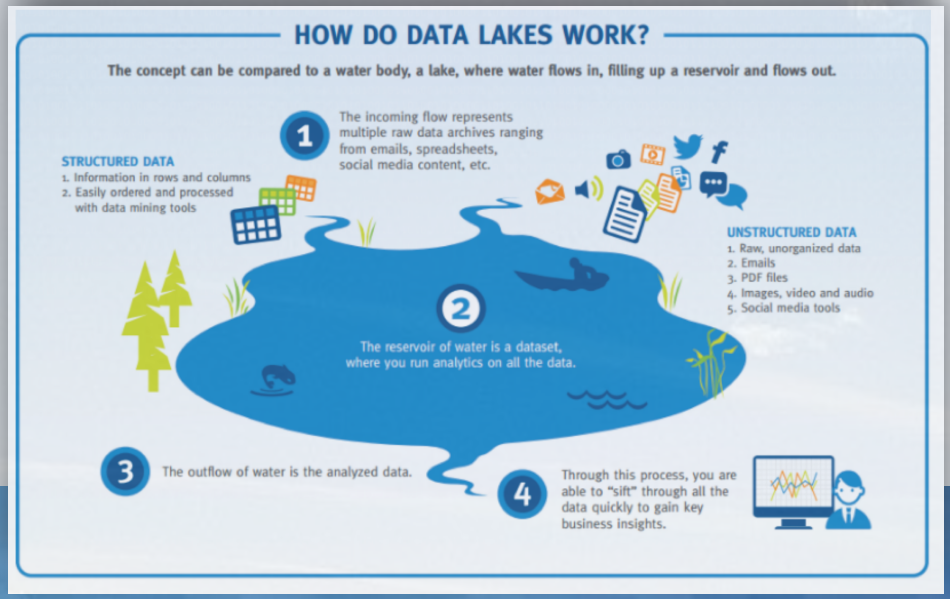


andreas.Joachim.peters@cern.ch

EOS dev for XDC & Data Lake Support

Qu'est-ce que le Data Lake, le nouveau concept "Big Data" en vogue
www.journaldunet.com › Web & Tech › DSI ▾ Diese Seite übersetzen
09.01.2017 - Le **Data Lake** doit permettre, enfin, de casser les silos des systèmes d'information. C'est aussi un moyen de gagner en agilité. L'expert Vincent Heuschling répond aux questions du JDN.

Andreas-Joachim Peters
CERN - IT
Storage Group

Data lake

From Wikipedia, the free encyclopedia

A **data lake** is a method of storing data within a system or repository, in its natural format,[1] that facilitates the collocation of data in various schemata and structural forms, usually object blobs or files. The idea of data lake is to have a single store of all data in the enterprise ranging from raw data (which implies exact copy of source system data) to transformed data which is used for various tasks including reporting, visualization, analytics and machine learning. The data lake includes structured data from relational databases (rows and columns), semi-structured data (CSV, logs, XML, JSON), unstructured data (emails, documents, PDFs) and even binary data (images, audio, video) thus creating a centralized data store accommodating all forms of data.[2]

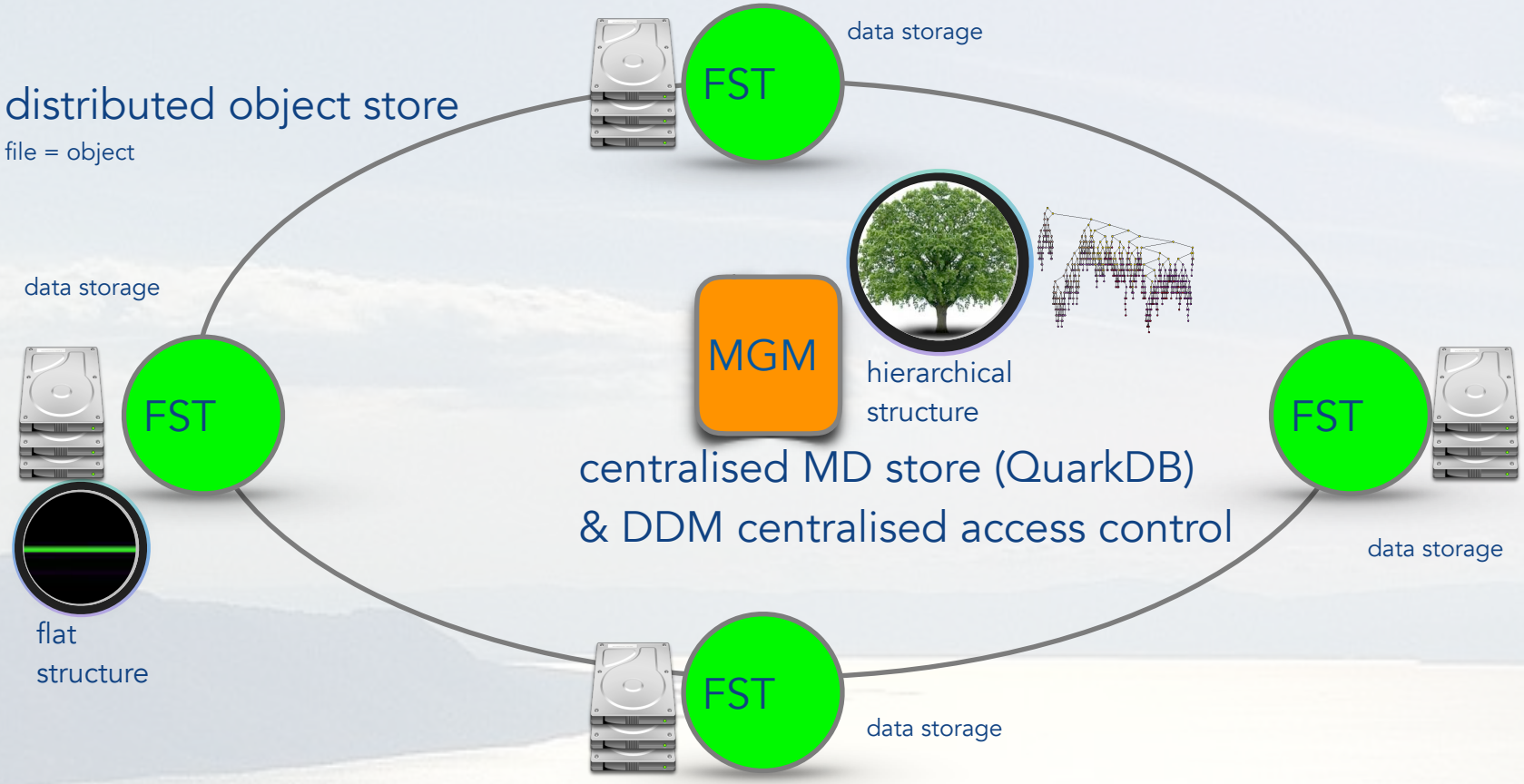
A **data swamp** is a deteriorated data lake, that is inaccessible to its intended users and provides little value.[3][4]

Overview

- **XDC/EOS Key components**
  1. Storage Workflows
  2. Storage Adaptor
  3. Managed Caches

- **EOS Development Items to operate a Data Lake**
  - External Storage Mounts & Synchronisation
  - Extension of File Layout Concepts
    - complex layouts
    - exposed locations
  - Internal & External Workflows
  - File Layout, Distribution & Lifecycle Policies

EOS - Distributed Architecture



EOS - External Storage Mounts

distributed object store

file = object

data storage

MGM

hierarchical structure

centralised MD store (QuarkDB)

& DDM centralised access control

mounted external storage with external namespace

data storage

data storage

flat structure

data storage

basic constraints:
write-once data
PUT semantic

EOS - External Storage Mounts

Current closed model

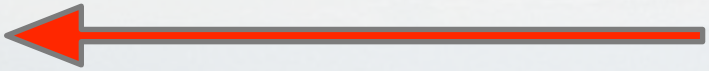
LFN: /eos/public/myfile => inode X @ storage Y

global logical namespace
creates flat storage namespace

External model

LFN: /eos/amazon/myfile <= LFN:/bucket/myfile

LDN: /eos/amazon mounts s3://bucket/

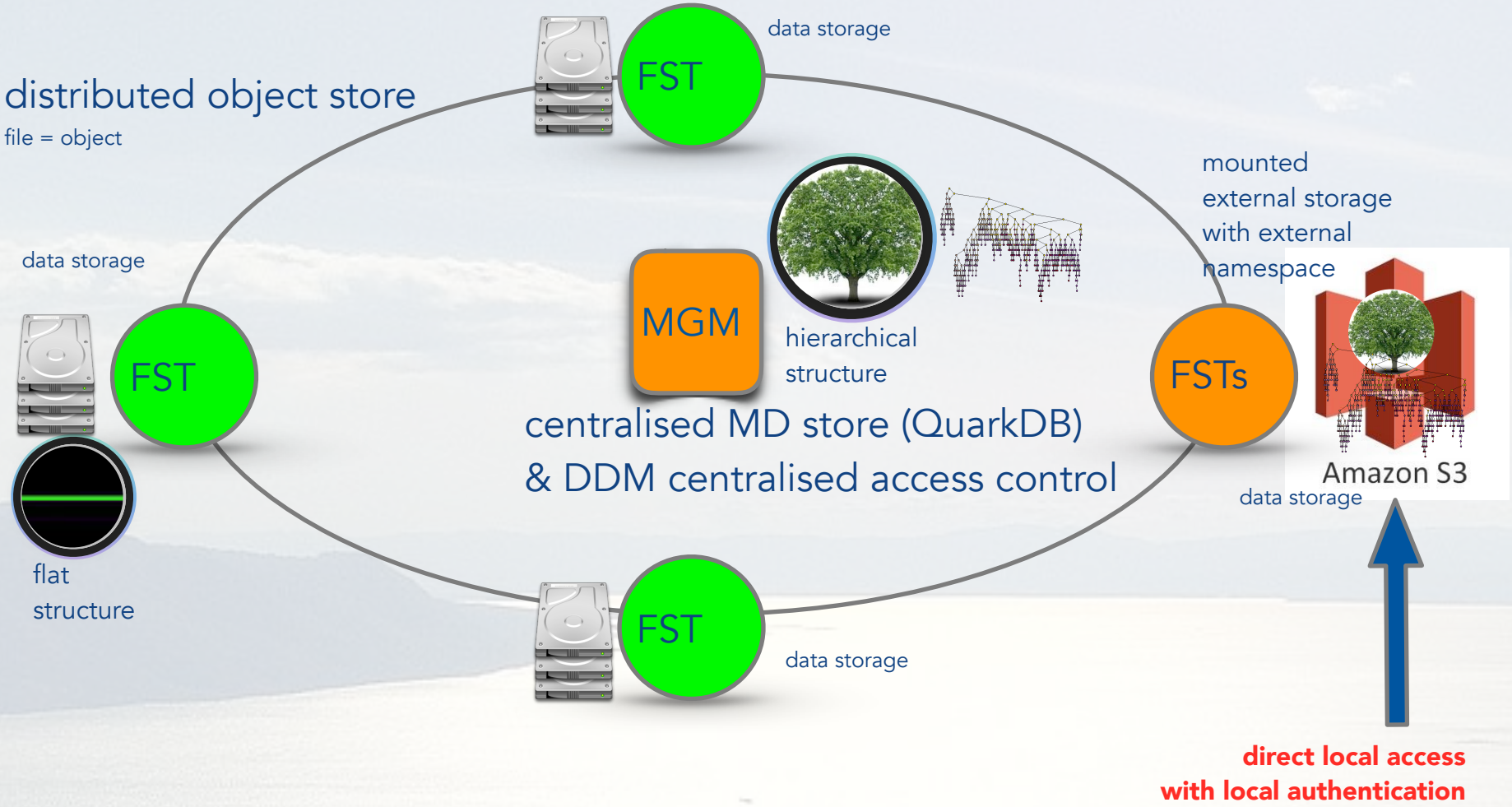


local external (flat or hierarchical) storage namespace synchronises into global namespace with predefined path and ownership mapping

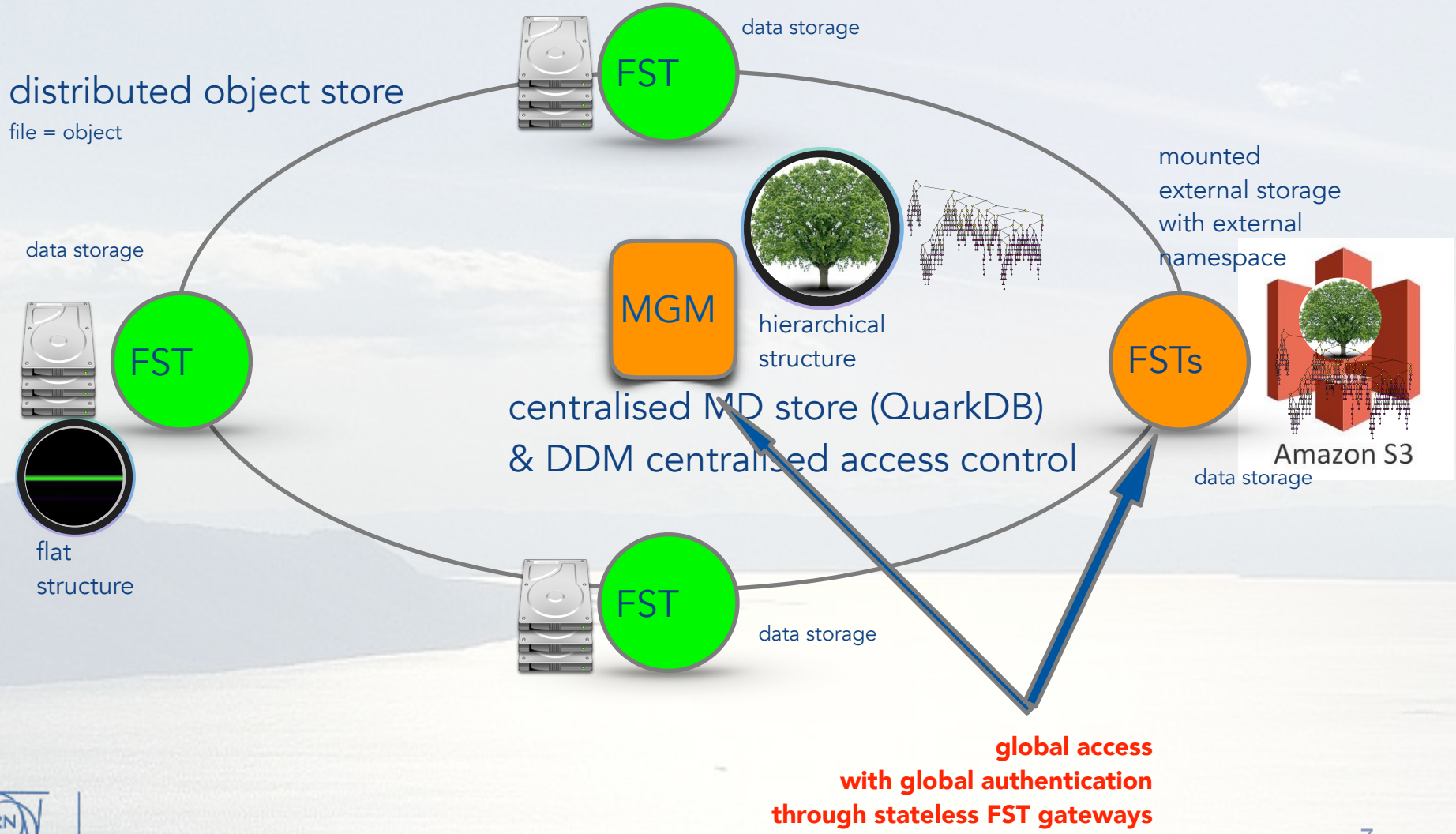
change discovery mechanisms:

- using scans
- using notification (inotify, AWS notifications)

EOS - External Storage Mounts



EOS - External Storage Mounts



EOS - Extended File Layouts

Today files described by static layout (type + parameters e.g replica:2)

```
EOS Console [root://localhost] |/eos/pps/users/apeters/> file info myfile
  File: '/eos/pps/users/apeters/myfile'  Flags: 0640
  Size: 1431
Modify: Mon Dec 18 23:28:52 2017 Timestamp: 1513636132.0
Change: Mon Dec 18 23:28:52 2017 Timestamp: 1513636132.336292718
  CUid: 0 CGid: 0  Fxid: 0bbcabae Fid: 196914094    Pid: 146768814   Pxid: 08bf83ae
XStype: adler    XS: 05 a7 f1 40     ETAG: 52858724615716864:05a7f140
replica Stripes: 2 Blocksize: 4k LayoutId: 00600112
  #Rep: 2
```

no.	fs-id	host	schedgroup	path	boot	configstatus	drainstatus	active	geotag
0	6783	p05614923d80639.cern.ch	default.33	/data39	booted	rw	nodrain	online	9918::R::0001::WB02
1	8345	lxfsre03a04.cern.ch	default.33	/data05	booted	rw	nodrain	online	0513::R::0050::RE03

New namespace backend (QuarkDB) allows to store additional meta data per file:

- extend the concept of layouts by distinguishing a **static** and a **dynamic** part
  - static part allows to guarantee longterm durability
  - dynamic part allows to track locations in caches, might be stale

static:

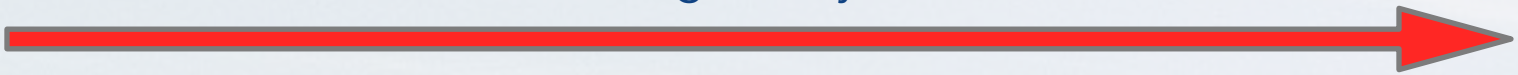
no.	fs-id	host	schedgroup	path	boot	configstatus	drainstatus	active	geotag
0	6783	p05614923d80639.cern.ch	default.33	/data39	booted	rw	nodrain	online	9918::R::0001::WB02
1	8345	lxfsre03a04.cern.ch	default.33	/data05	booted	rw	nodrain	online	0513::R::0050::RE03

dynamic:

2	8400	bucket.aws1.fzk	aws	/aws.fzk	booted	rw	nodrain	online	AWS::DE::FZK
3	8401	bucket.aws2.fzk	aws	/aws.muc	booted	rw	nodrain	online	AWS::DE::MUC

EOS - Layouts Life cycles

automatic change of layouts over time [how, when, where]



	on creation	**after 1 month**	**after 3 month**	**after 6 month**
	replica:3 + dyn. caching	RAIN: (4,2) no dyn. caching	replica: 1 + 1 tape copy	1 tape copy
on disk	300% + dyn.	150%	100%	0%
on tape	0%	0%	100%	100%

need to extend language to express layout life cycles in extended attributes

CERN

EOS - Location Exposure

- today all files are 'located' at the namespace node
- need to integrate virtual location lookup with job scheduling system to optimise cpu/disk proximity
  - XRootD **location query**
  - agree on using **metalinks** (?) created via smart files*

* smart files are virtual files creating contents on the fly by executing an EOS command as implemented for WLCG storage monitoring/description files

EOS - Internal workflows

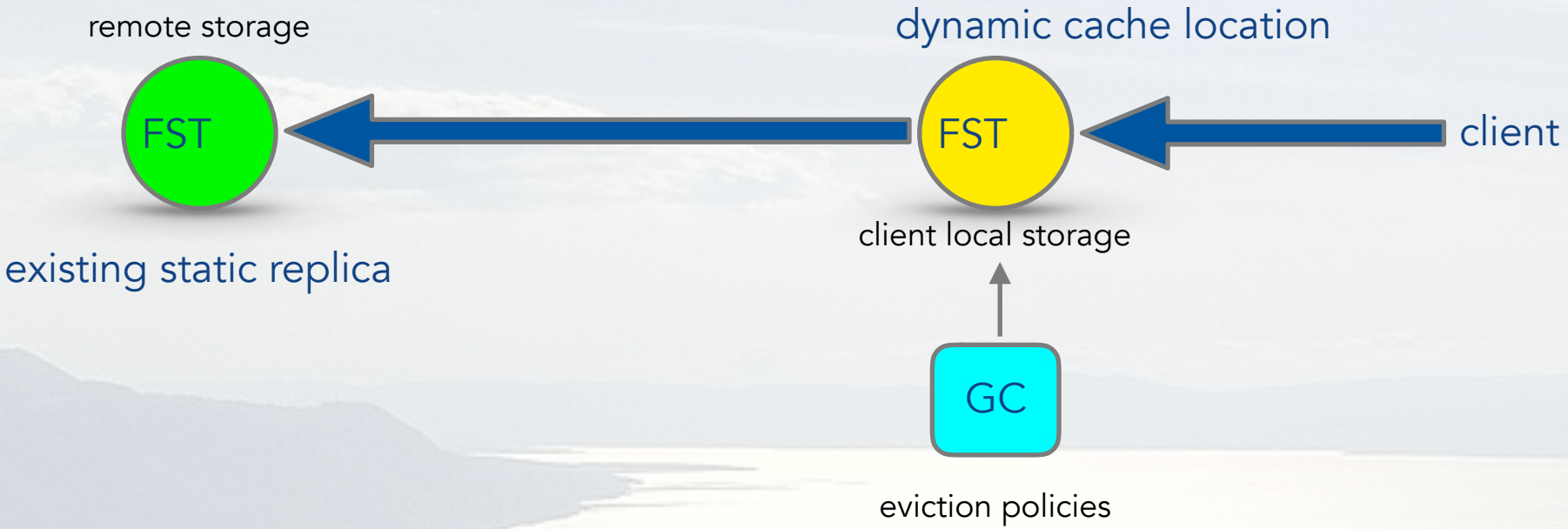
- **core component** developed for **tiered architecture** (EOS+CTA)
  - steer migration & recall

- **core component** for automatic **layout lifecycle**
  - **time based** scheduling (avoids full table scans)
  - workflow **chains**
    - schedule workflow B when A was executed
      e.g. when a file was converted after 1 week to a RAIN file
      we schedule after 6 month to migrate to tape only

EOS - External workflows

- workflow implementation is changed now to send **protobuf** messages to arbitrary **external services** e.g.
  - will feed CTA service
  - can feed FTS service
  - can feed POPularity service to manually trigger layout lifecycles
  - trigger automatic job submission when files are generated
  - trigger automatic file registration in experiment databases
  - aso.

EOS - FST read-through cache &GC



no passive untracked caches inside EOS