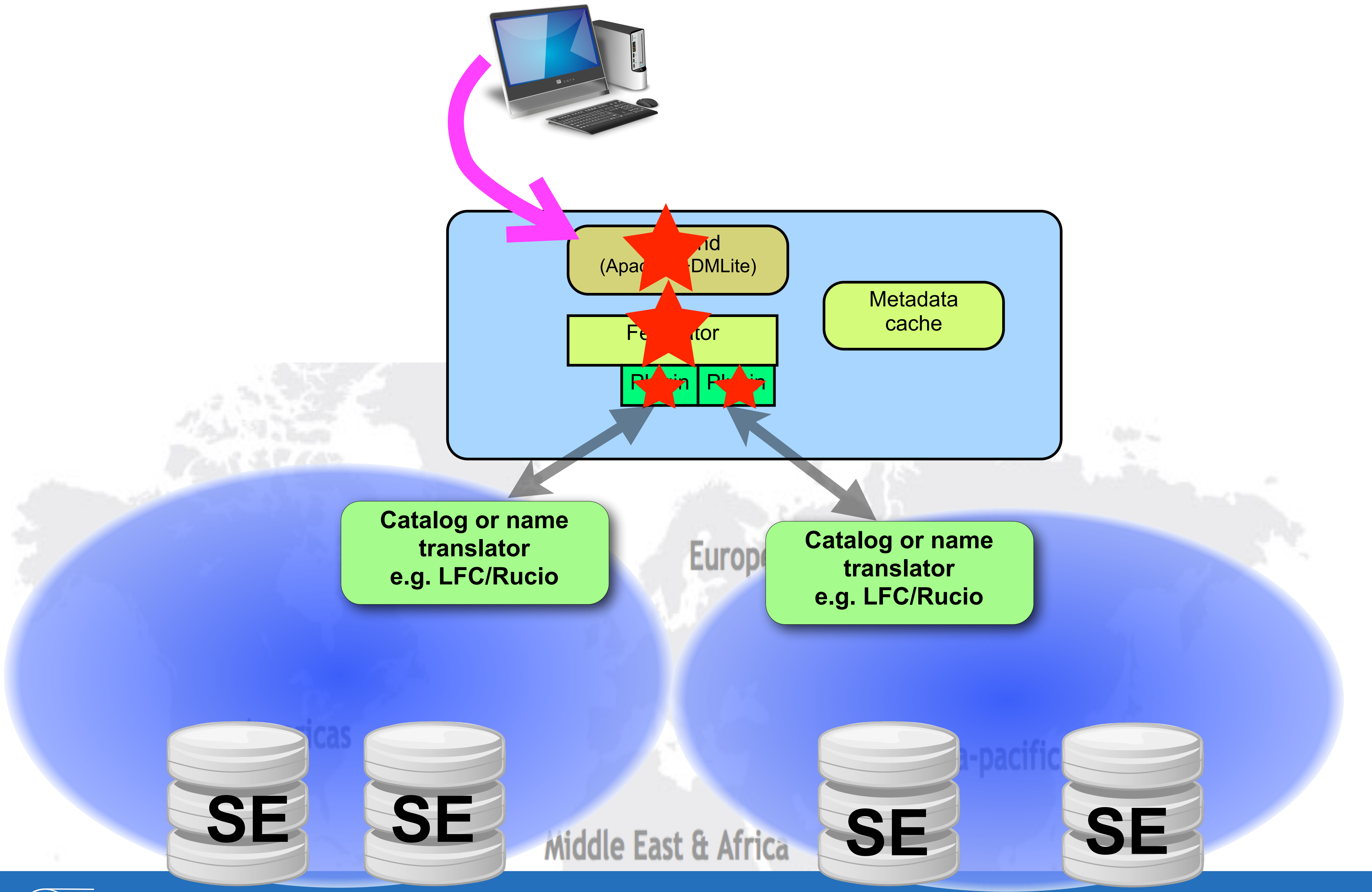
Catalog or name translator e.g. LFC/Rucio

SE

SE

SE

SE



nd
(Apac -DMLite)

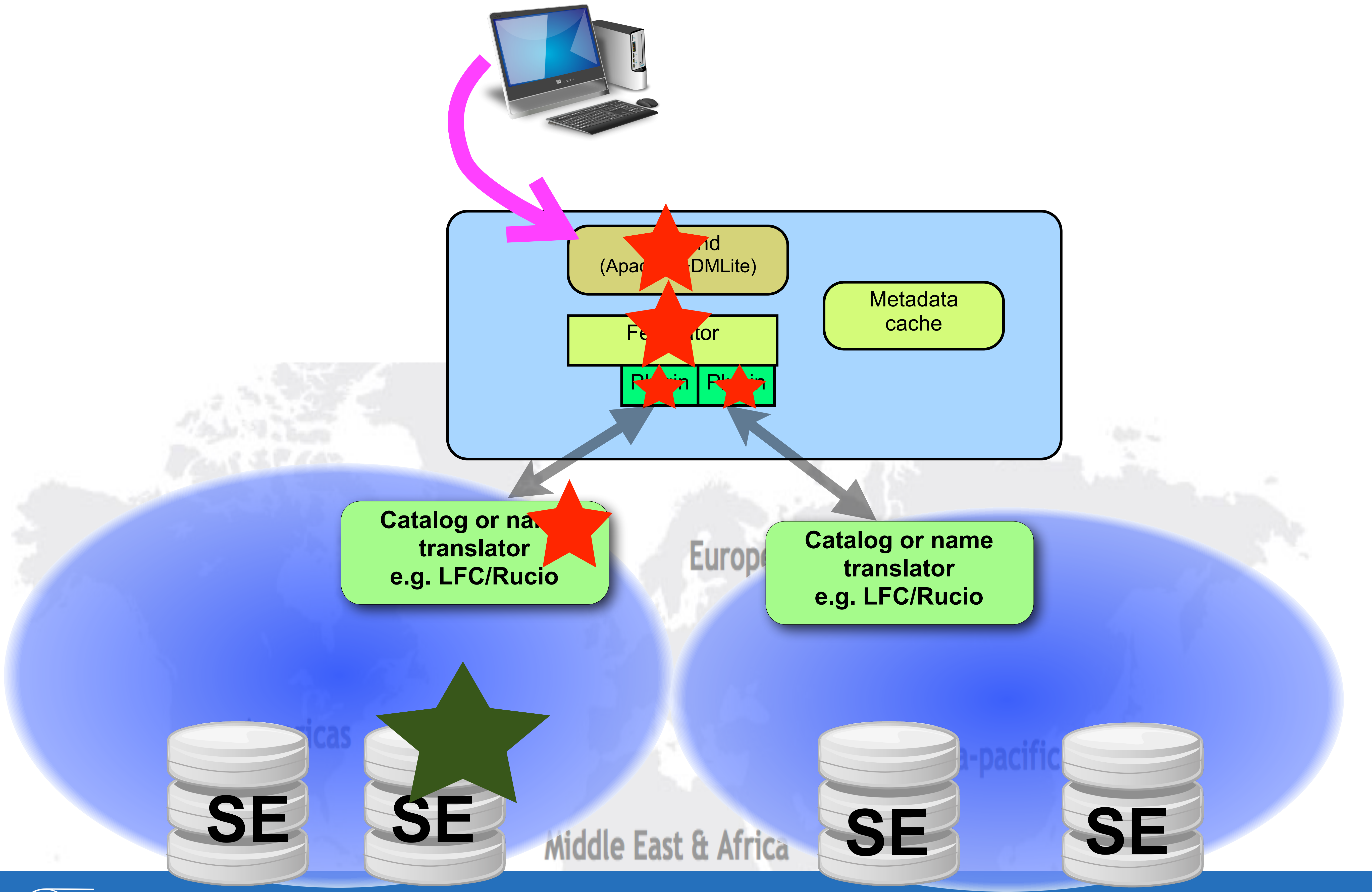
Fe tor

Pl in Pl in

Metadata cache

Catalog or name translator
e.g. LFC/Rucio

Catalog or name translator
e.g. LFC/Rucio



nd
(Apac -DMLite)

Fe tor

Pl in Pl in

Metadata
cache

Catalog or na
translator
e.g. LFC/Rucio

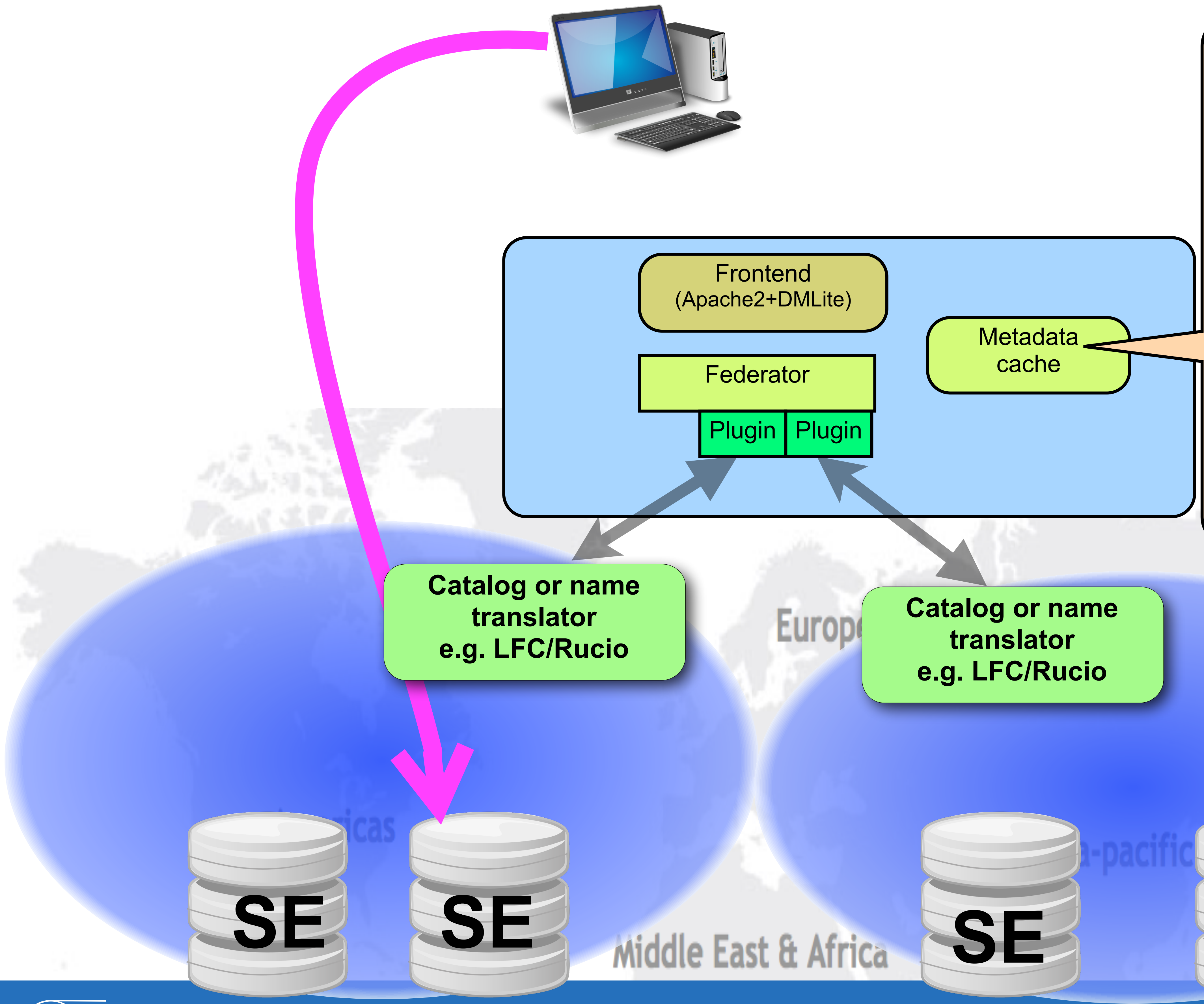
Catalog or name
translator
e.g. LFC/Rucio

SE

SE

SE

SE



The cache remembers what happened

The next **metadata** interactions will very likely be fed by the cache

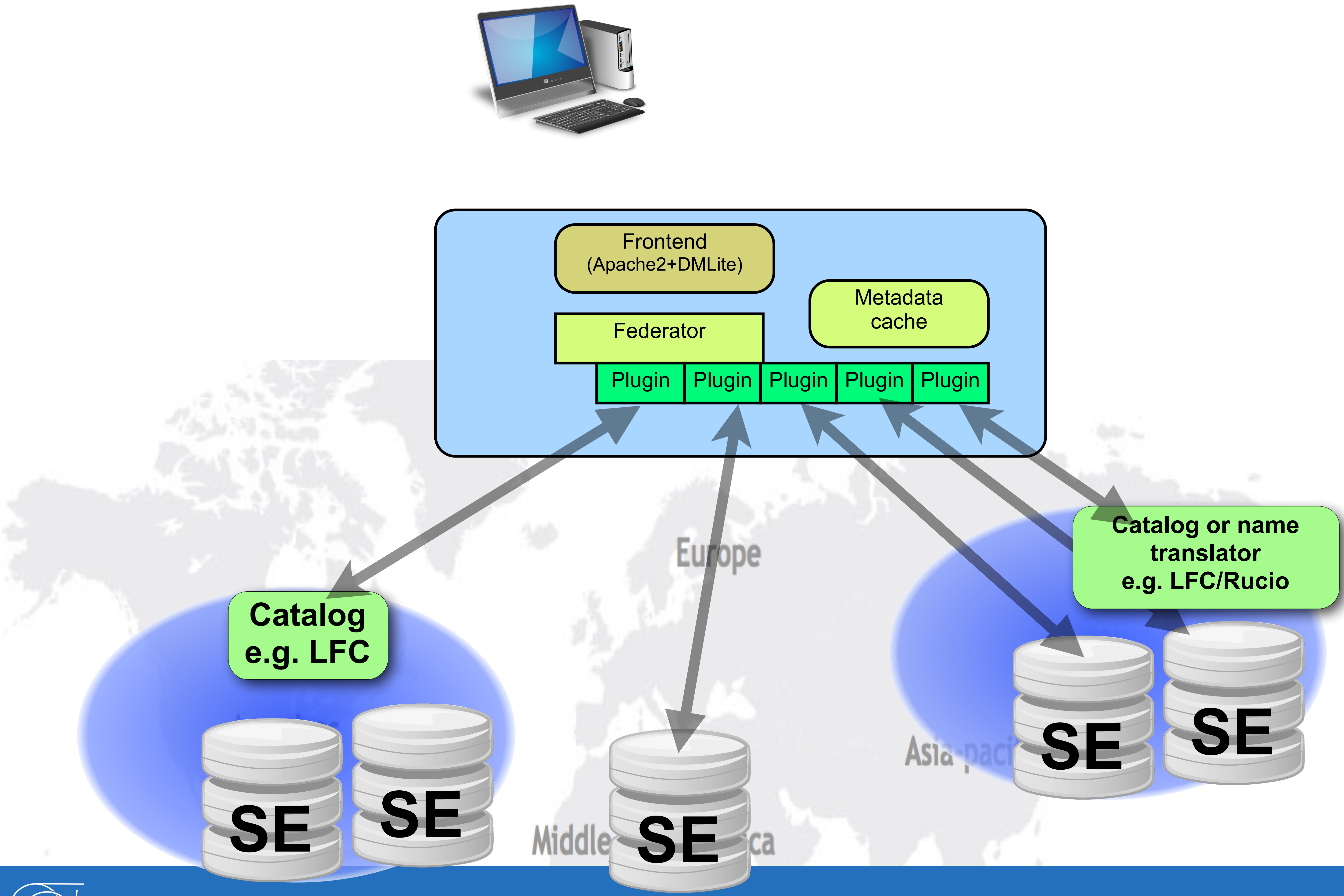
The cache can be shared

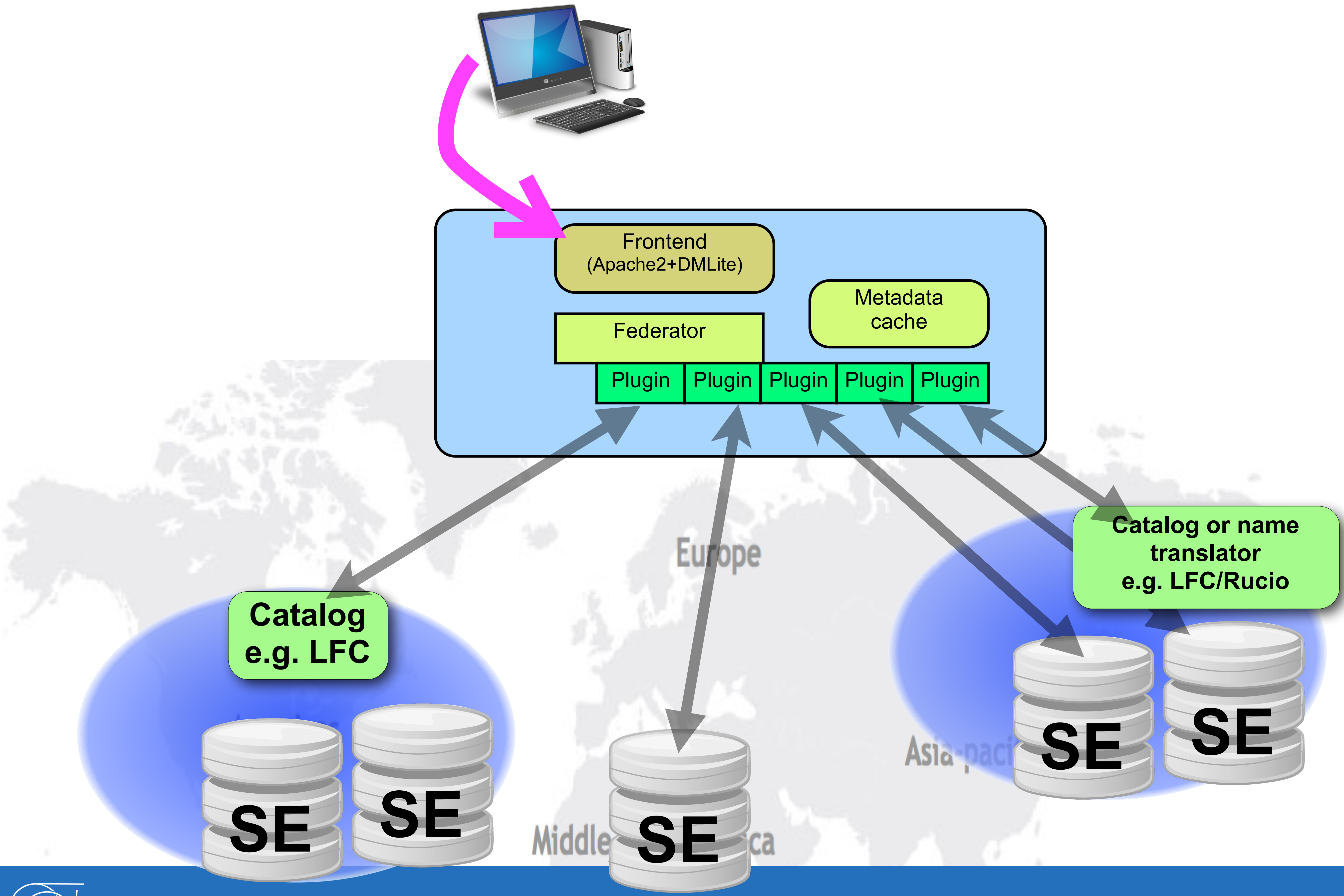
Example #3

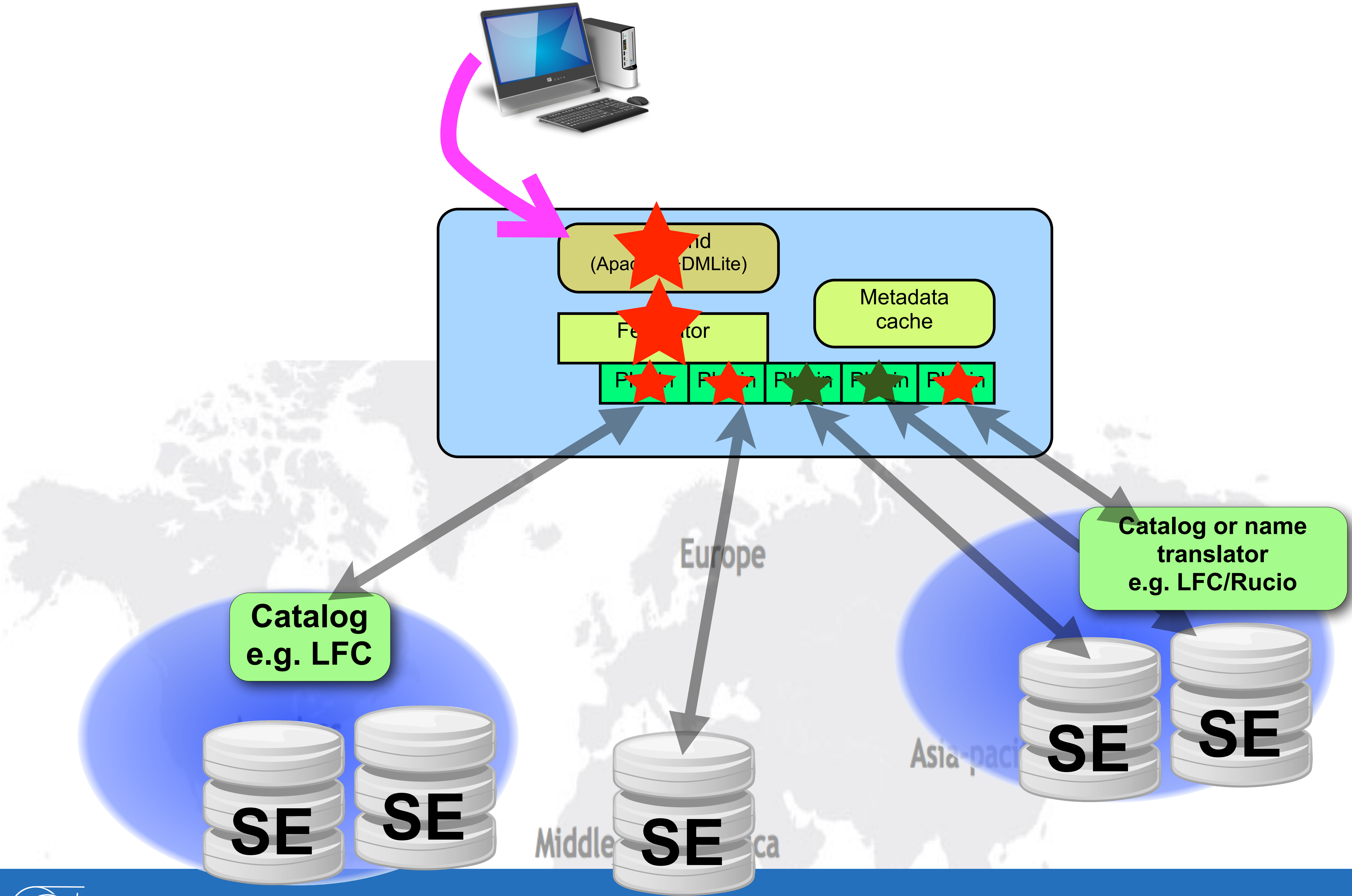
- Federating it all together:
    - Catalogues with SEs connected to the federator
    - Catalogues with SEs disconnected from the federator
    - Standalone storage endpoints (can be caches or cloud services)

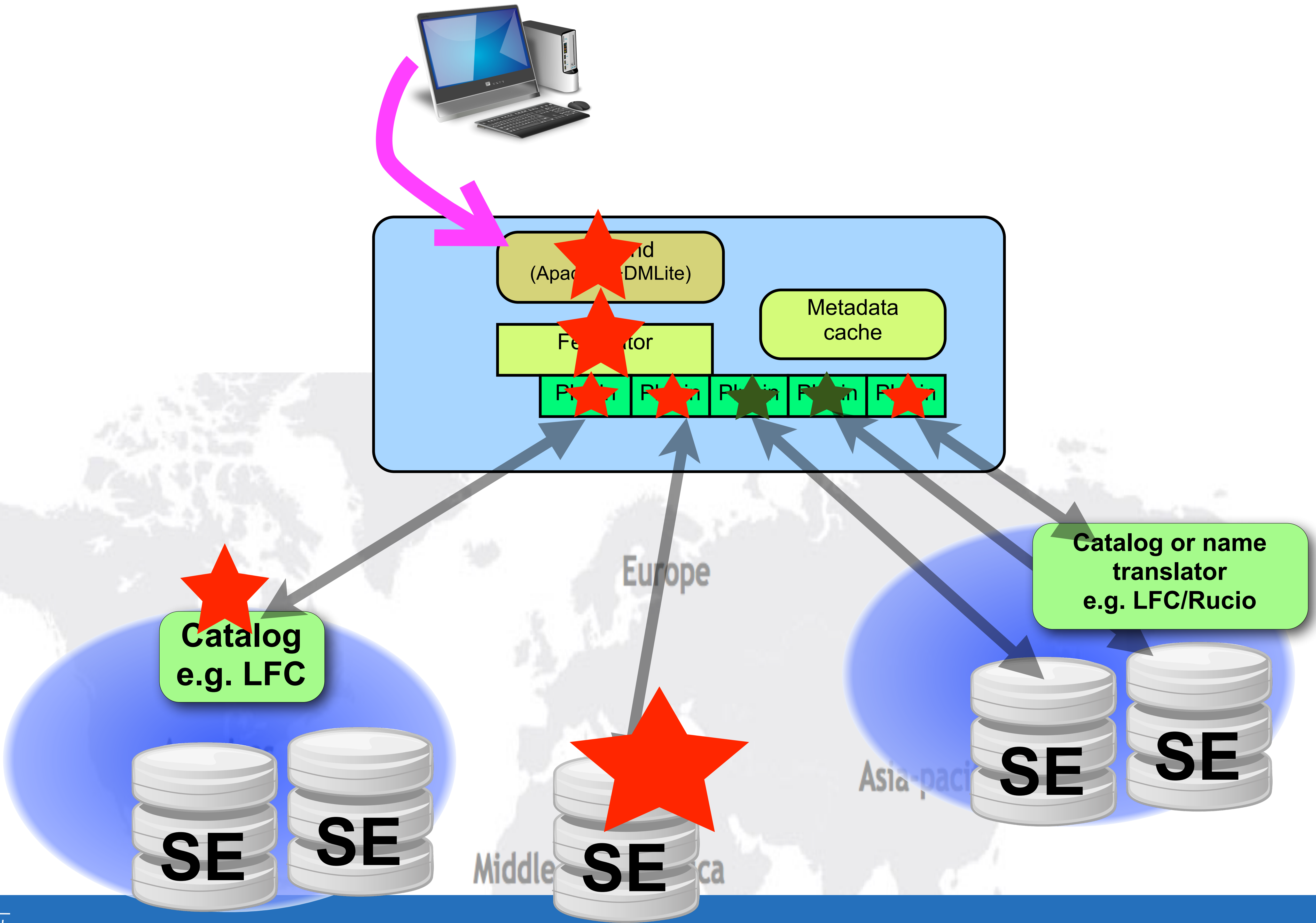
- In this case the catalogues are considered as
    - Listing providers (if they can do it)
    - Replica locators and name translators
- In this case the storage endpoints can be whatever, depending on how we connect them
    - Listing providers (for their own listings, if they support it)
    - Replica containers (for their own files)
    - Standalone servers, clusters or site caches

- The dynafed looks like a browseable catalogue that has the full content
- A replica request will redirect following the response of the ‘best’ storage element
- Files with no replicas will still be visible in the browser


Frontend
(Apache2+DMLite)
Metadata cache
Federator
Plugin
Plugin
Plugin
Plugin
Plugin
Europe
Catalog e.g. LFC
Catalog or name translator e.g. LFC/Rucio
SE
SE
SE
SE
SE
Asia-paci
Middle
ca


Frontend
(Apache2+DMLite)
Metadata cache
Federator
Plugin
Plugin
Plugin
Plugin
Plugin
Europe
Catalog or name translator e.g. LFC/Rucio
Catalog e.g. LFC
SE
SE
SE
SE
SE
Asia-paci
Middle
ca


nd
(Apac -DMLite)
Fe tor
Metadata
cache
Pl n
Pl in
Pl in
Pl in
Pl in
Europe
Catalog or name
translator
e.g. LFC/Rucio
Catalog
e.g. LFC
SE
SE
SE
SE
SE
Middle ca
Asia-paci



nd
(Apac -DMLite)

Fe tor

Metadata cache

Pl in | Pl in | Pl in | Pl in | Pl in

Catalog e.g. LFC

Catalog or name translator e.g. LFC/Rucio

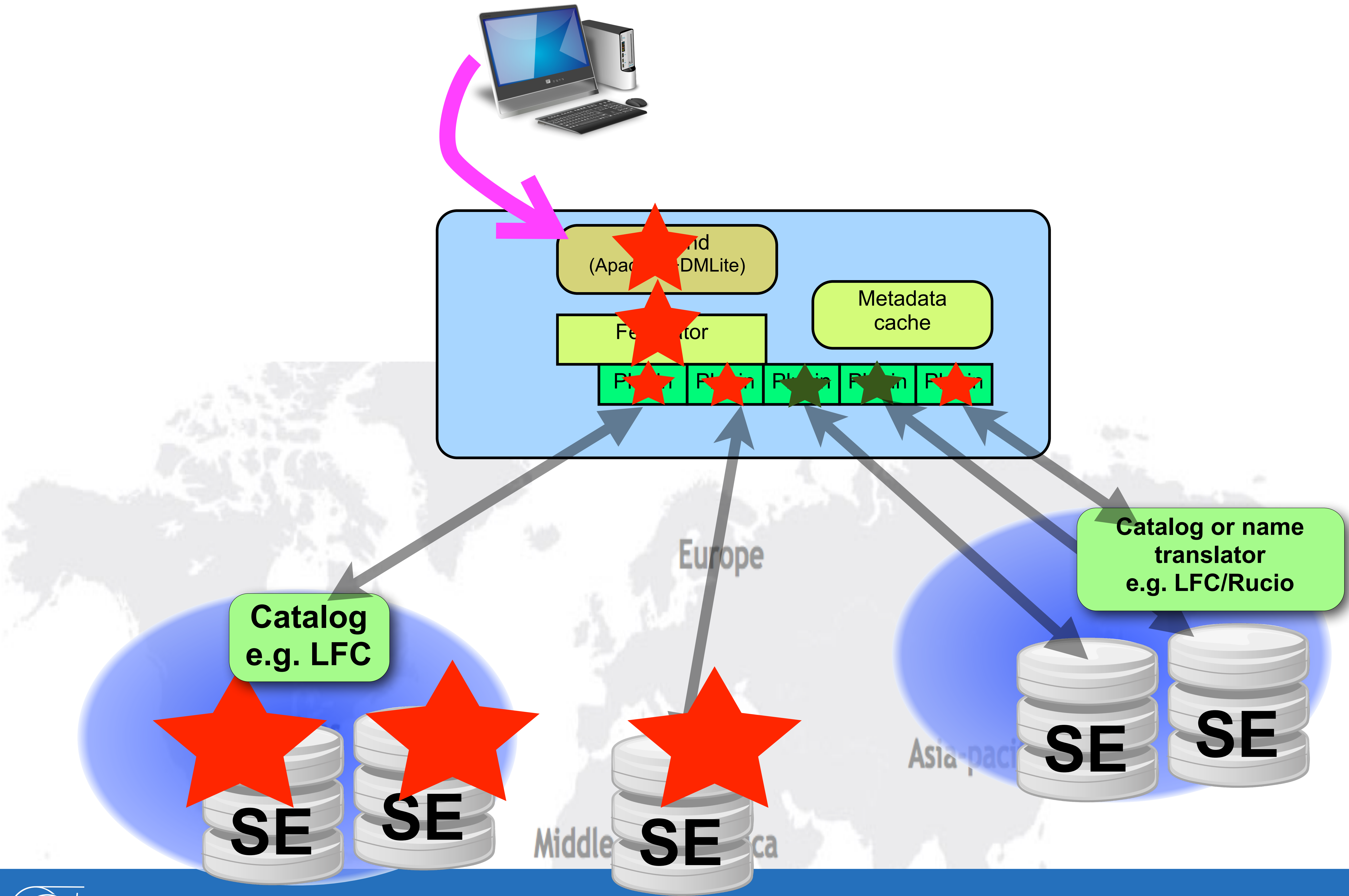
SE

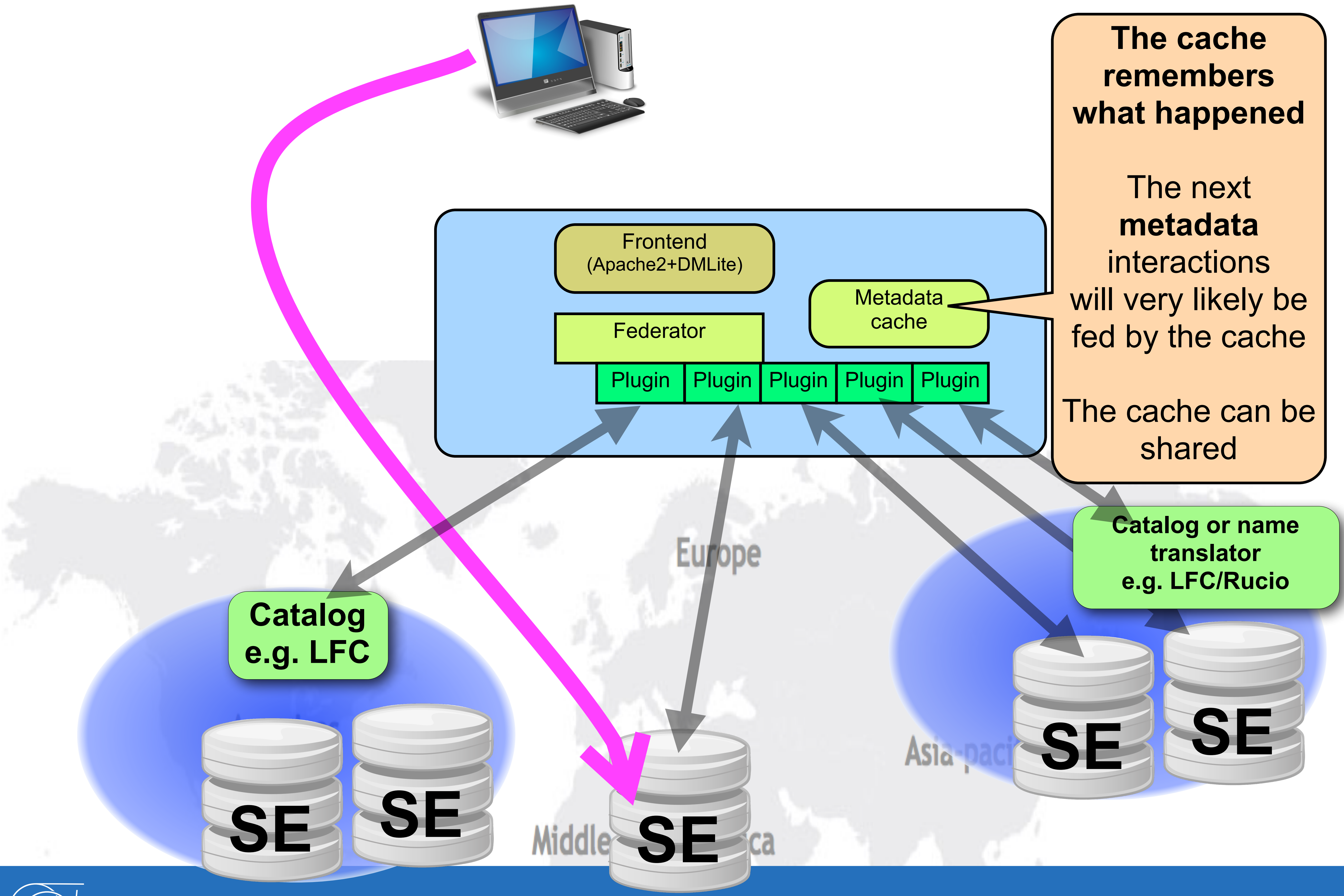
SE

SE

SE

SE


nd
(Apac -DMLite)
Fe tor
Metadata cache
Pl n
Pl in
Pl in
Pl in
Pl in
Catalog
e.g. LFC
Catalog or name translator
e.g. LFC/Rucio
Europe
Middle
ca
Asia-paci
SE
SE
SE
SE
SE



The cache remembers what happened

The next **metadata** interactions will very likely be fed by the cache

The cache can be shared

Demo testbed

- We have a stable demo testbed, using HTTP/DAV http://federation.desy.de/
- It is actually 2 demos in one
  - A fully dynamic catalogue-free demo between DESY, CERN
  - A small ATLAS demo, federating 8 sites, plus LFC as name translator and listing provider
    - Note that this is not the full ATLAS repo, it's just 8 sites like Example #3. Most of the files have no replicas. Note that the client is never redirected to unknown places
- **The feeling it gives is surprising and (un)impressively normal**
  - Browsing performance is in avg much higher than contacting the endpoints
- We see the directories as merged, as if it were only one system
- 10K files are interleaved in a 4-levels deep directory /fed/interleaved
  - Oddly-numbered files are at CERN ,evenly-numbered files are at Desy
- 10K files have 2 replicas in DESY and CERN: /fed/everywhere

NEP-101

canarie

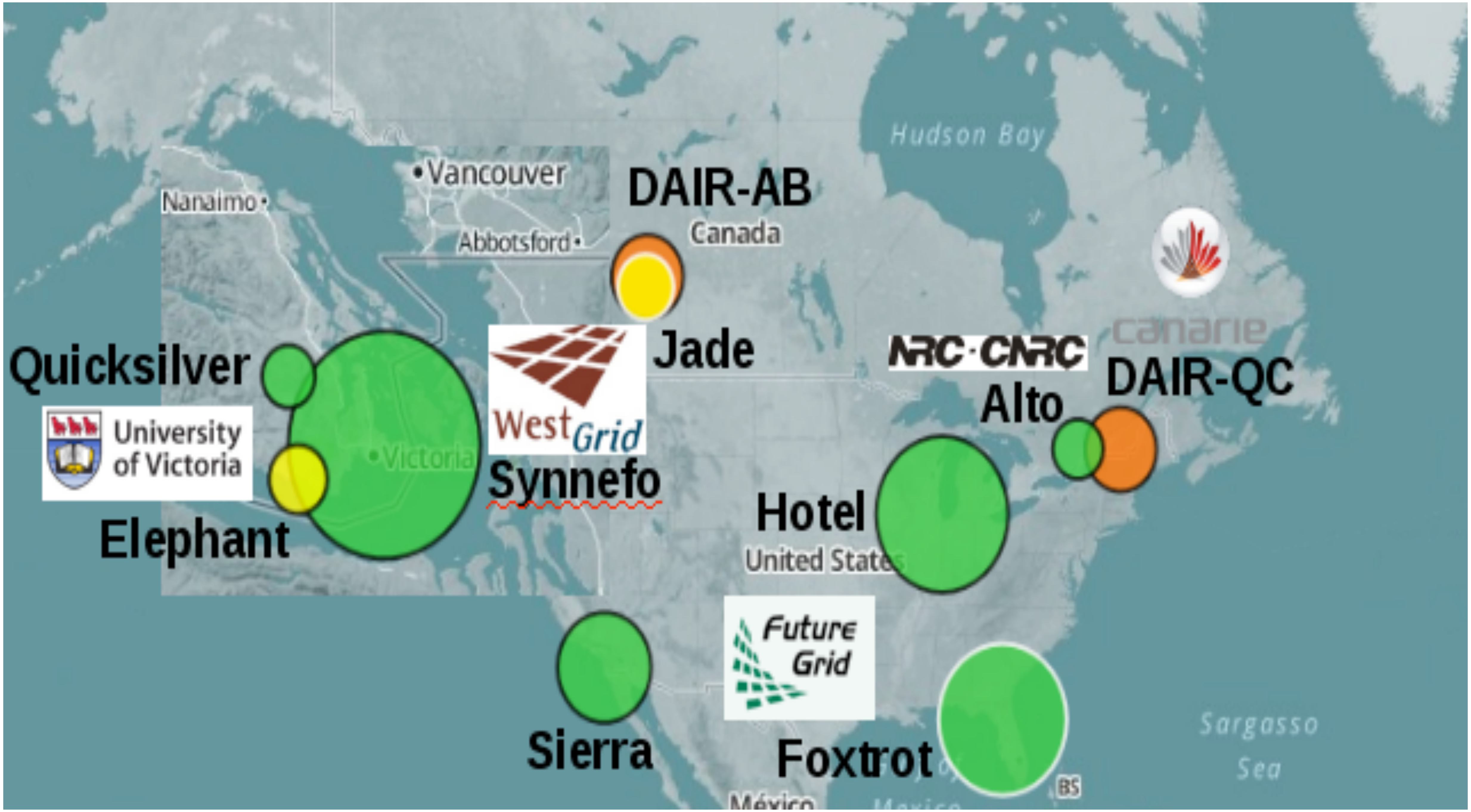
Canada's Advanced Research and Innovation Network
Le réseau évolué de recherche et d'innovation du Canada

canarie

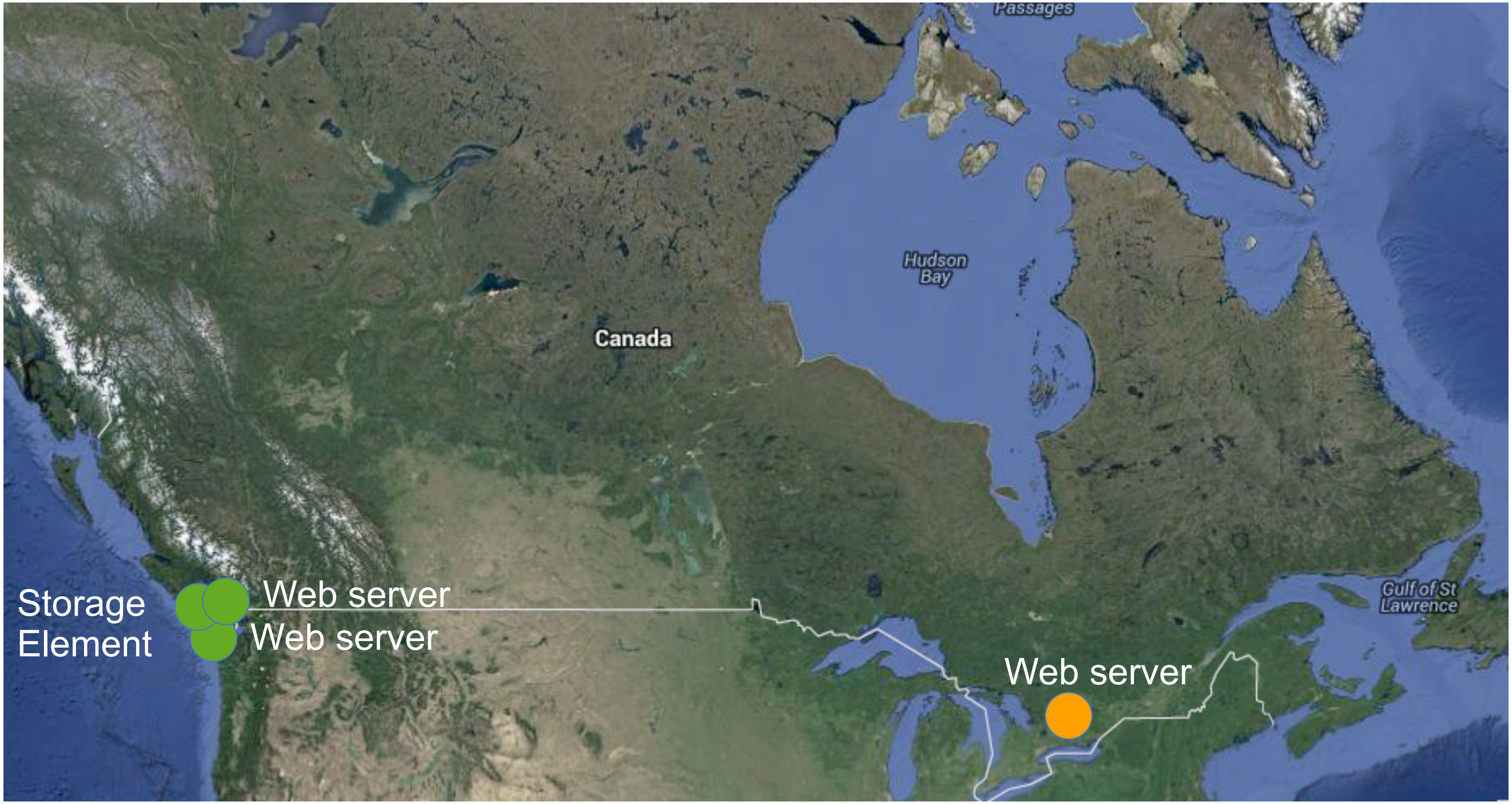
Canada's Advanced Research and Innovation Network
Le réseau évolué de recherche et d'innovation du Canada

- NEP-101 is a project to enable data-intensive applications to run on distributed clouds
  - Batch services, Software distribution, Storage Federation, Image Distribution
- Need to use standard protocols, open-source components, avoid anything HEP-specific
- Have multiple clouds and SEs in various locations; cloud jobs need to find SEs
- Planning to use the Dynamic Federations

North American Clouds

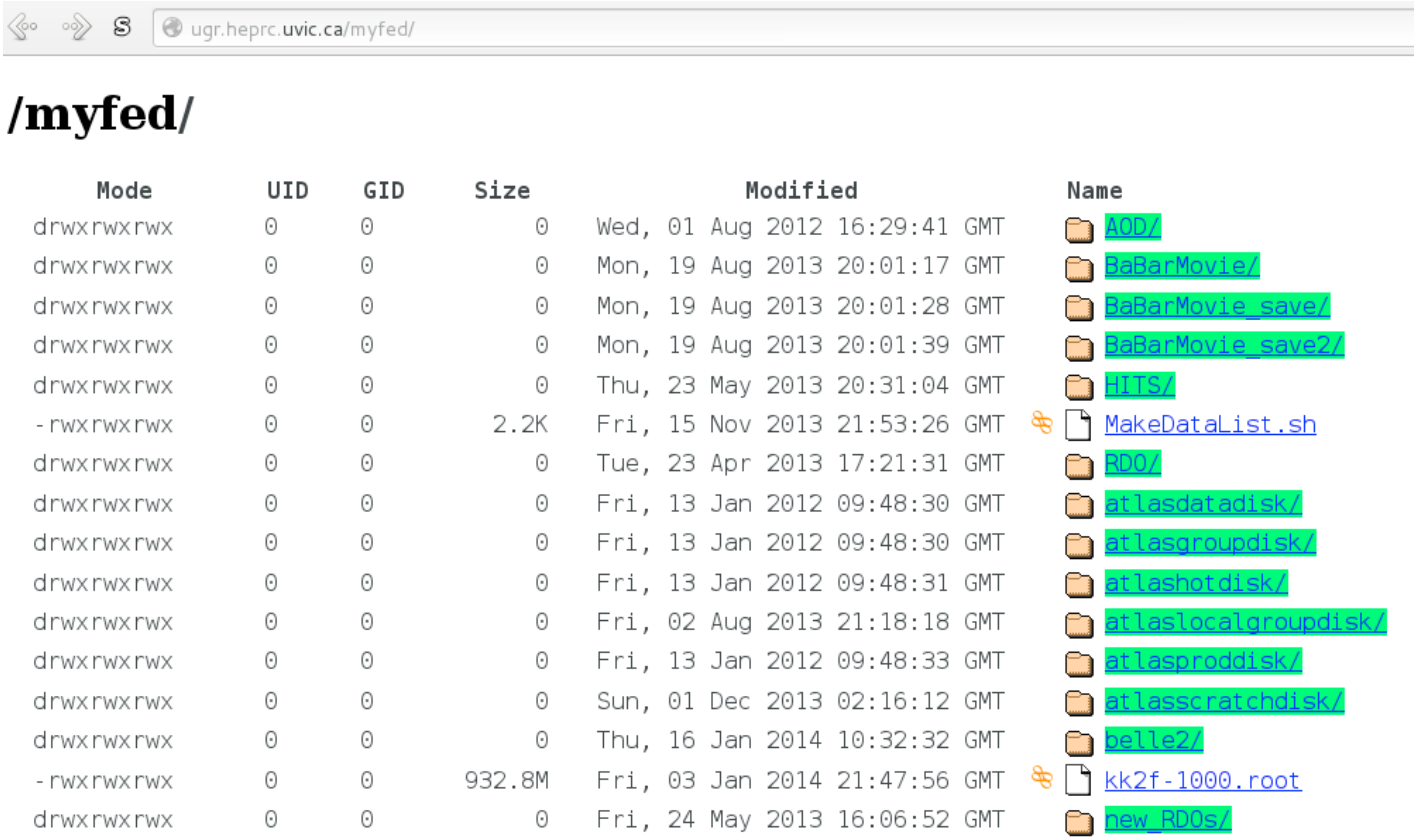


Federation Test Deployment



- More SEs could be added for production deployment

ugr.heprc.uvic.ca/myfed



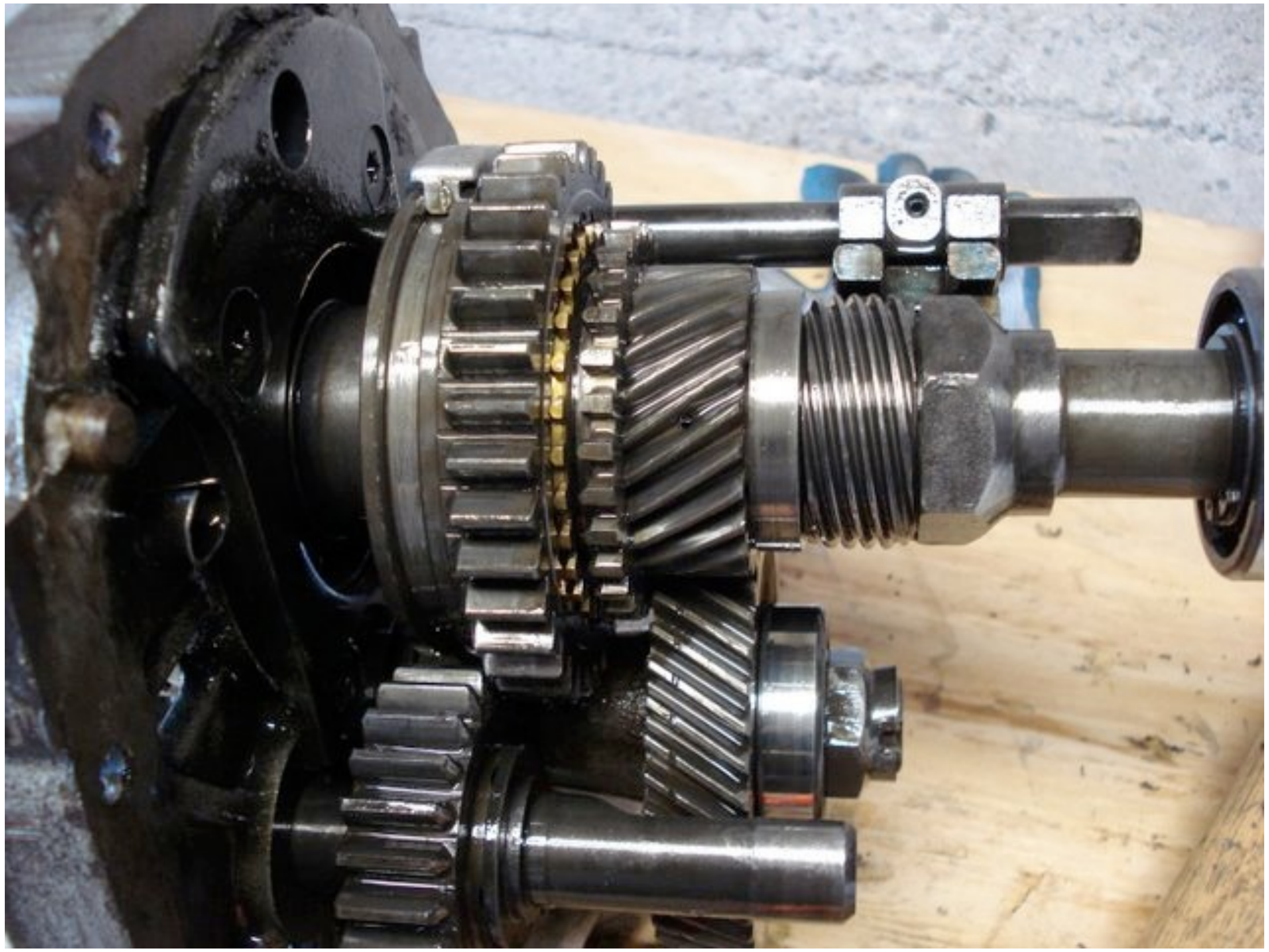
ugr.heprc.uvic.ca/myfed/

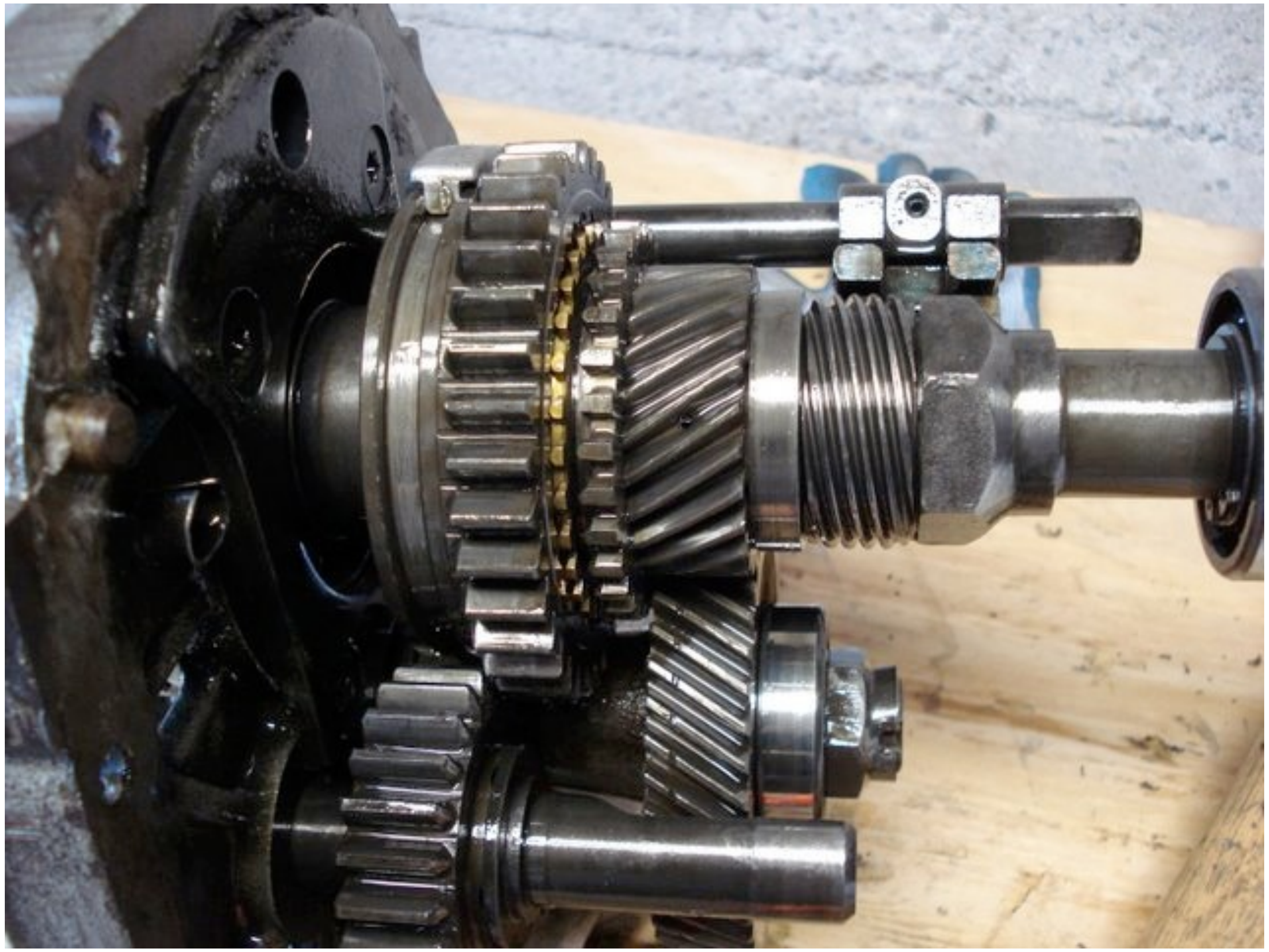
/myfed/

Mode	UID	GID	Size	Modified	Name
drwxrwxrwx	0	0	0	Wed, 01 Aug 2012 16:29:41 GMT	AOD/
drwxrwxrwx	0	0	0	Mon, 19 Aug 2013 20:01:17 GMT	BaBarMovie/
drwxrwxrwx	0	0	0	Mon, 19 Aug 2013 20:01:28 GMT	BaBarMovie_save/
drwxrwxrwx	0	0	0	Mon, 19 Aug 2013 20:01:39 GMT	BaBarMovie_save2/
drwxrwxrwx	0	0	0	Thu, 23 May 2013 20:31:04 GMT	HITS/
-rwxrwxrwx	0	0	2.2K	Fri, 15 Nov 2013 21:53:26 GMT	MakeDataList.sh
drwxrwxrwx	0	0	0	Tue, 23 Apr 2013 17:21:31 GMT	RDO/
drwxrwxrwx	0	0	0	Fri, 13 Jan 2012 09:48:30 GMT	atlasdatadisk/
drwxrwxrwx	0	0	0	Fri, 13 Jan 2012 09:48:30 GMT	atlasgroupdisk/
drwxrwxrwx	0	0	0	Fri, 13 Jan 2012 09:48:31 GMT	atlashotdisk/
drwxrwxrwx	0	0	0	Fri, 02 Aug 2013 21:18:18 GMT	atlaslocalgroupdisk/
drwxrwxrwx	0	0	0	Fri, 13 Jan 2012 09:48:33 GMT	atlasproddisk/
drwxrwxrwx	0	0	0	Sun, 01 Dec 2013 02:16:12 GMT	atlasscratchdisk/
drwxrwxrwx	0	0	0	Thu, 16 Jan 2014 10:32:32 GMT	belle2/
-rwxrwxrwx	0	0	932.8M	Fri, 03 Jan 2014 21:47:56 GMT	kk2f-1000.root
drwxrwxrwx	0	0	0	Fri, 24 May 2013 16:06:52 GMT	new_RDOs/

Federation Deployment Experience

- Easy to set up
  - ~1-2 days to learn, install, configure
- Trivial to add additional storage endpoints
  - Very important for growing the federation
- Software is simple and well-designed
- Next steps
  - Performance tuning
  - Production deployment





The Tech corner

Tech: DynaFeds and Xrootd feds

- Several points in common, with some differences
- DynaFeds is protocol-agnostic, we have used it with an HTTP/DAV frontend
- Replica location:
  - The cmsd (xrootd) clustering is based on location “pauses” (the famous 5 secs per cell)
  - The dynafeds clustering keeps the endpoints under stricter control, so that they can be trusted when they have finished a lookup
    - Result: no 5 secs “pause”
    - Much easier to apply on the fly filters, GeoIP sorting, etc...
  - cmsd privileges sites that answer fast (=are closer) to the redirector. DynaFeds can privilege locations that are closer to the client according to some pluggable metric
- Interactive browsing:
  - The Dynafeds acts as a file listing realtime gatherer and cache
    - Goal: make users comfortable, efficiently feed browsers
  - The cmsd clustering does not gather nor provide listings
    - Different principle: the client has to crawl the whole federation to compose a listing, even of a few files
- A Dynafed can include third-party XrdHttp endpoints and cloud providers
  - Low latency: clustering DAV endpoints works well in both LAN and WAN

Dynamic Federations

- Available in the RC repo of LCGDM
  - https://svnweb.cern.ch/trac/lcgdm/wiki/Dynafeds
- Stable, planning to push it to EPEL
- Technically TODAY we can dynamically aggregate:
  - dCache DAV/HTTP instances
  - DPM DAV/HTTP instances
  - LFC DAV/HTTP and old Cns_* API instances
  - Cloud DAV/HTTP services
  - Anything that can be plugged into DMLite (the new architecture for DPM/LFC)
  - **Can be extended to other metadata sources**
- The system can load a “Geo” filter 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)

Design parameters

- Flexibility: accommodate almost any kind of endpoint (our focus is to HTTP/DAV things)
- High robustness (=correctly treats failures)
- High performance (=many requests per second per frontend)
- Allow interactivity (achievable only with quick systems, suitable protocols, suitable clients)
- Horizontal Scalability (=frontends can be replicated)
- Vertical Scalability (=can have sub-federations)
- Geographical scalability (=one federation can have many frontends in different places)
- Extendability (= easy to add features, intelligence or filters, e.g. geography-related things)

System design

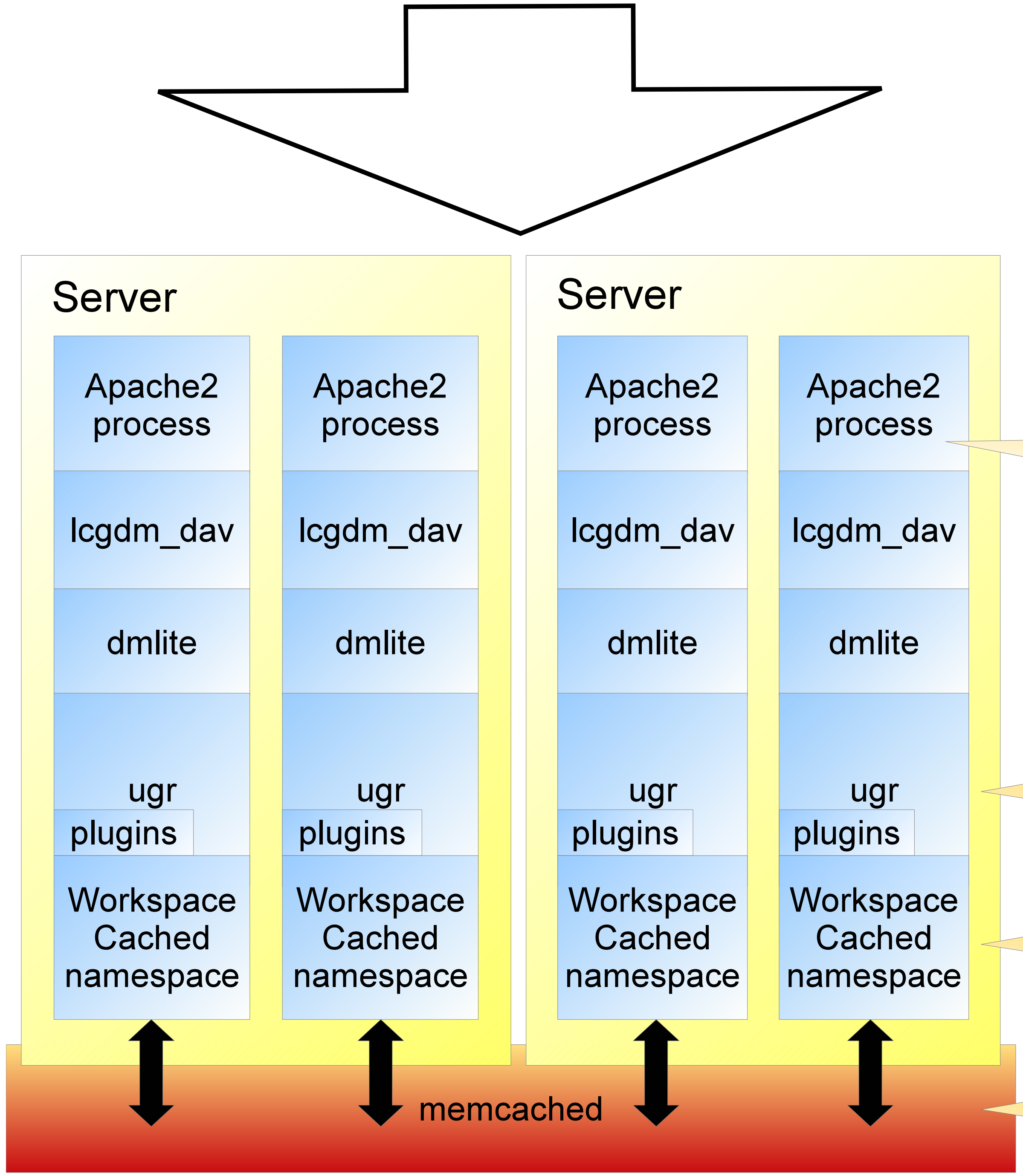
- A system that only works is not sufficient
- To be usable, it must privilege speed, parallelism, scalability
- The core component is a plugin-based component called originally “Uniform Generic Redirector” (Ugr)
  - Can plug into an Apache server thanks to the DMLITE and DAV-DMLITE modules (by IT-GT)
  - Composes on the fly the aggregated metadata views by managing parallel tasks of information location
    - Never stacks up latencies!
  - Makes browsable a sparse collection of file/directory metadata
  - Able to redirect clients to replicas in hosts known to be working in that moment
  - By construction, the responses are a data structure that models a partial, volatile namespace
  - Keep them in an LRU fashion and we have a fast 1st level namespace cache
    - Peak performance is ~500K->1M hits/second per core by now

Focus: performance

- Performance and scalability have primary importance
  - Otherwise it's useless…
- Full parallelism
  - 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
- Spurred a high performance DAV client implementation (DAVIX)
  - Wraps DAV calls into a POSIX-like API, saves from the difficulty of composing requests/ responses
  - Loaded by the core as a "location" plugin
  - http://dmc.web.cern.ch/projects/davix/home
  - Available in ROOT 5 and 6 as TDavixFile

Clients come and are distributed through:
different machines (DNS alias)
different processes (Apache config)

Clients are served by the UGR. They can browse/stat or be redirected for action. The architecture is multi/manycore friendly and uses a fast parallel caching scheme



Server

Apache2 process

lcgdm_dav

dmlite

ugr

plugins

Workspace Cached namespace

Server

Apache2 process

lcgdm_dav

dmlite

ugr

plugins

Workspace Cached namespace

memcached

Apache can spawn several processes per machine
Each process has many worker threads
We may need several frontend machines

UGR and the plugins are multithreaded
Every process has its own fast workspace

Each workspace works as a local, fast 1st level cache

Memcached acts as 2nd level cache
Synchronizes the private cached workspaces

A WAN metadata performance test

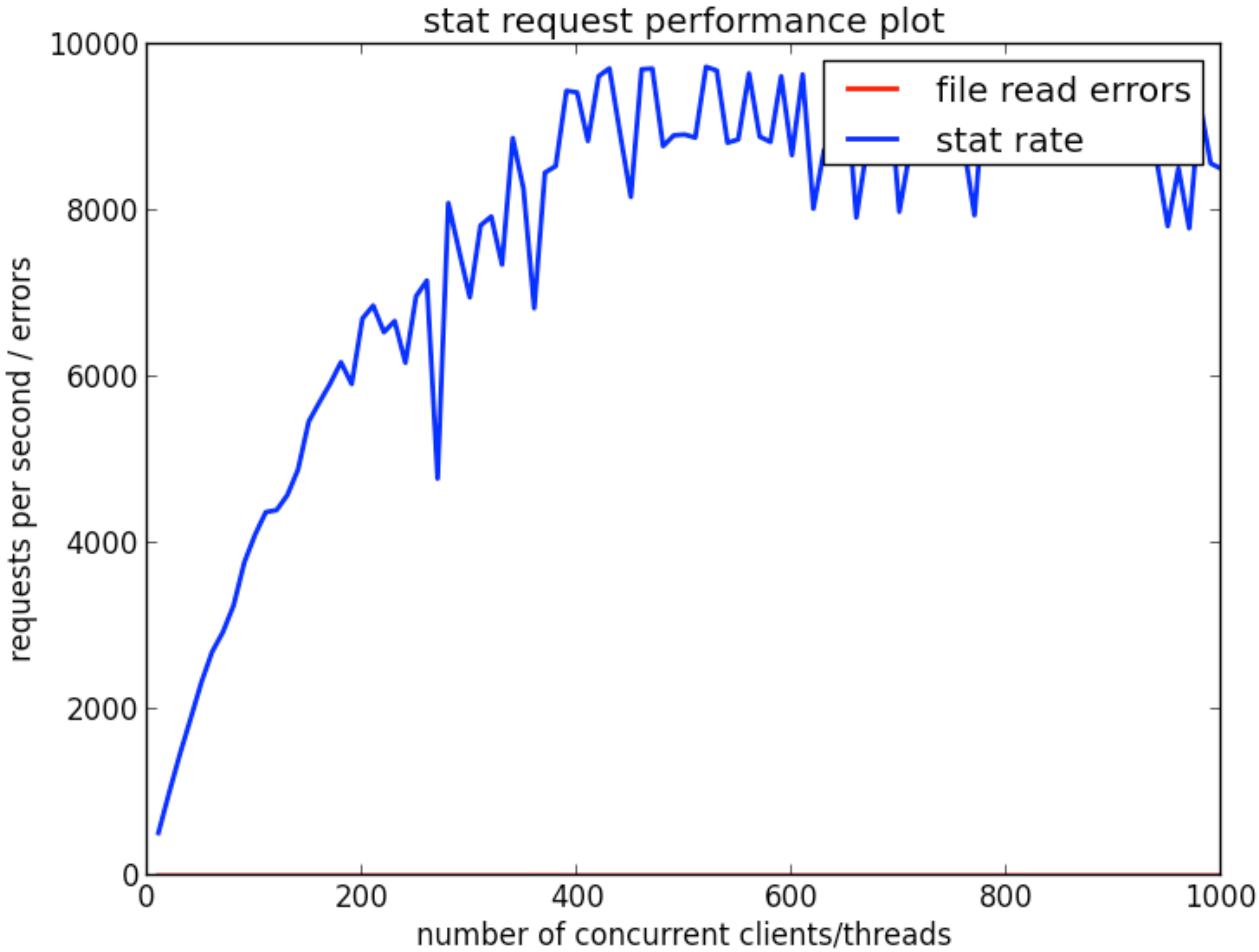
- We wanted to see if we can blast our demo of requests, the results are very interesting

- One UGR federator at DESY, clients at CERN
  - The federator has 10-12 endpoints of various kinds

- 10K files are interleaved in a 4-levels deep directory
  - Oddly-numbered files are at CERN
  - Evenly-numbered files are at Desy

- The test (written in C++) invokes Stat once per file, using many parallel clients doing stat() at the maximum pace

WAN access Stat() test (CERN->DESY)



Federated security

- The dynafeds are agnostic to security
  - The federator never writes to the endpoints
  - The federator never sees data, only metadata!
- The dynafeds are a powerful data location and browsing machinery
  - Supports X509, VOMS, etc.
- Unique sec requirement: The federator user only needs read access to metadata
- Harmonization of the security aspects of a worldwide WebDAV/Cloud deployment is one of the objectives of the Identity Federations working groups
  - Joining these efforts has enormous potential
  - The effort goes more to planning things using macro building blocks

Status

- Very stable, installable from the wiki
- Survived very well any stress test we could do, also federating disk caches in LAN
- LAN tests showed no worst-case performance difference with ICMP (Squid), better performance in the best case (cached metadata, which squid can't do)
- External demo in http://federation.desy.de/

- Next stop: EPEL
- Next stop: Drupal (Google-reachable documentation)
- Next stop: ATLAS and Rucio
  - We have a nice testbed, federating many ATLAS SEs
  - We want to federate the Rucio services and the LFC(s) seamlessly together
  - Just needs to parse the JSON produced by Rucio… technically not a big deal
- Power users wanted
  - Helping in getting the best out of the system. Your cooperation and ideas are very appreciated.

Conclusions

- Dynamic Federations: an efficient, persistency-free, scalable, easily manageable approach to federate remote storage and metadata endpoints
- Usable for fast changing caches and clouds
- Gives ways to solve some nasty Data Management problems
- Made to work with generic, standards-based components. OK for HEP and other domains as well
- Can coexist and complement Xrootd federations
  - See the presentation on XrdHTTP
- Status is stable, demoable, installable, documented
- https://svnweb.cern.ch/trac/lcgdm/wiki/Dynafeds
- http://federation.desy.de/