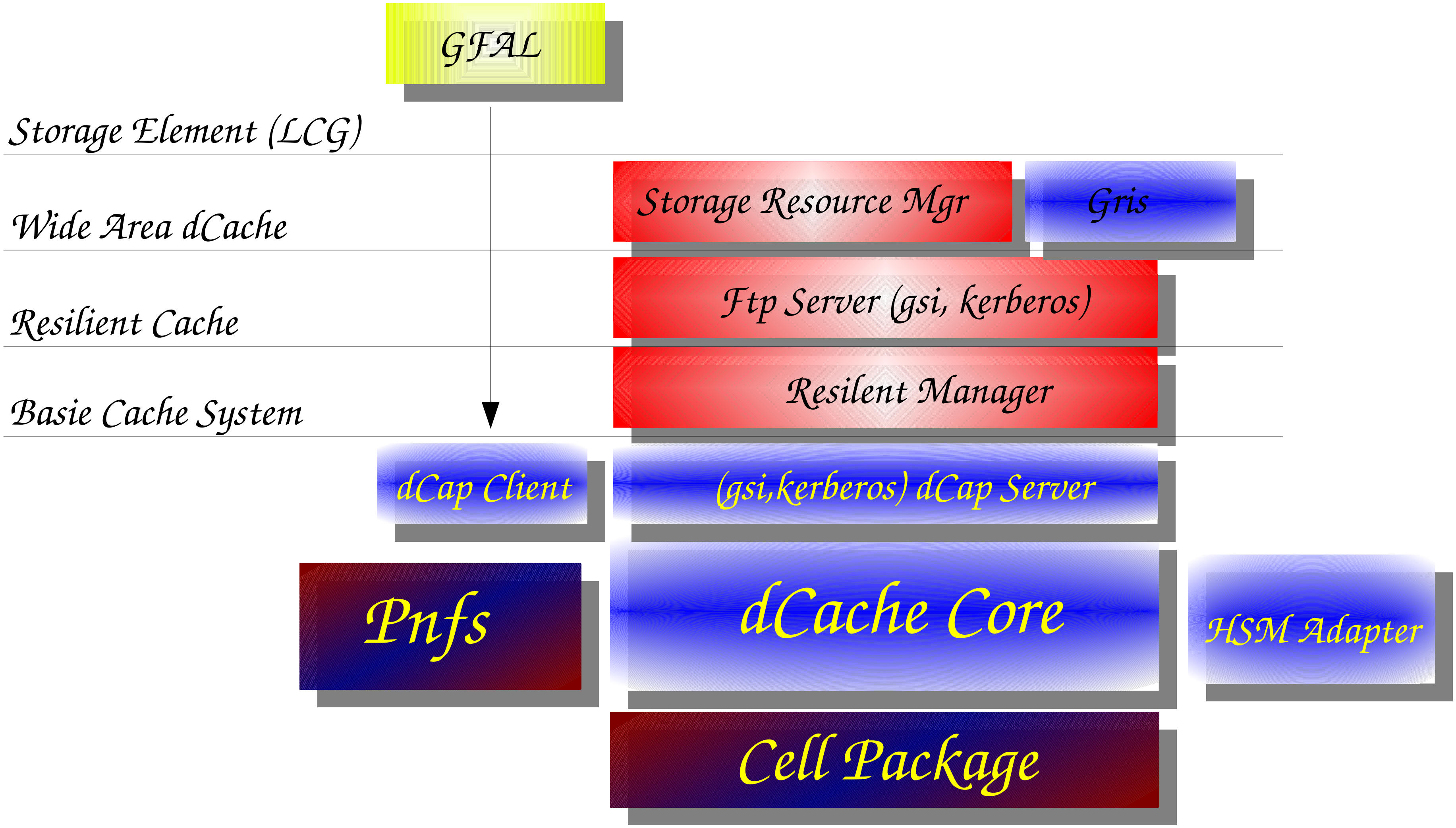
Insight dCache

Patrick Fuhrmann, DESY

for the dCache Team

dCache is a joint effort between the Deutsches
Elektronen Synchrotron (DESY)
and the Fermi National Laboratory (FNAL)

dCache Functionality Layers



GFAL

Storage Element (LCG)

Wide Area dCache

Resilient Cache

Basie Cache System

Storage Resource Mgr

Gris

Ftp Server (gsi, kerberos)

Resilent Manager

dCap Client

(gsi,kerberos) dCap Server

Pnfs

dCache Core

HSM Adapter

Cell Package

dCache Functionality Layers

Basic dCache System

- Single 'rooted' file system name space tree
- Data may be distributed among a huge amount of disk servers.
- Supports multiple internal and external copies of a single file
- Automatic load balancing by cost metric and interpool transfers.
- Data removed only if space is needed
- Distributed Access Points (Doors)
- Using standard 'ssh' protocol for administration interface.

dCache Functionality Layers

Basic dCache System (cont.)

- Fine grained configuration of pool attraction scheme
- Pool to pool transfers on config. or data access hot spot detection
- CRC checksum calculation and comparison (partially implemented yet)
- Pluggable door / mover pairs
- Automatic HSM migration and restore
- Convenient HSM connectivity (done for enstore,osm,tsm, bad for Hpss)

dCache Functionality Layers

dCap Protocol and Implementation

- implements I/O and name space operations including 'readdir'
- available as standard shared object and preload library

  ```
  ls -l dcap://dcachedoor.desy.de/user/patrick
  ```

- ROOT has interface to dCache
- positive tested for Linux, Solaris, Irix (untested for XP)
- automatic reconnect on server door and pool failures
- supports read ahead buffering and deferred write
- supports ssl, kerberos and gsi security mechanisms
- Thread safe

dCache Functionality Layers

Resilient dCache

- Controls number of copies for each dCache dataset
- Makes sure n < copies < m
- Adjusts replica count after pool failures
- Adjusts replica count on scheduled pool maintenance

Wide Area dCache

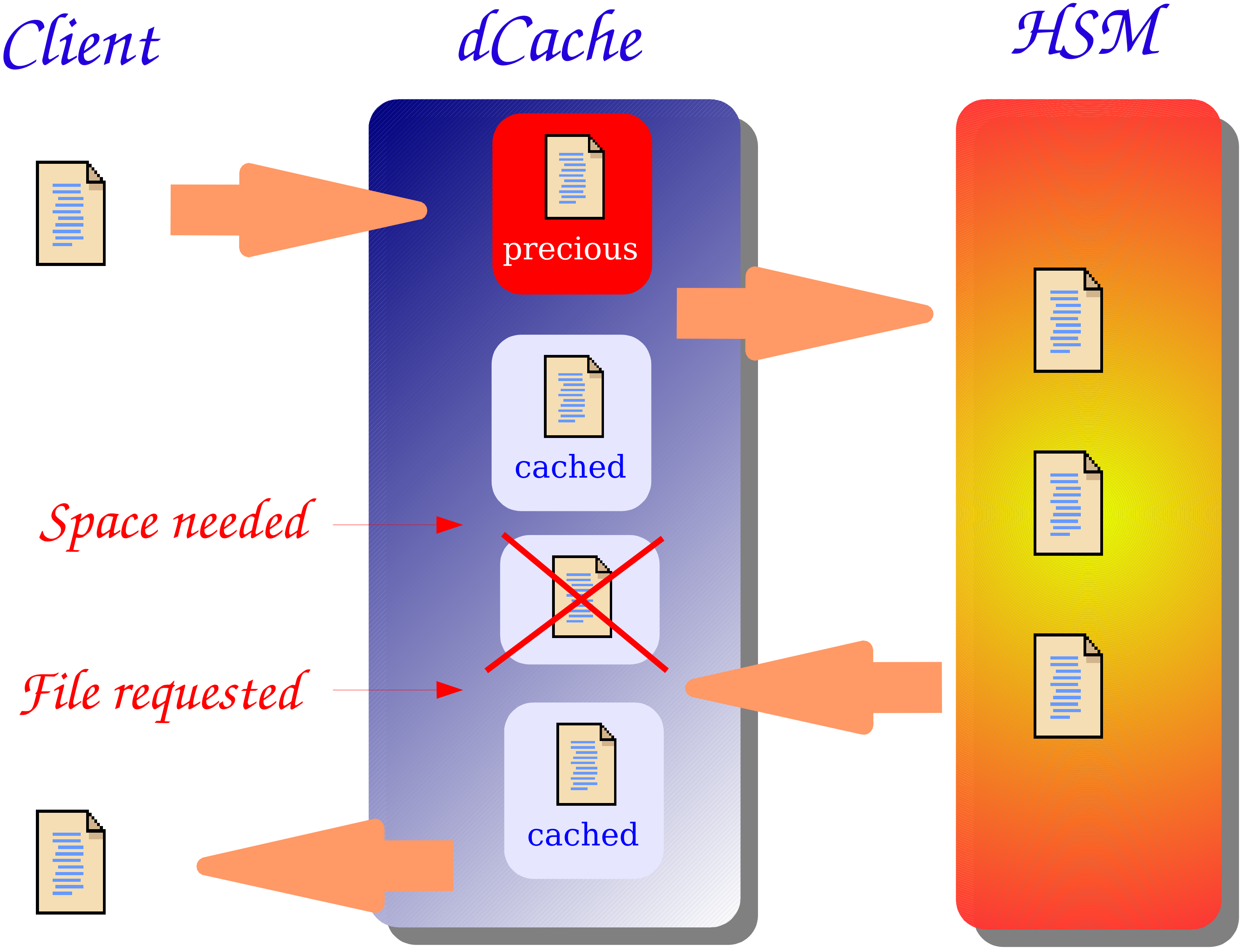
- Support of Kerberos and Gsi FTP
  - Support of Http and Https

LCG Storage Element

- DESY dCap lib incorporates with CERN GFAL library
- gsiFtp supported
- SRM version ~ 1 (1.7) supported
- no buildin GRIS functionality available yet (but workarounds)

dCache
The HSM Interface

dCache - HSM Interactions



Client

dCache

HSM

precious

cached

Space needed

File requested

cached

Patrick Fuhrmann, Insight dCache

dCache - HSM Interactions : deferred HSM flush

Precious data is separately collected per storage class

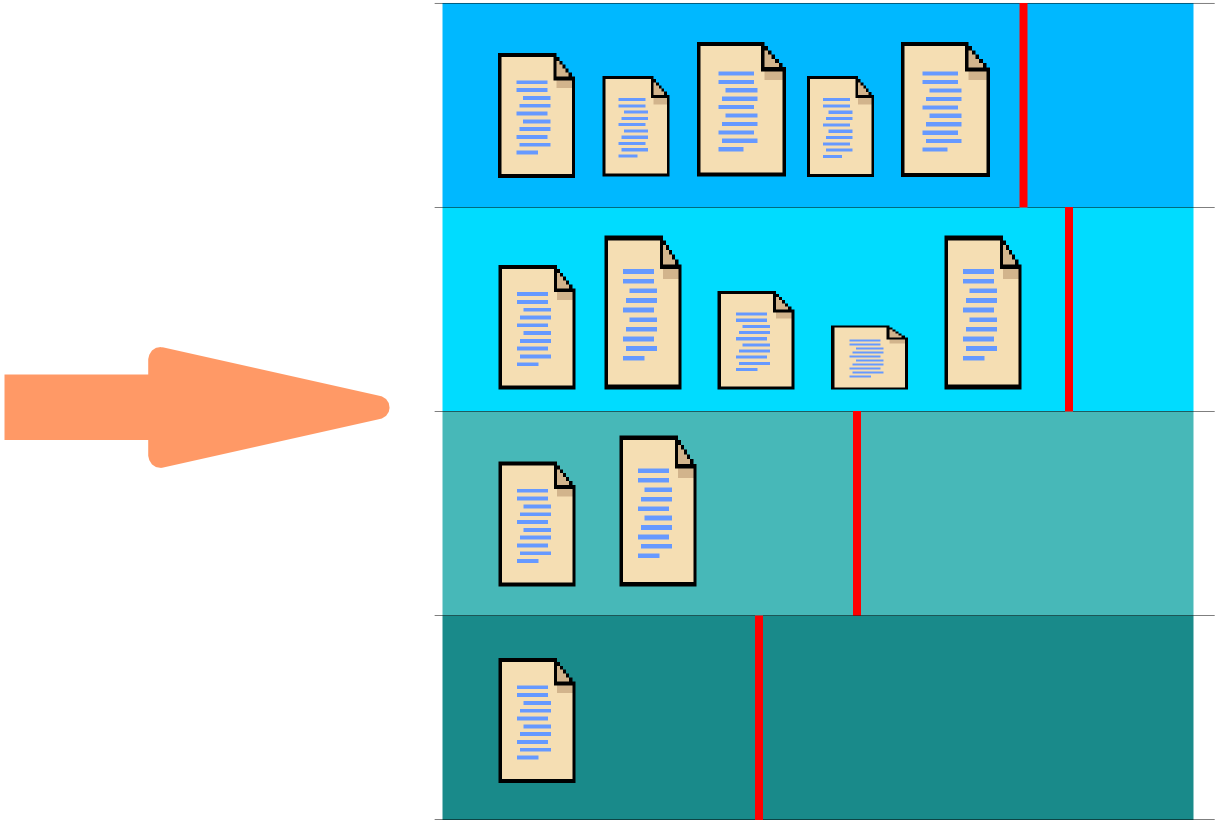
Each 'storage class queue' has individual parameters, steering the HSM flush operation.

- Maximum time, a file is allowed to be 'precious' per 'storage class'.
- Maximum number of precious bytes per 'storage class'
- Maximum number of precious files per 'storage class'

Maximum number of simultaneous 'HSM flush ' operations can be configured

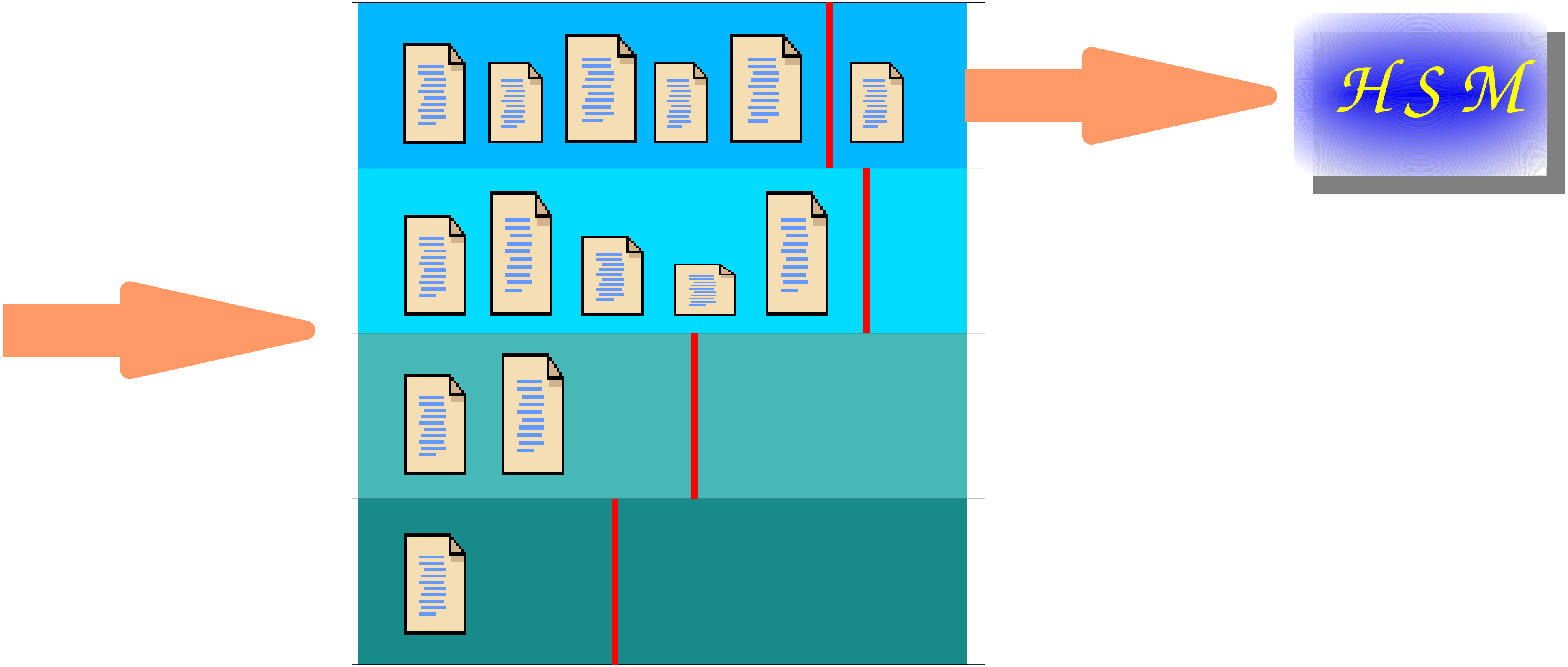
Multiple (even different) HSMs are supported.

dCache - HSM Interactions : deferred HSM flush



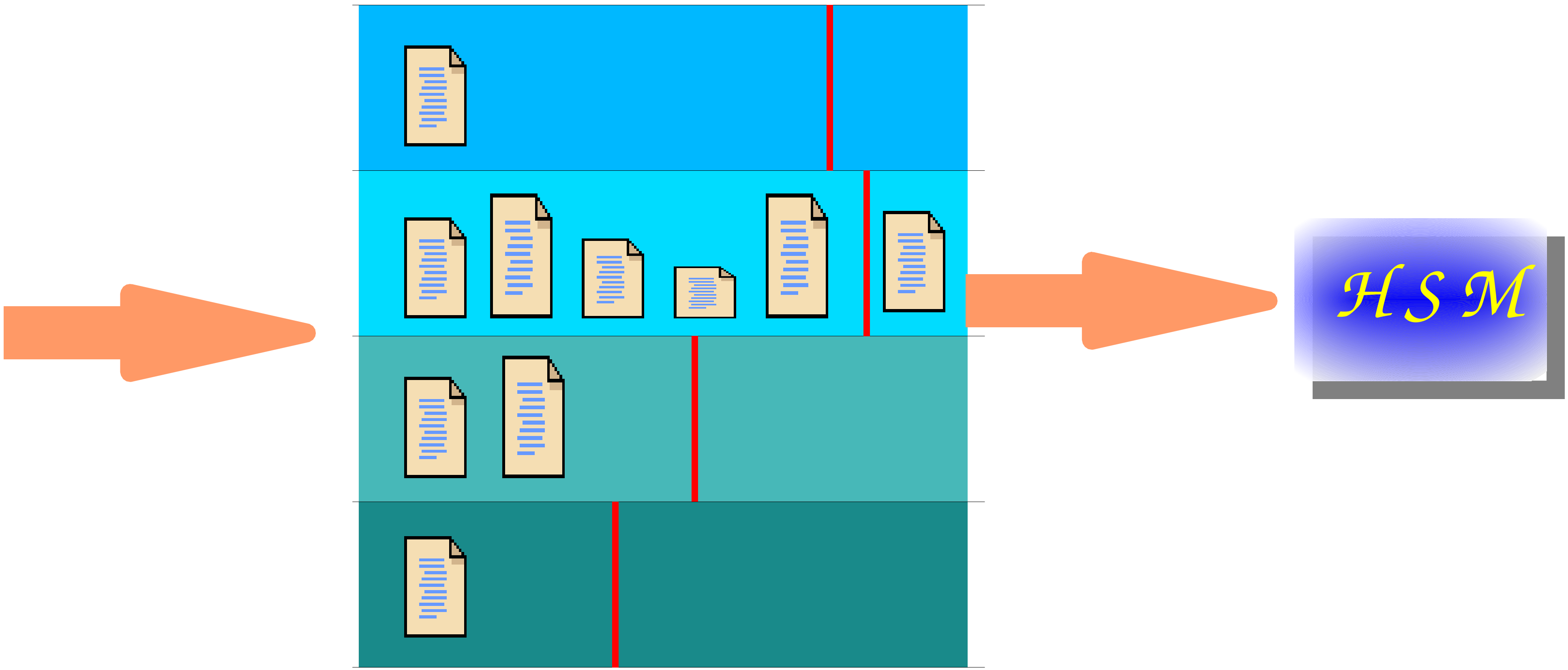
Incoming data sorted according to 'storage groups'

dCache - HSM Interactions : deferred HSM flush

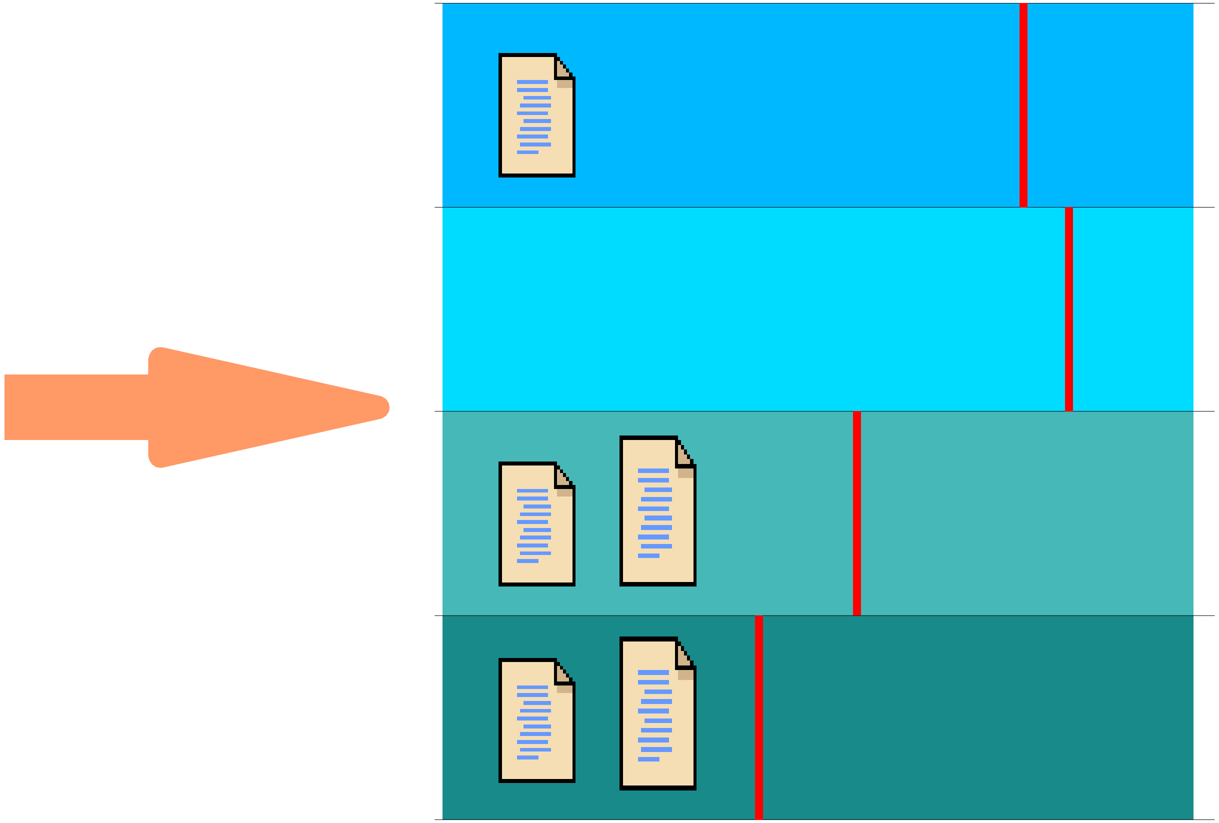


Incoming data sorted according to 'storage groups'
and flushed if individual queue limit reached.

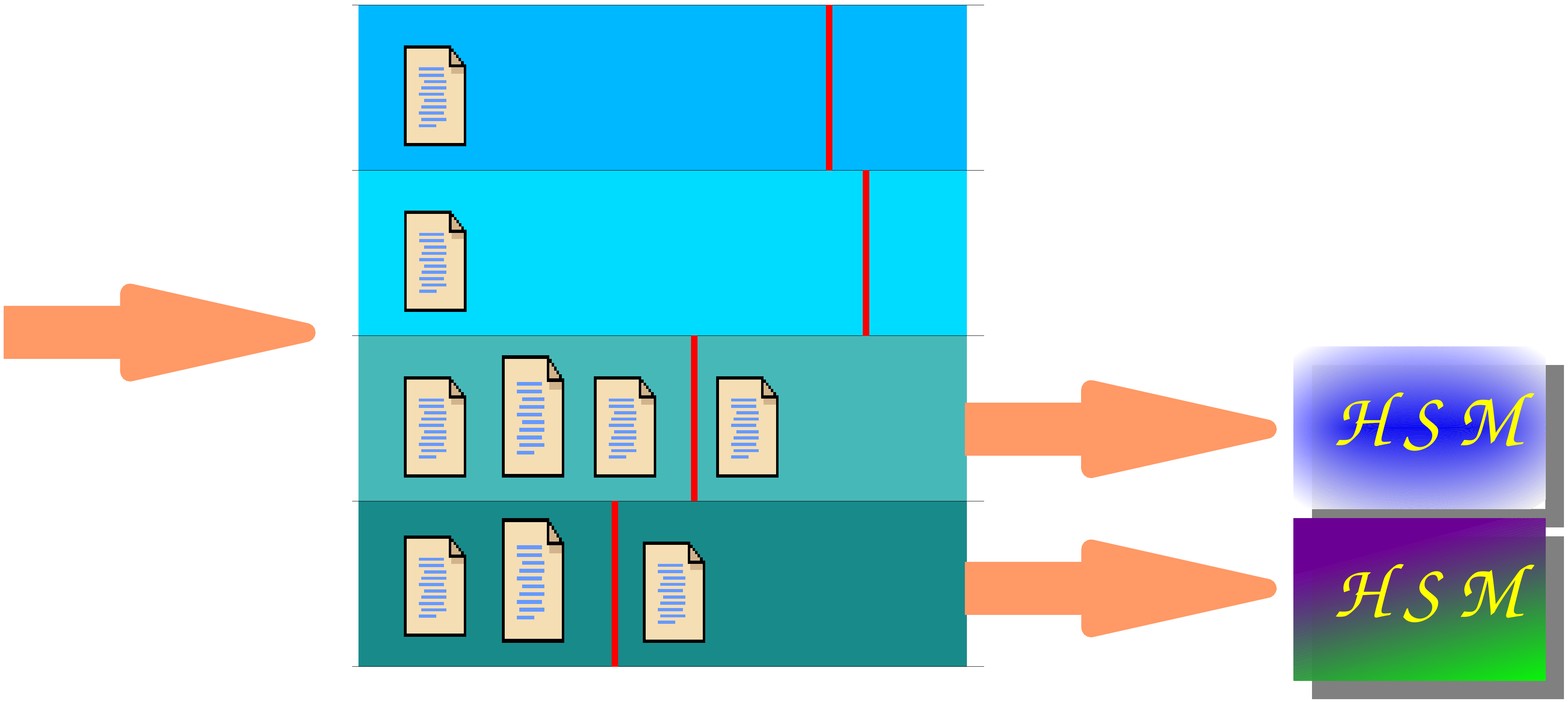
dCache - HSM Interactions : deferred HSM flush



dCache - HSM Interactions : deferred HSM flush

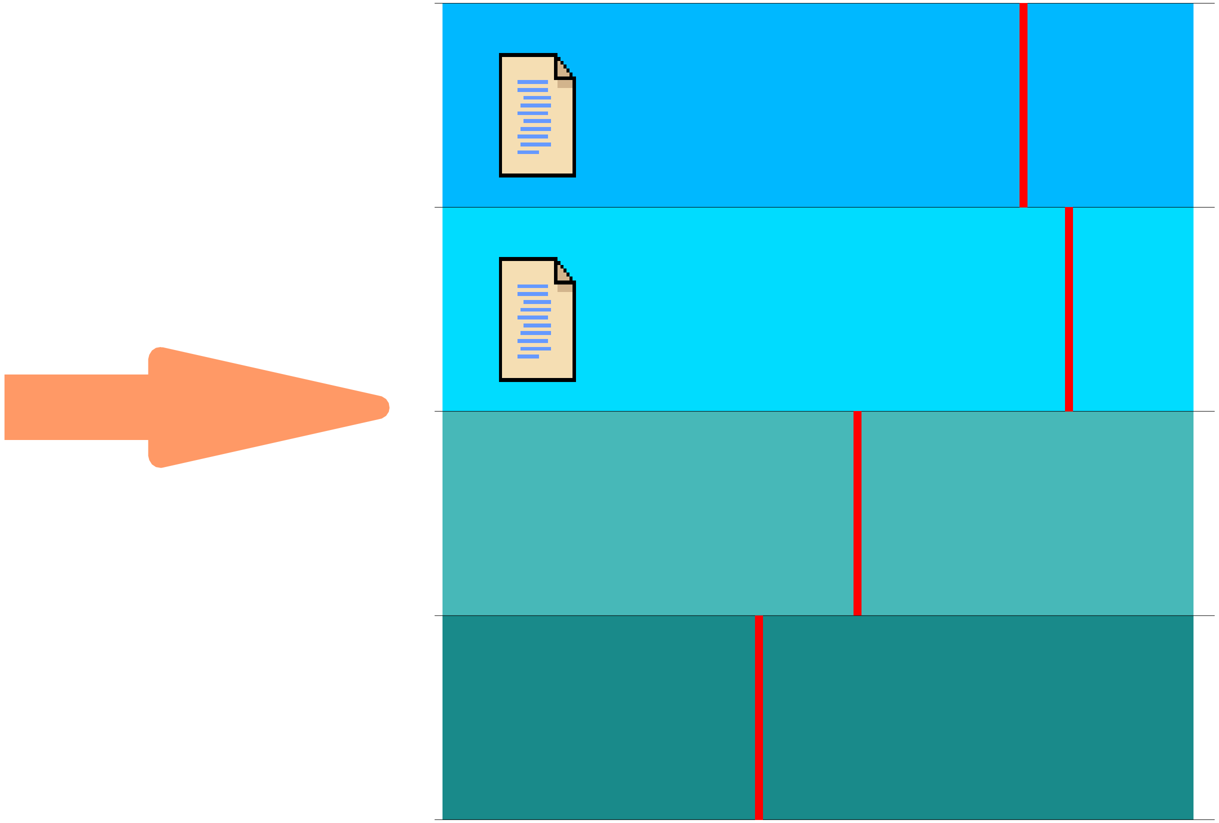


dCache - HSM Interactions : deferred HSM flush

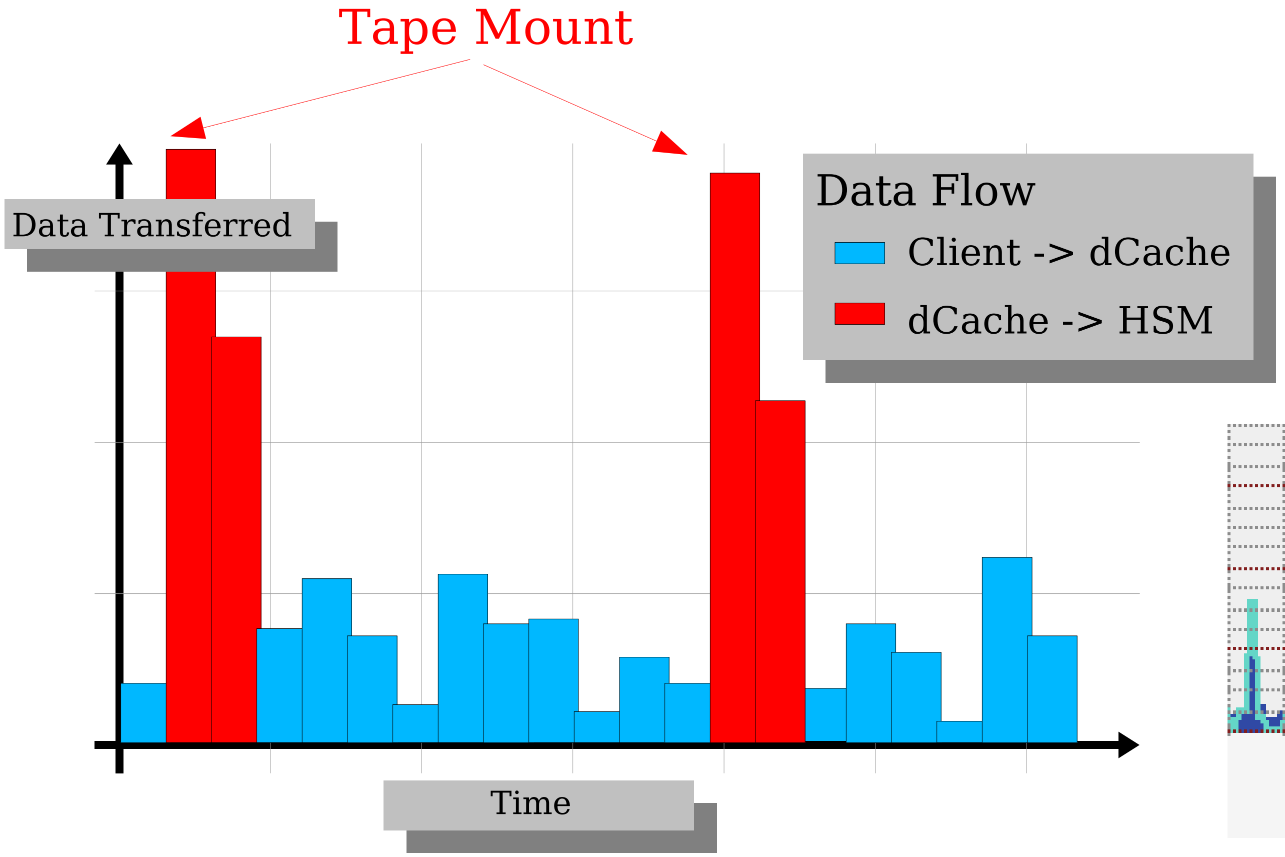


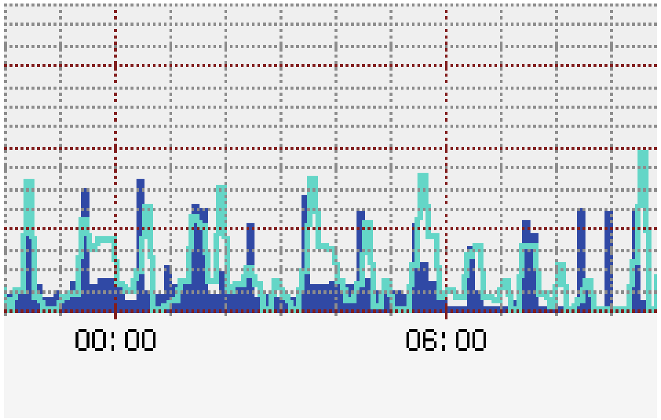
Storage Queues may belong to different HSM's.

dCache - HSM Interactions : deferred HSM flush



dCache - HSM Interactions : deferred HSM flush





The Pool Selection Mechanism

Static Configuration

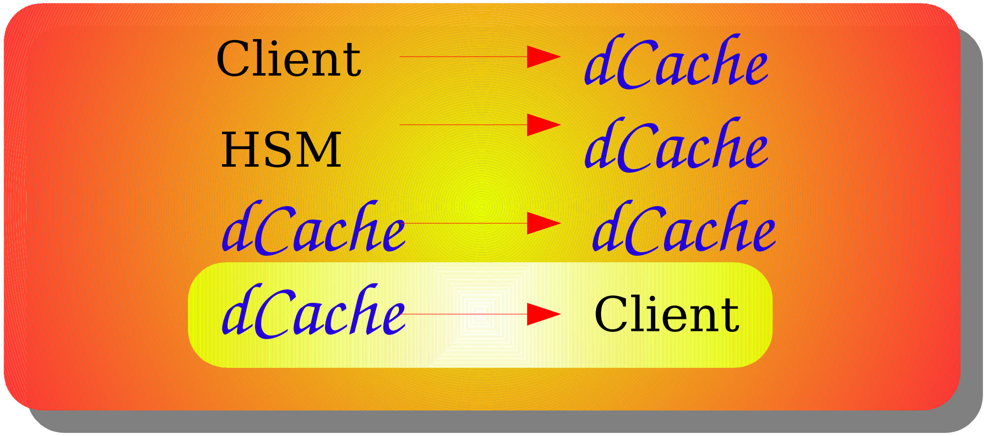
Use cases ...

Dynamic Behaviour

Tuning ...

Pool Selection Mechanism

Pool Selection required for



Client → dCache

HSM → dCache

dCache → dCache

dCache → Client

Pool selection is done in 2 steps

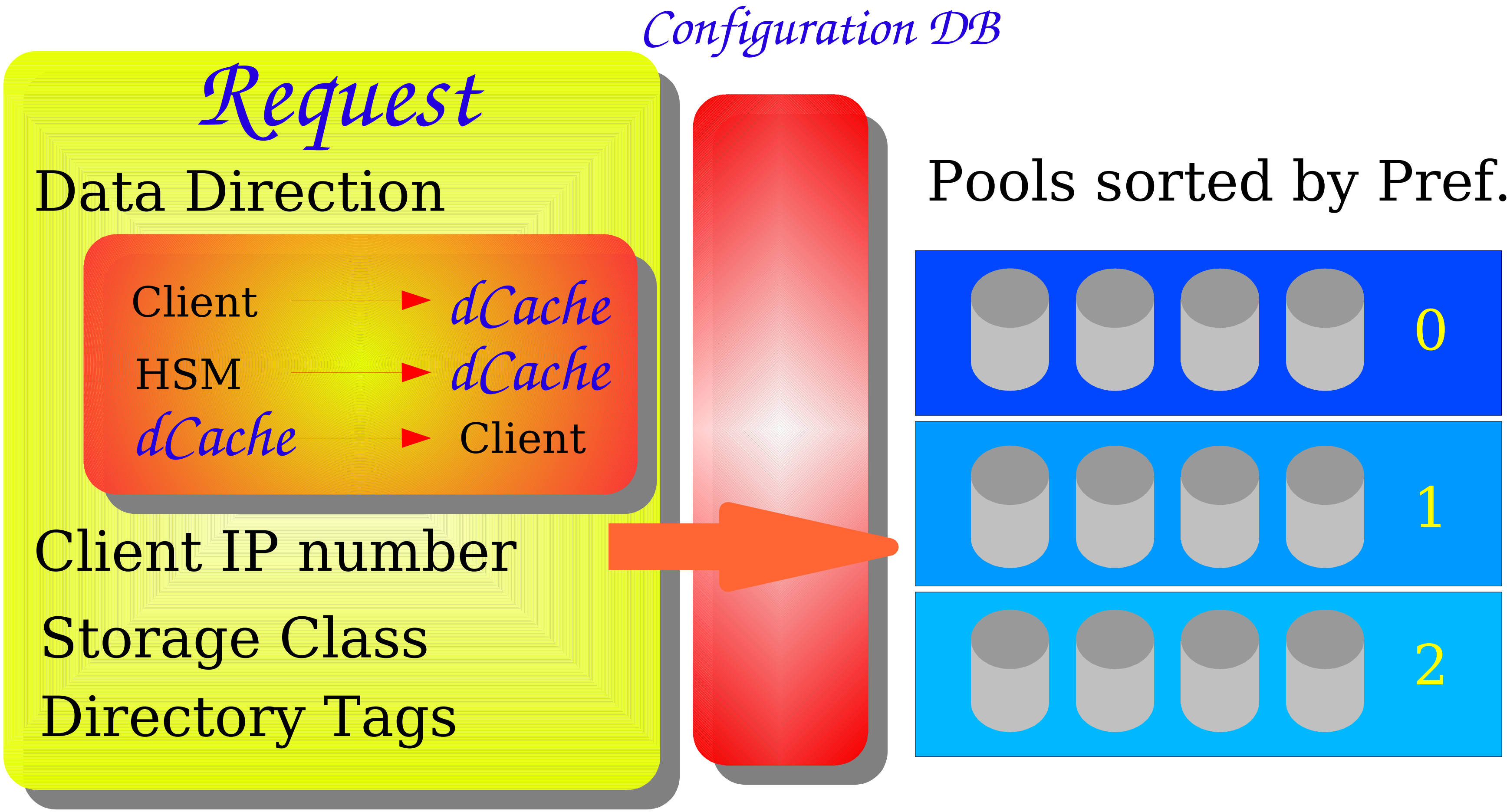
I) Query configuration database :

which pools are allowed for requested operation

II) Query 'allowed pool' for their vital functions :

find pool with lowest cost for requested operation

Pool Selection Mechanism : configuration



Configuration DB

Request

Data Direction

Client → dCache

HSM → dCache

dCache → Client

Client IP number

Storage Class

Directory Tags

Pools sorted by Pref.

0

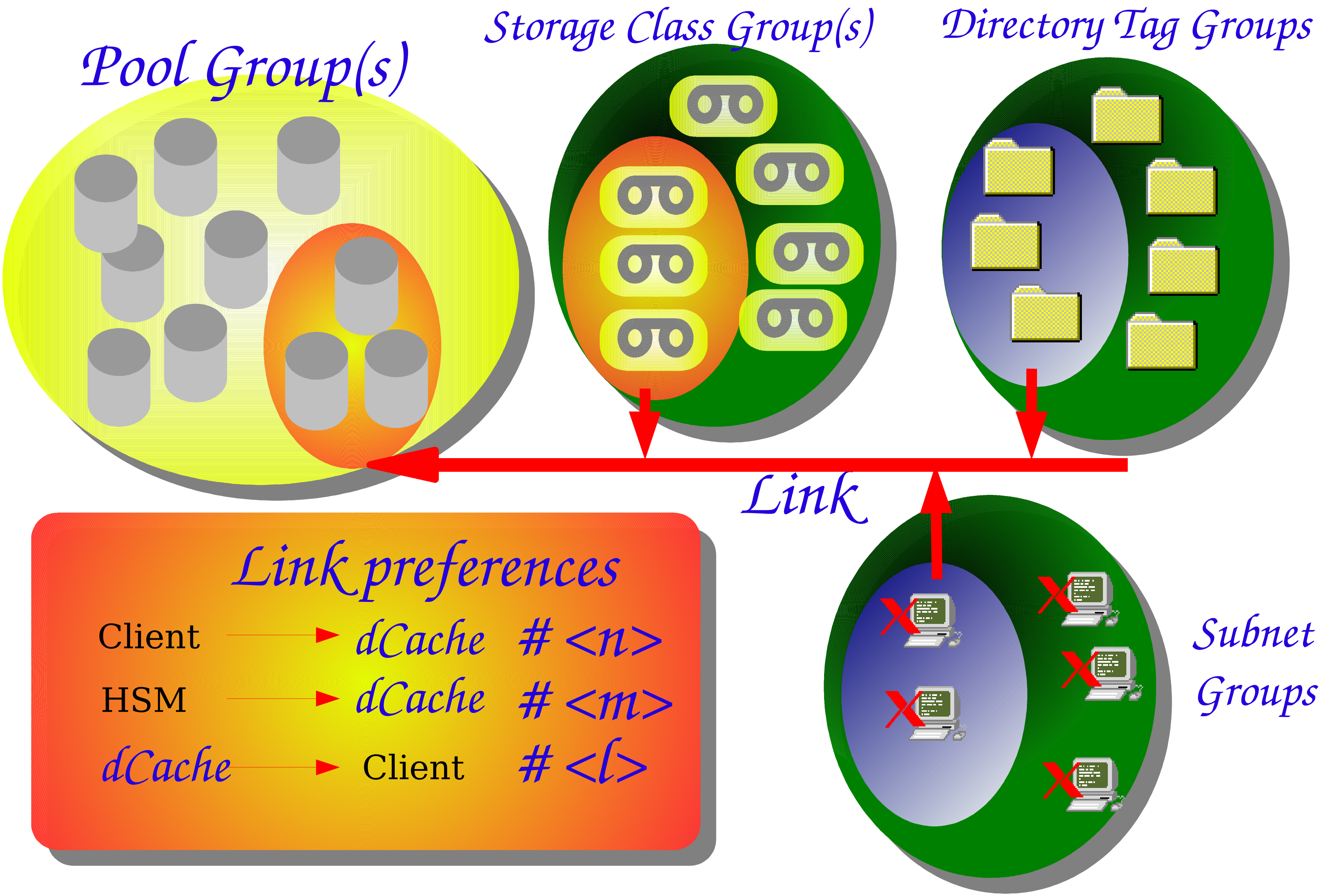
1

2

Mode A : fall-back only if all pools of pref. <x> are down.

Mode B : fall-back if cost of pools of pref. <x> is too high.

Pool Selection Mechanism : configuration



Link preferences

Client	→	dCache	# <n>
HSM	→	dCache	# <m>
dCache	→	Client	# <l>

Pool Selection Mechanism : configuration

Goals / Use cases

<u>Dedicated write pools (select by data direction)</u>

Allow 'precious' files on secure disks only.
Read requests will trigger p2p to cheap disks. (e.g. datataking)

<u>Support multiple HSMs (select by storage class)</u>

Assign different pool set to different HSMs (e.g. HSM migration)

<u>Support 'group owned' pool sets (select by storage class or tag)</u>

Assign 'experiment data' to 'experiment owned pools'
BUT have 'fallback' pools common to all experiments.

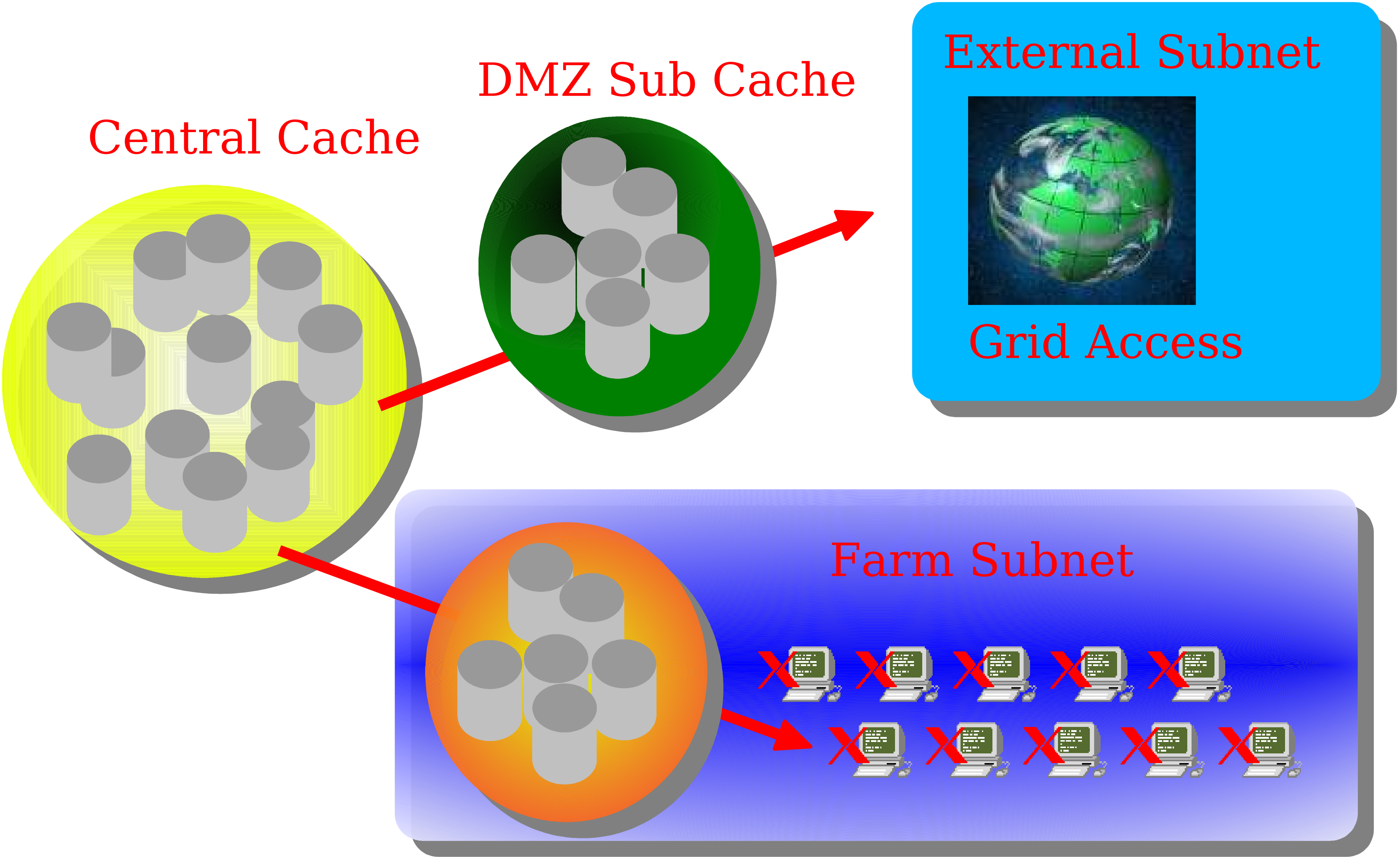
<u>Support 'working group' quotas (select by storage class or tag)</u>

Assign different number of pools to different working groups resp. 'data types' (raw,dst...)

<u>Special pools for farm subnets or external subnets</u>

e.g. : Grid users vers. Internal users.

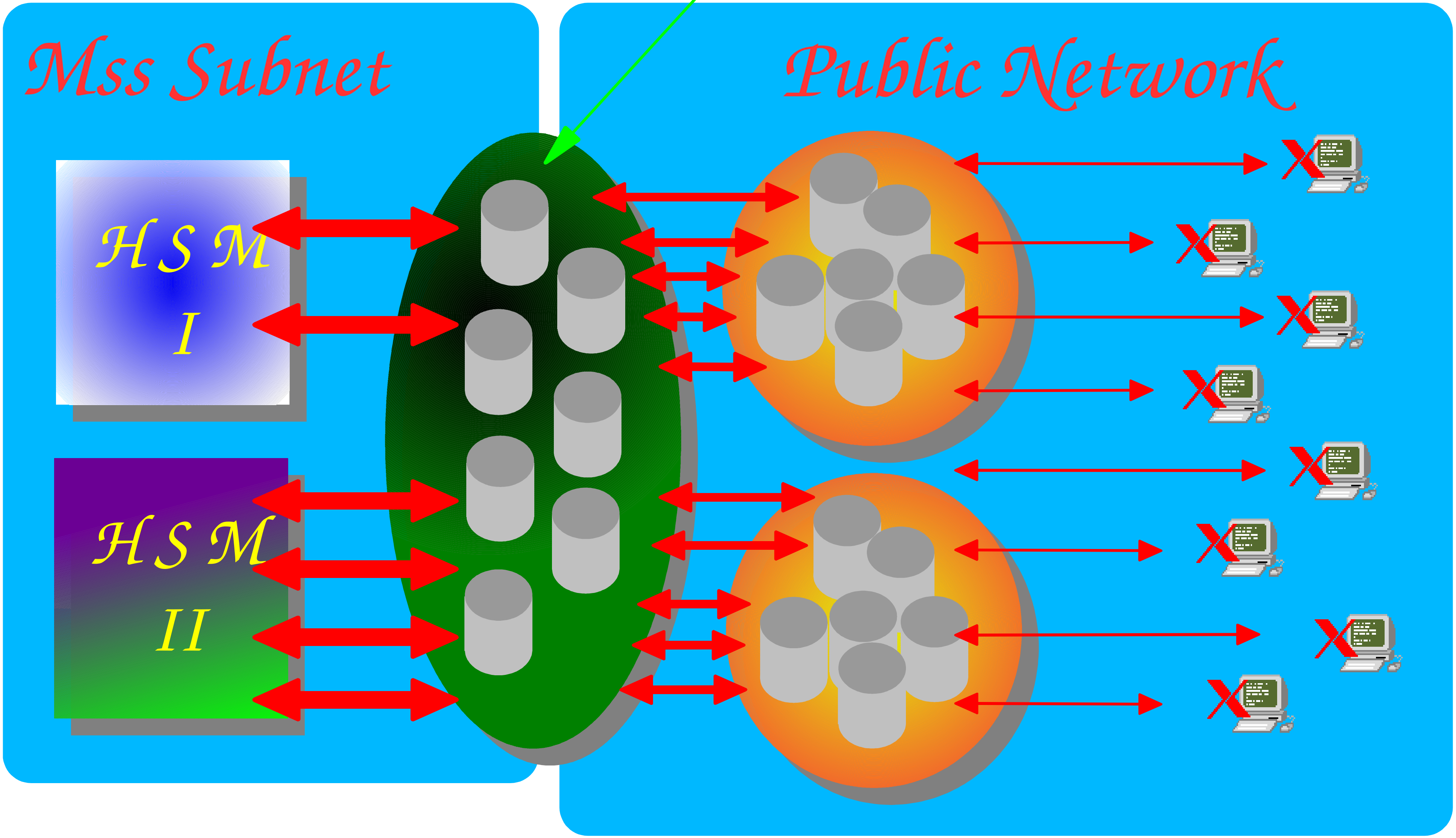
Use case : configuration by subnet



Use case : HSM decoupling

Dual Interface

One high speed link per drive



Pool Selection Mechanism : dynamic selection

Method

Frequent update of 'pools vital functions'

- available space
- least recently used 'timestamp'
- number of movers (in,out,store,stage,p2p)

Performing 'smart' guess between updates.

Goal

Uniform (even) distribution of requests per pool for requests coming 'in bunches'.

Pool Selection Mechanism : Tuning (1)

Space vs. Load

For each request, the central cost module generates two cost values for each pool :

Space : Cost based on available space or LRU timestamp

CPU : Cost based on the number of different movers (in,out,...)

The final cost, which is used to determine the best pool, is a linear combination of Space and CPU cost.

The coefficients needs to be configured.

Space coefficient << Cpu coefficient

Pro : Movers are nicely distributed among pools.

Con : Old files are removed rather than filling up empty pools.

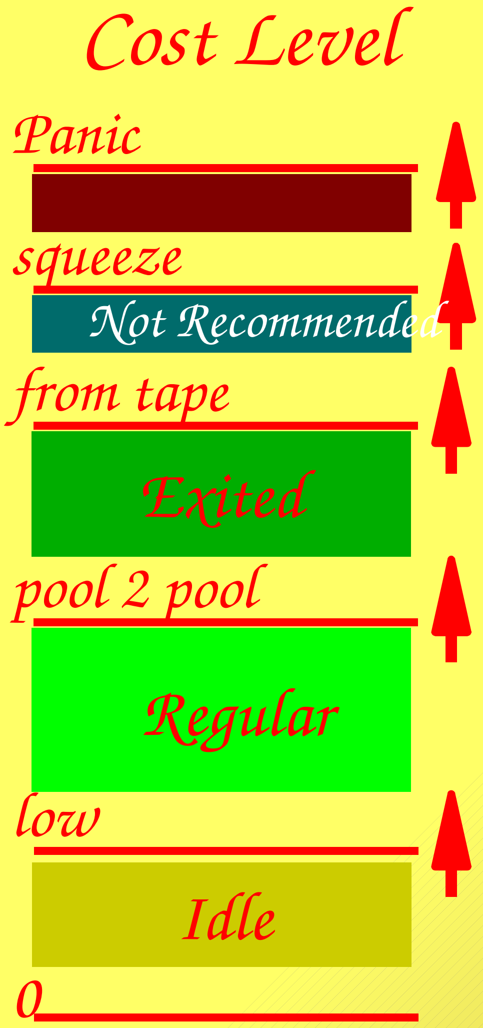
Space coefficient >> Cpu coefficient

Pro : Empty pools are filled up before any old file is removed.

Con : 'Clumping' of movers on pools with very old files or much space.

Pool Selection Mechanism : Tuning (2)

Pool to pool transfers etc. ...



Cost Level

Panic

PANIC

squeeze

Not Recommended

Take file from pool with lowest cost.

from tape

Fetch file from Hsm to cheap pool, before delivering to it to client.

Exited

pool 2 pool

Copy file to cheap pool first, before delivering it to client.

Regular

low

Take file from pool with lowest cost, otherwise try to get rid of duplicates.

Idle

0

dCache : Basic Design
Road map of a data transfer request

dCache Basic Design

Involved Components

Door

- Prot. specific end point for client connection (inetd)
- Stays alive as long as client proc. is alive
- Clients proxy within the dCache world

Name space Provider (Pnfs)

Interface to a file system name space

A) Maps dCache name space operations to filesystem operations

B) Stores extended file metadata (dCache or external)

PoolManager

Performs pool selection

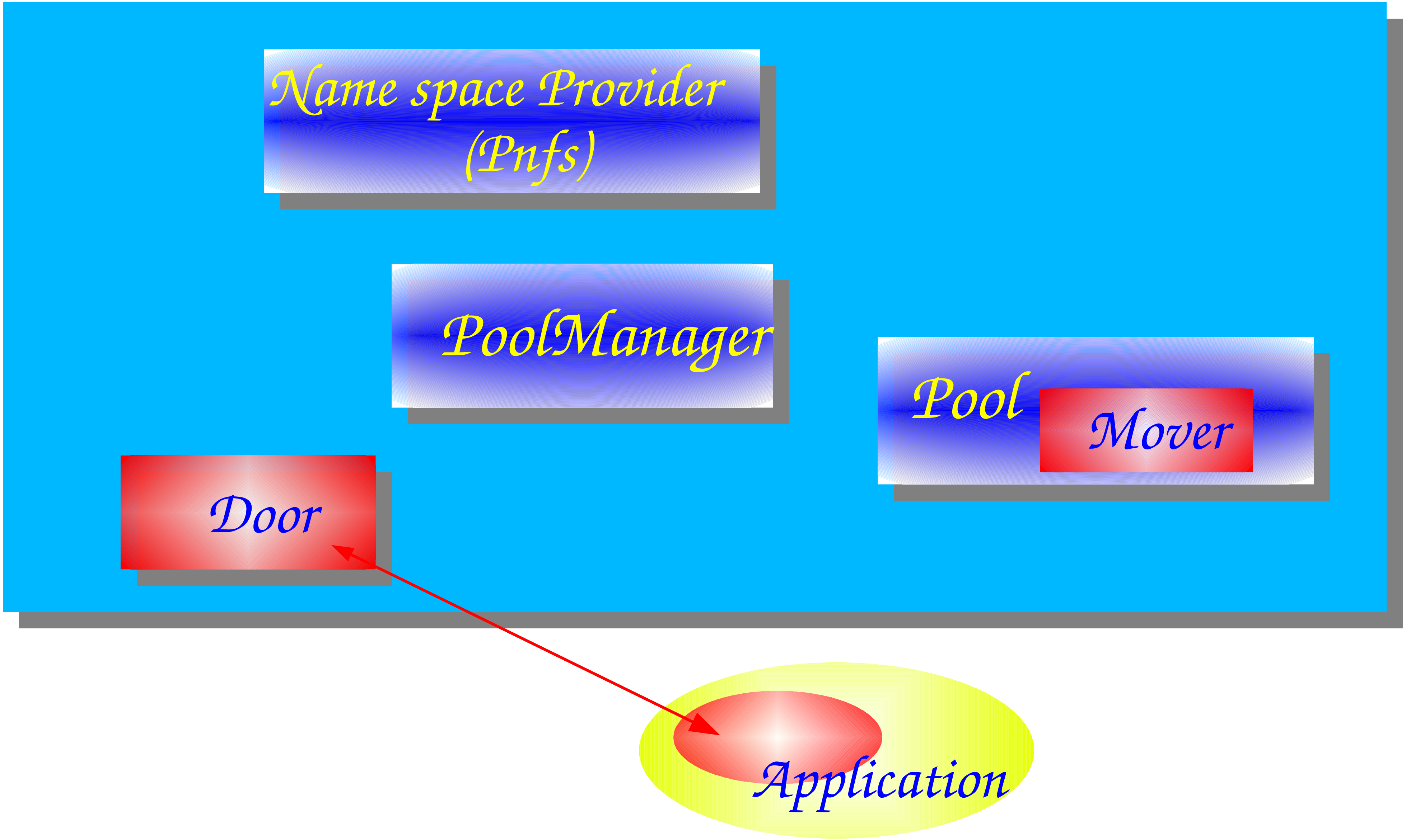
Pool **Mover**

- Data Repository handler (cleaner a.s.o)
- Launches requested data transfer protocols
- Data transfer protocol handler

dCap, http, ftp, HSM hooks.

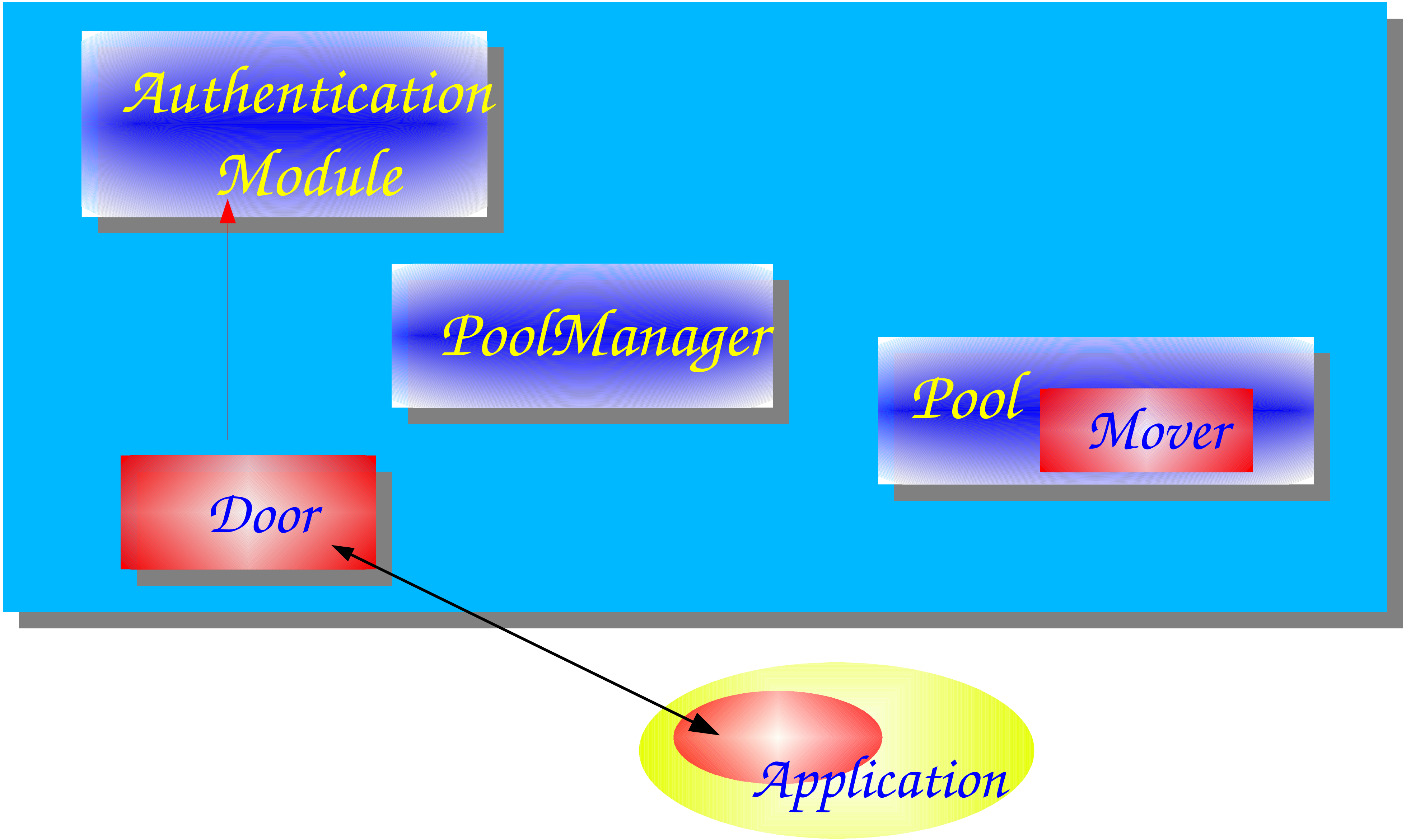
dCache Basic Design

Connect to well known door



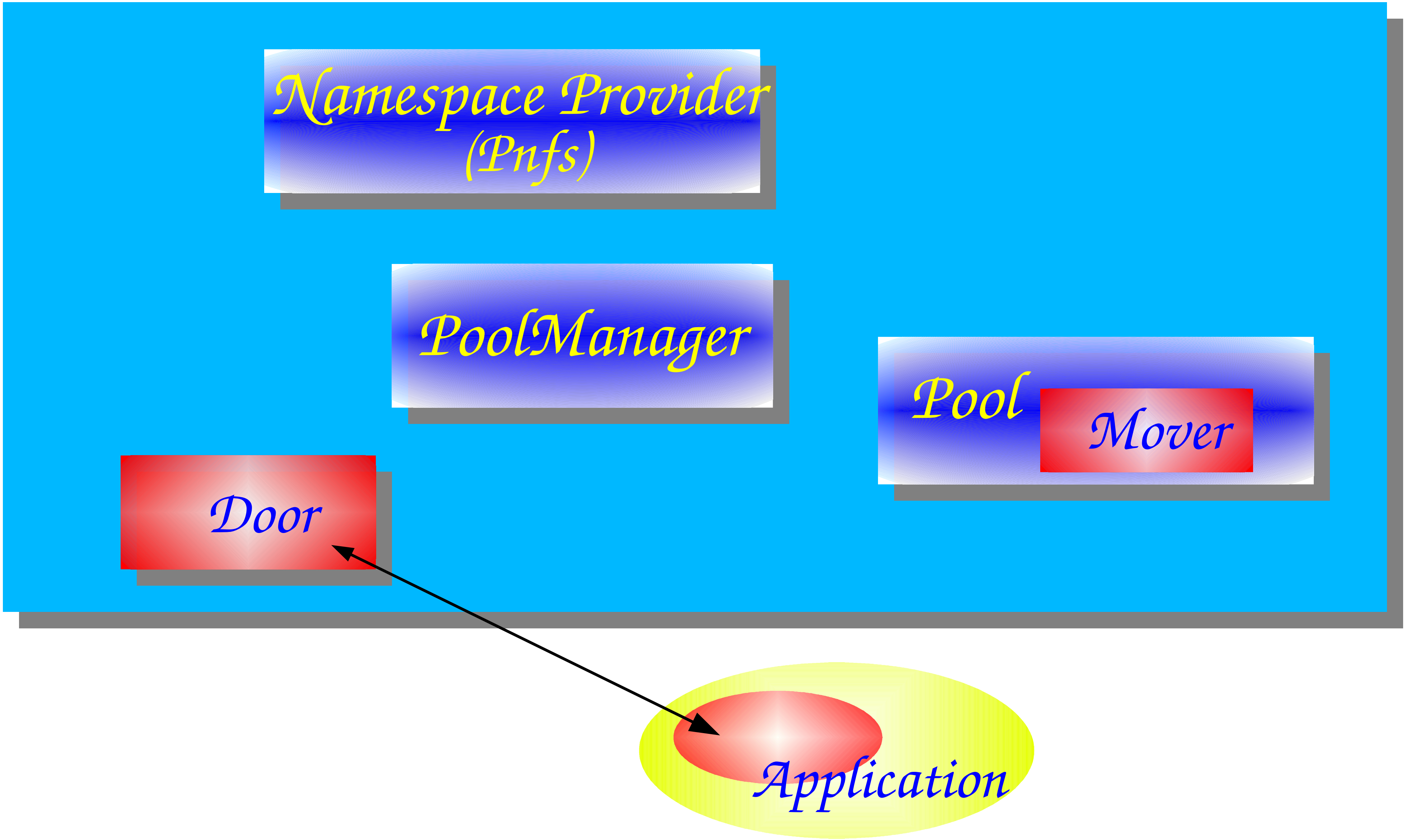
dCache Basic Design

Authenticate User



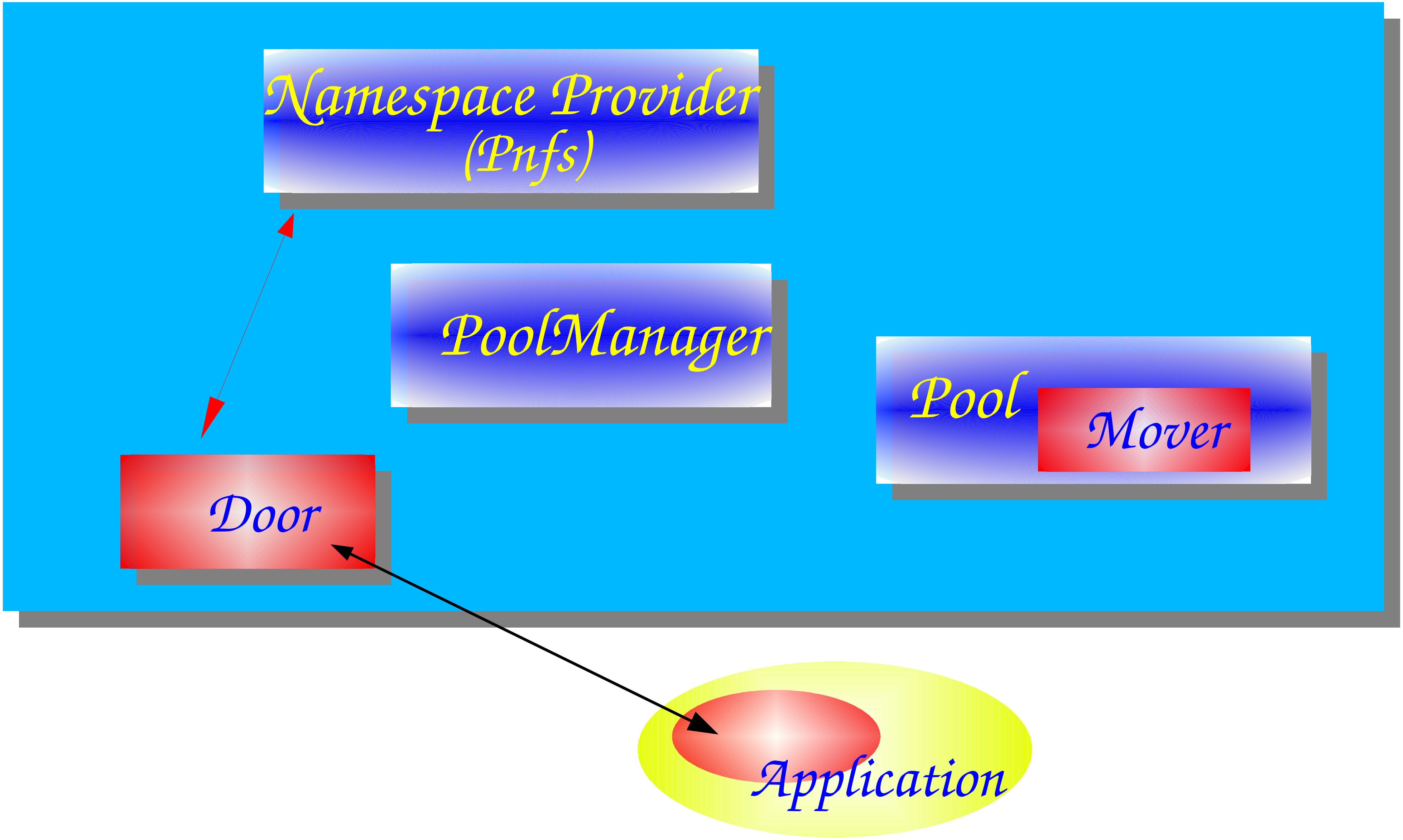
dCache Basic Design

Send 'get file' request



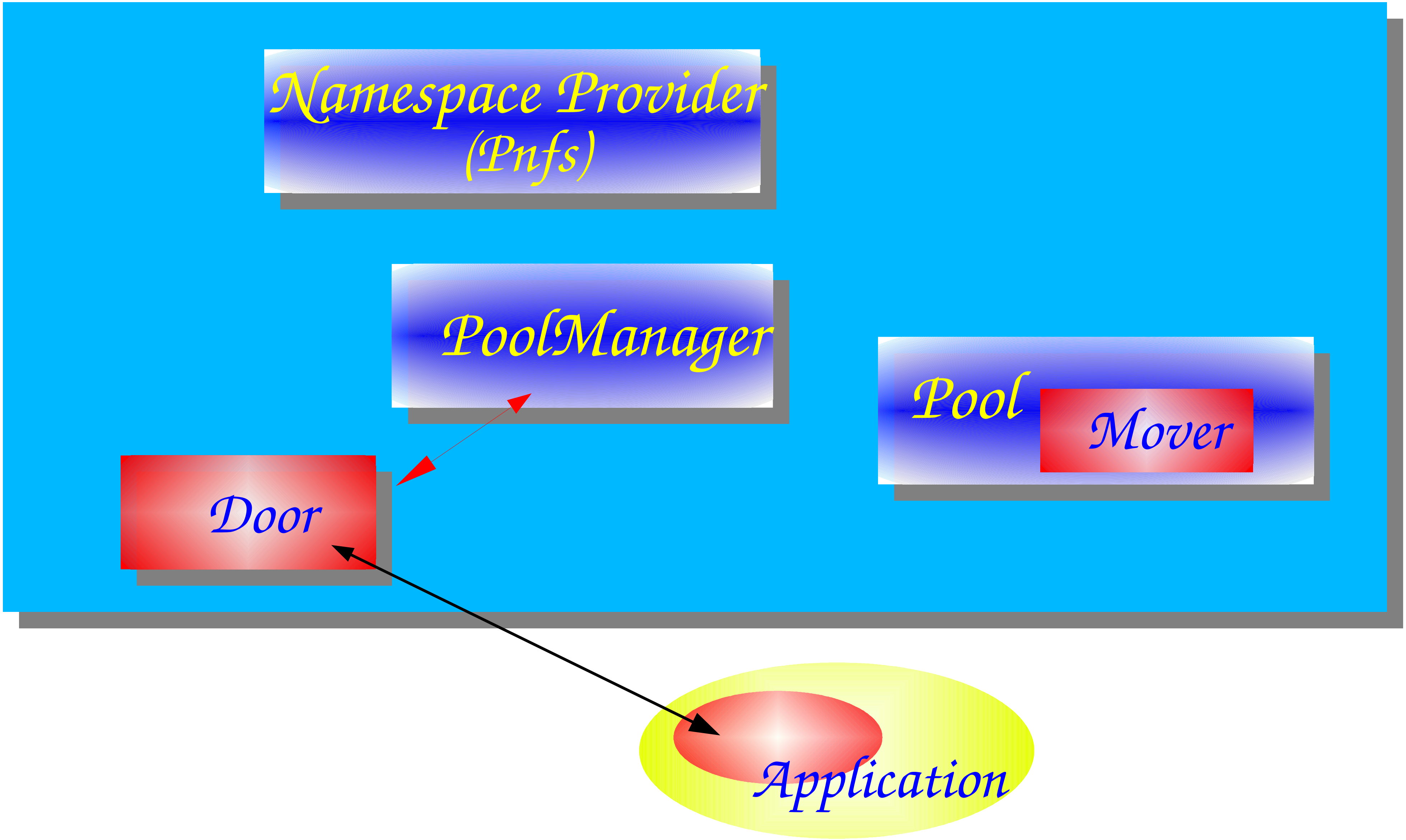
dCache Basic Design

Ask for meta data



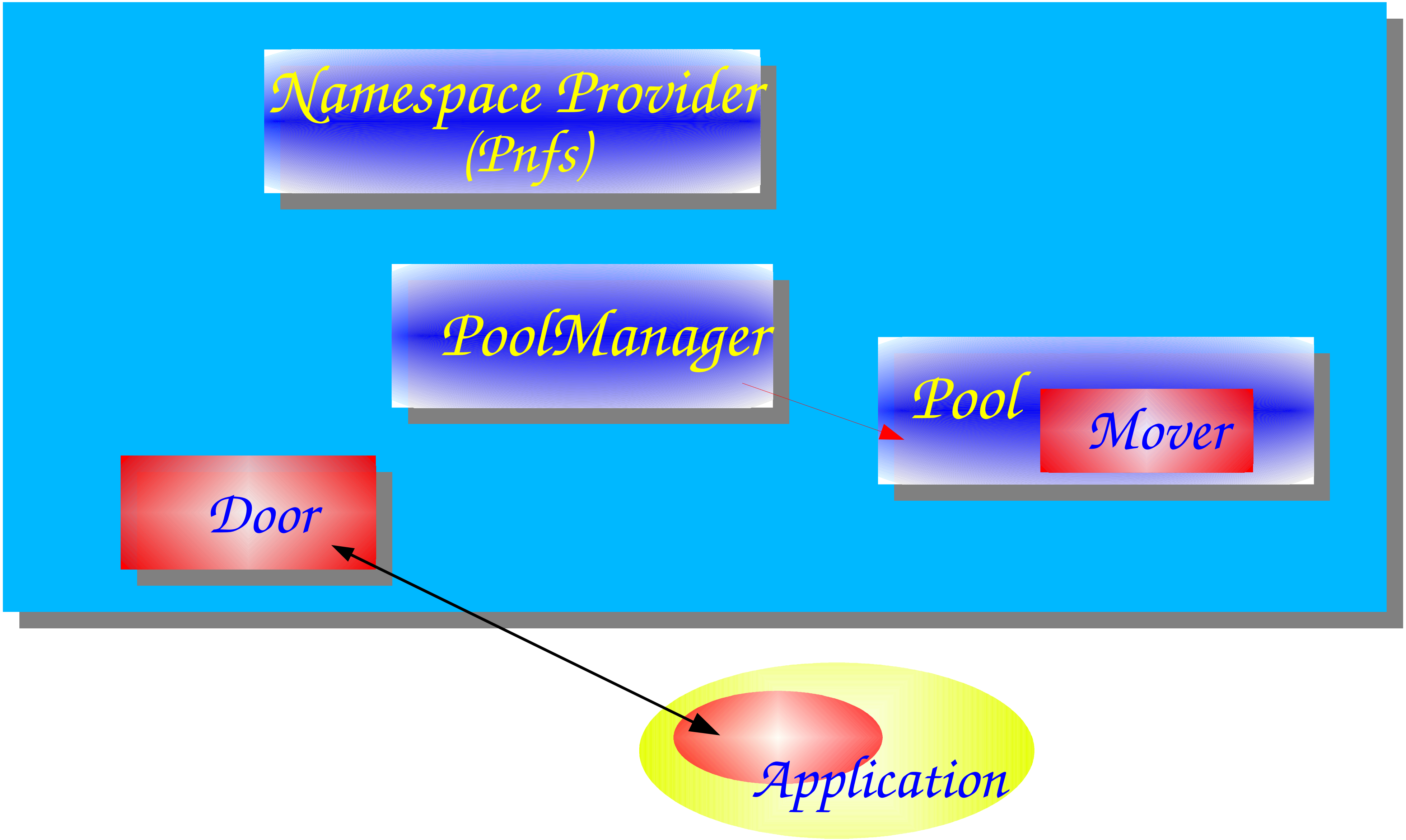
dCache Basic Design

Ask for best pool

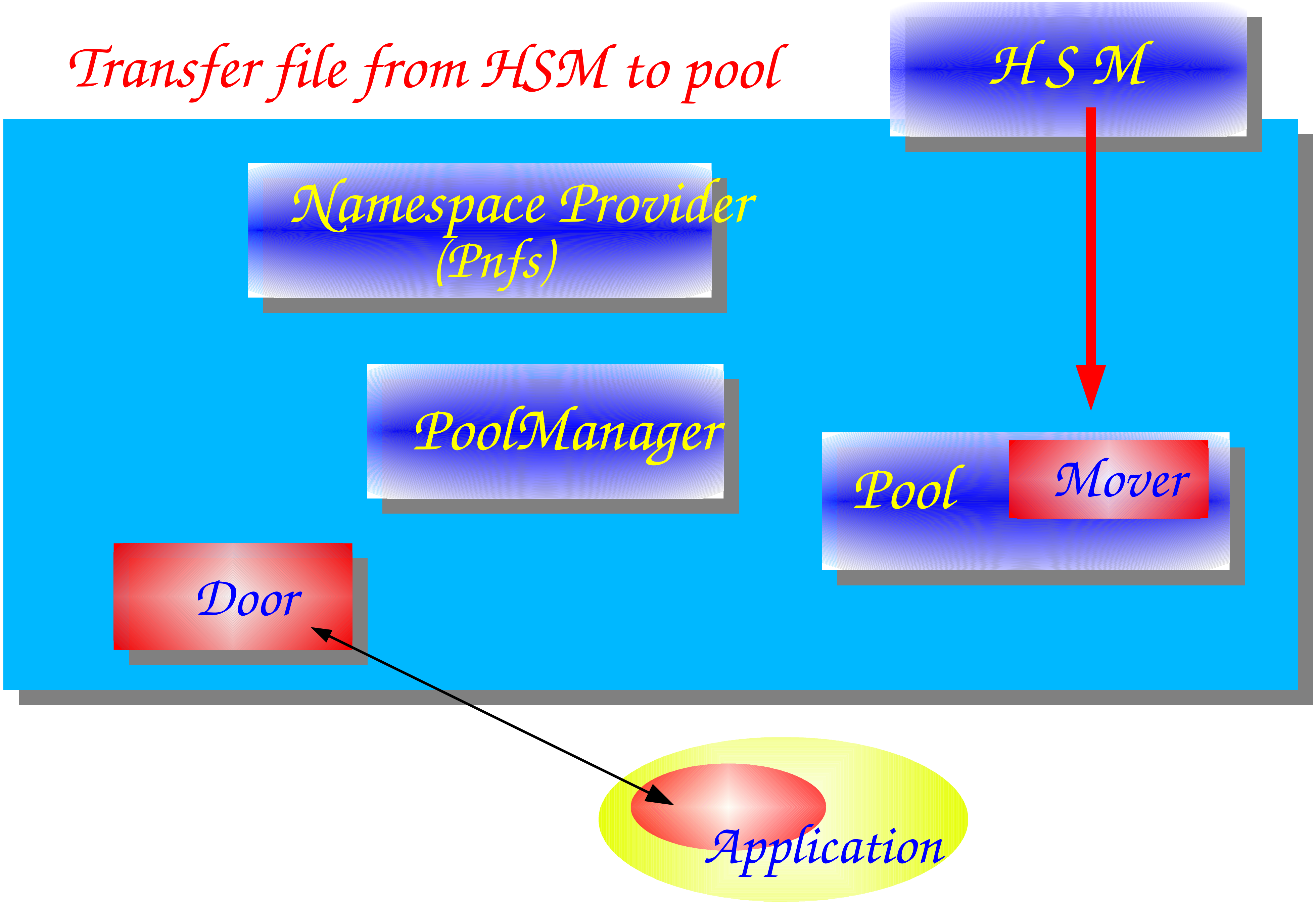


dCache Basic Design

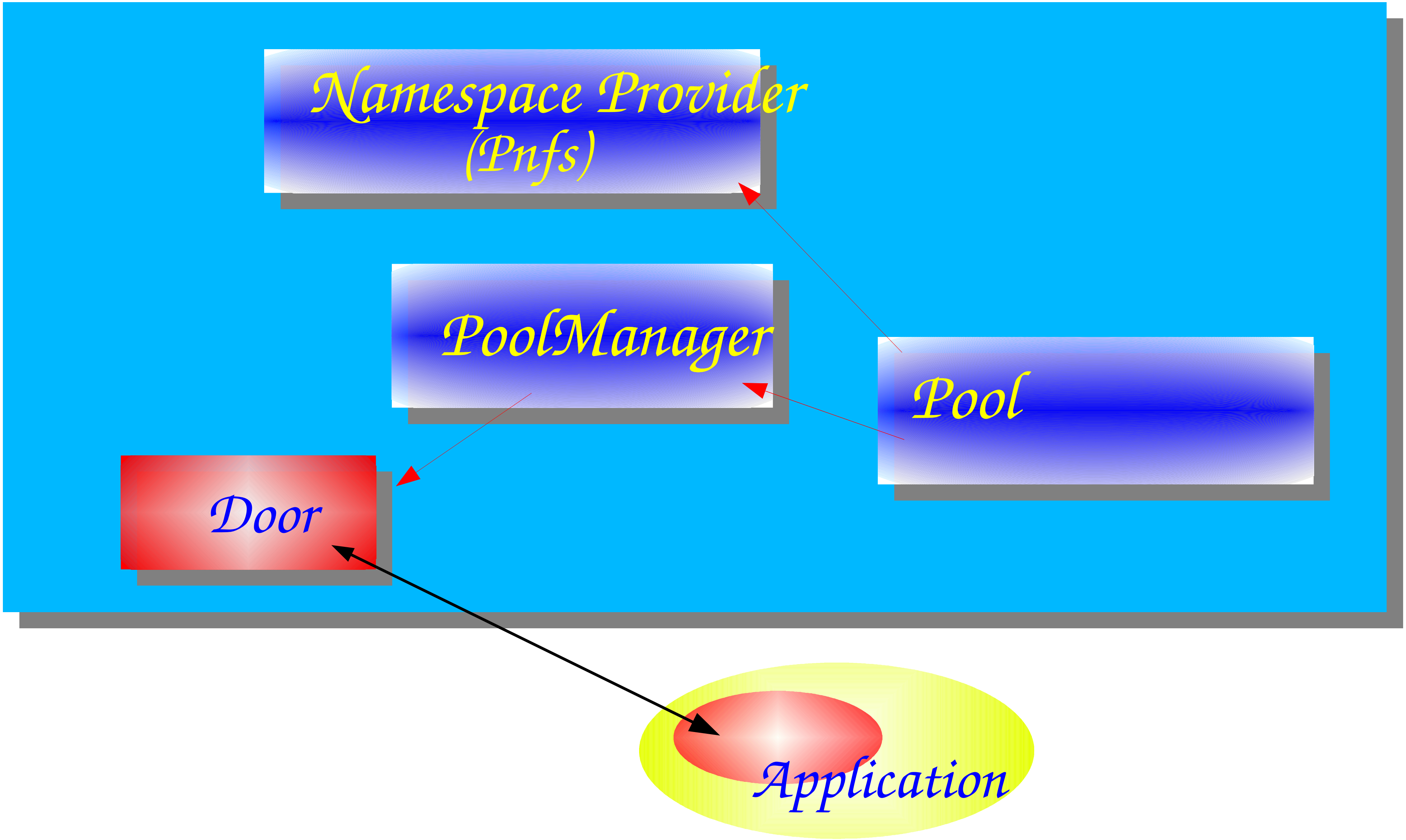
make file ready for transfer



Transfer file from HSM to pool

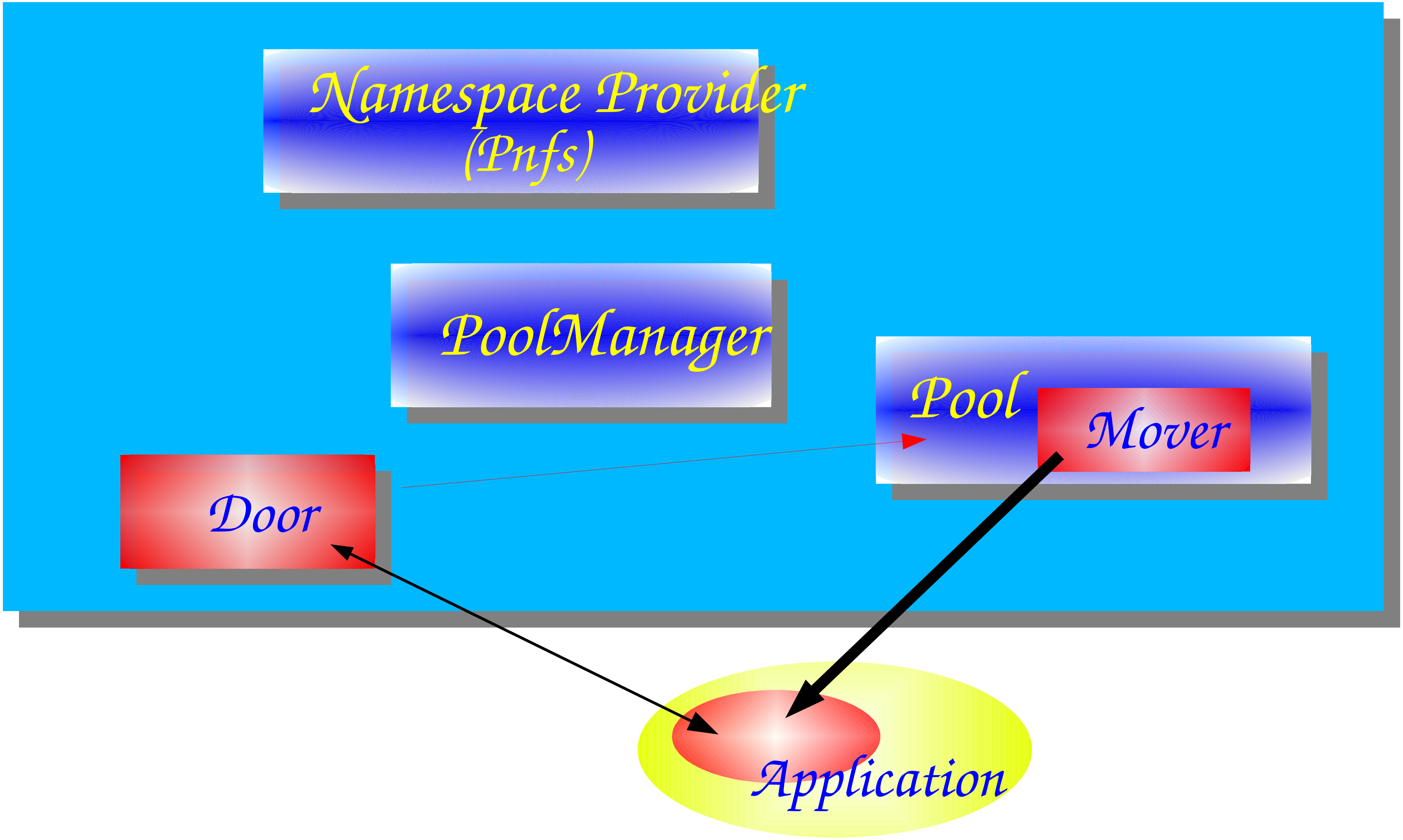


File ready & update meta data



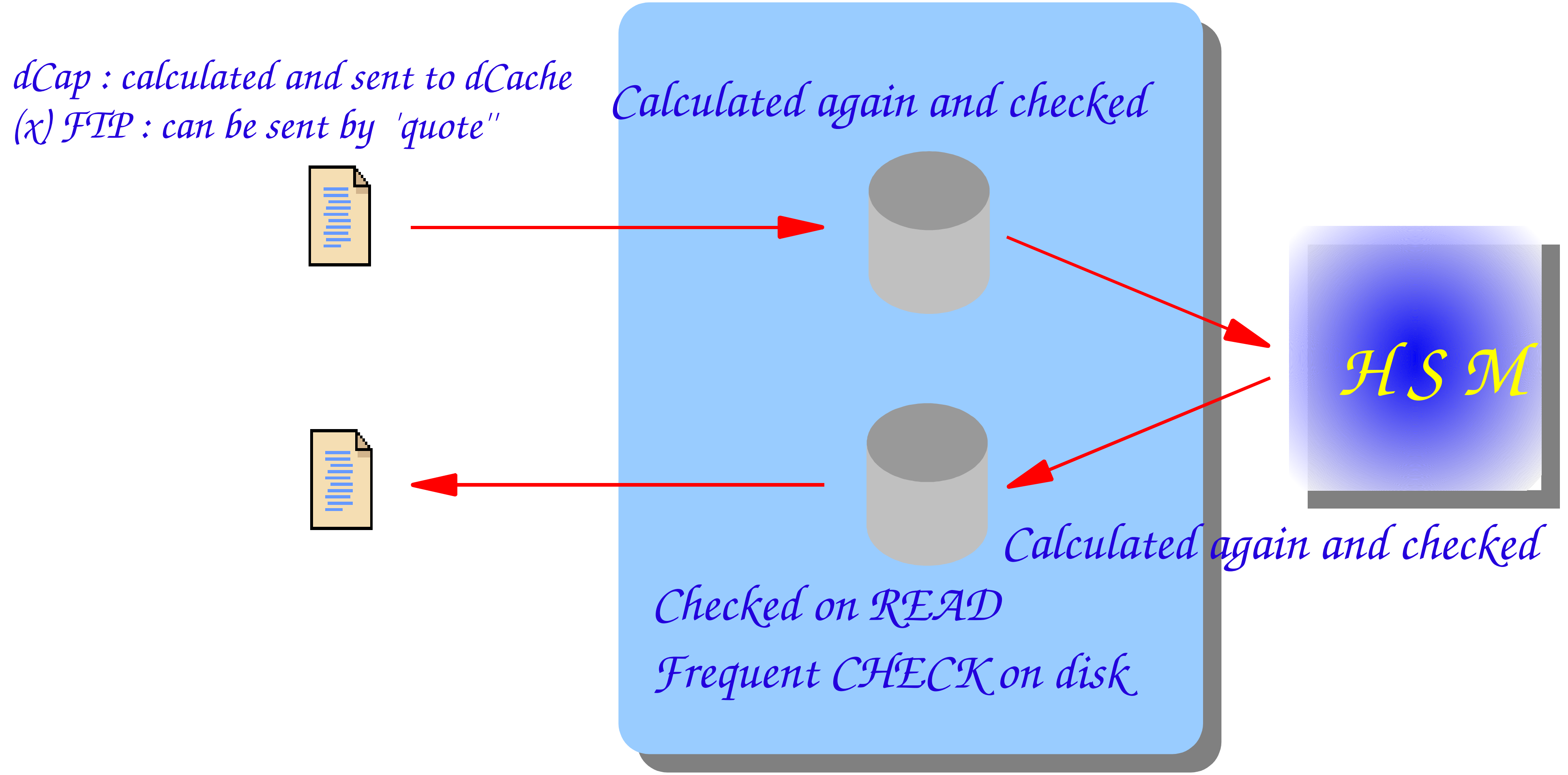
Transfer file from pool to client

dCache Basic Design



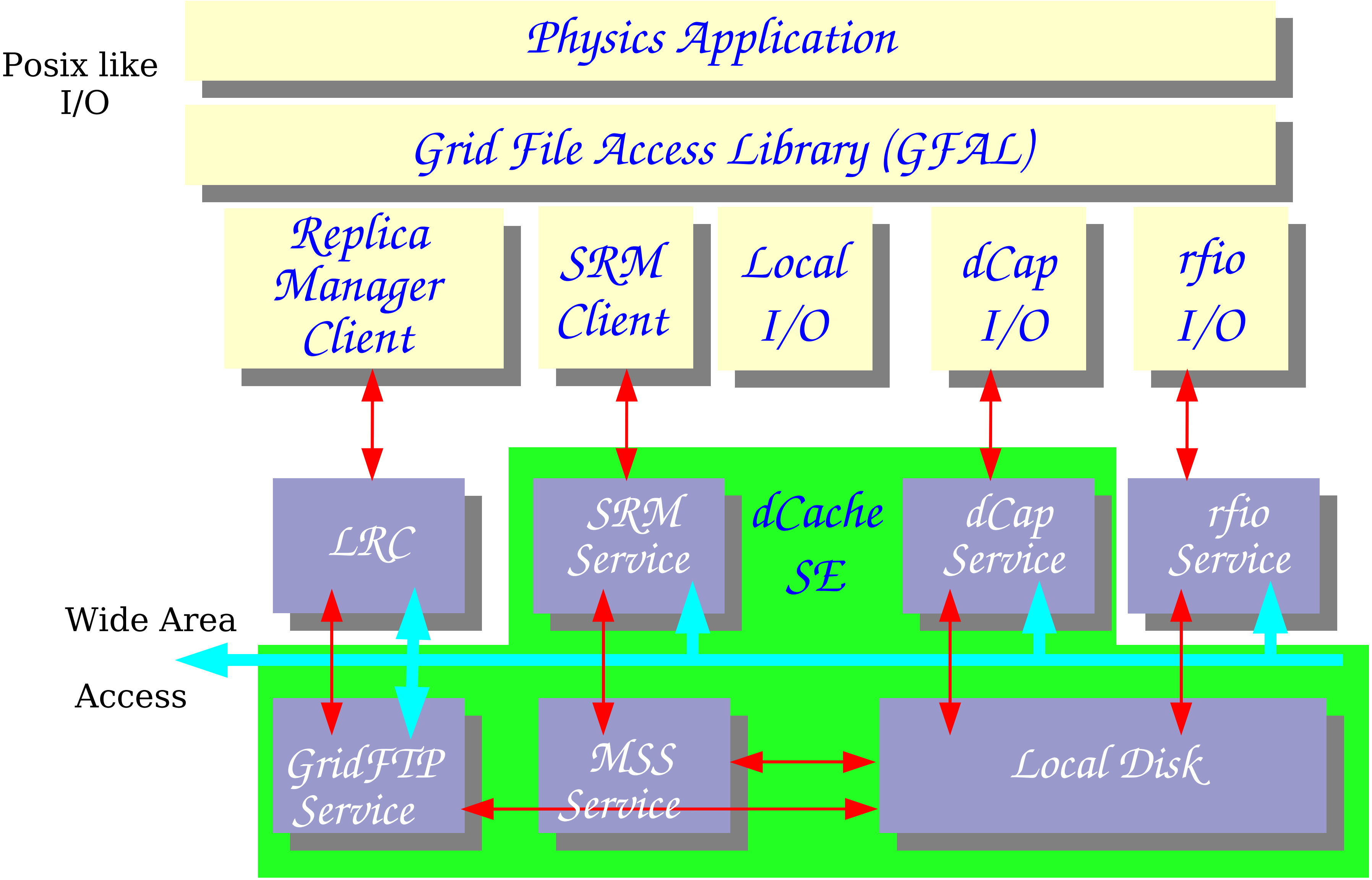
dCache Basic Design : checksum

Supported Type : adler32


dCap : calculated and sent to dCache
(x) FTP : can be sent by 'quote''
Calculated again and checked
H S M
Calculated again and checked
Checked on READ
Frequent CHECK on disk

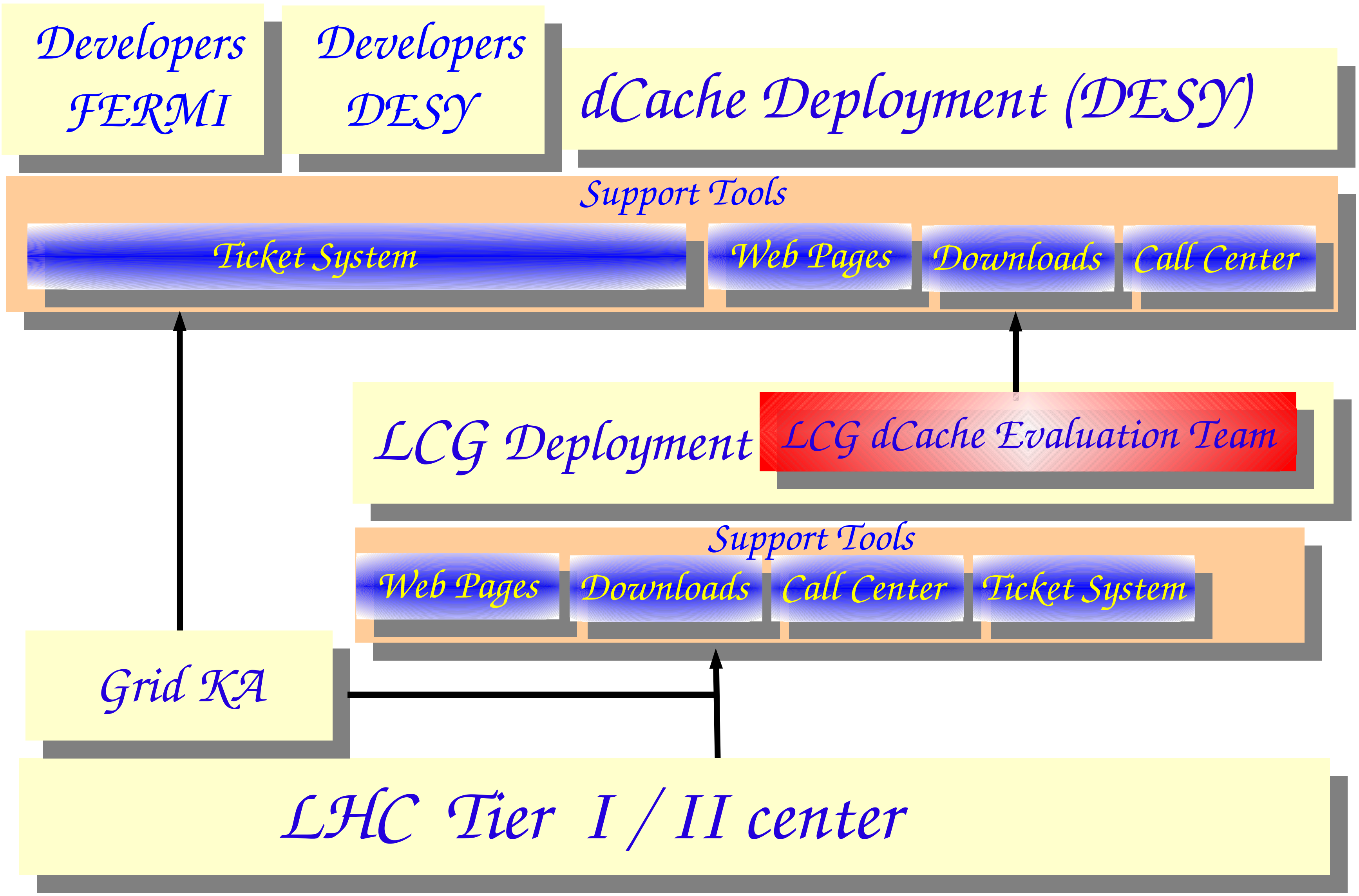
dCache
End of official presentation

LCG Storage Element : File access



Source : Michael Ernst 18/5/2004

dCache - LCG support model



Developers FERMI

Developers DESY

dCache Deployment (DESY)

Support Tools

Ticket System

Web Pages

Downloads

Call Center

LCG Deployment

LCG dCache Evaluation Team

Support Tools

Web Pages

Downloads

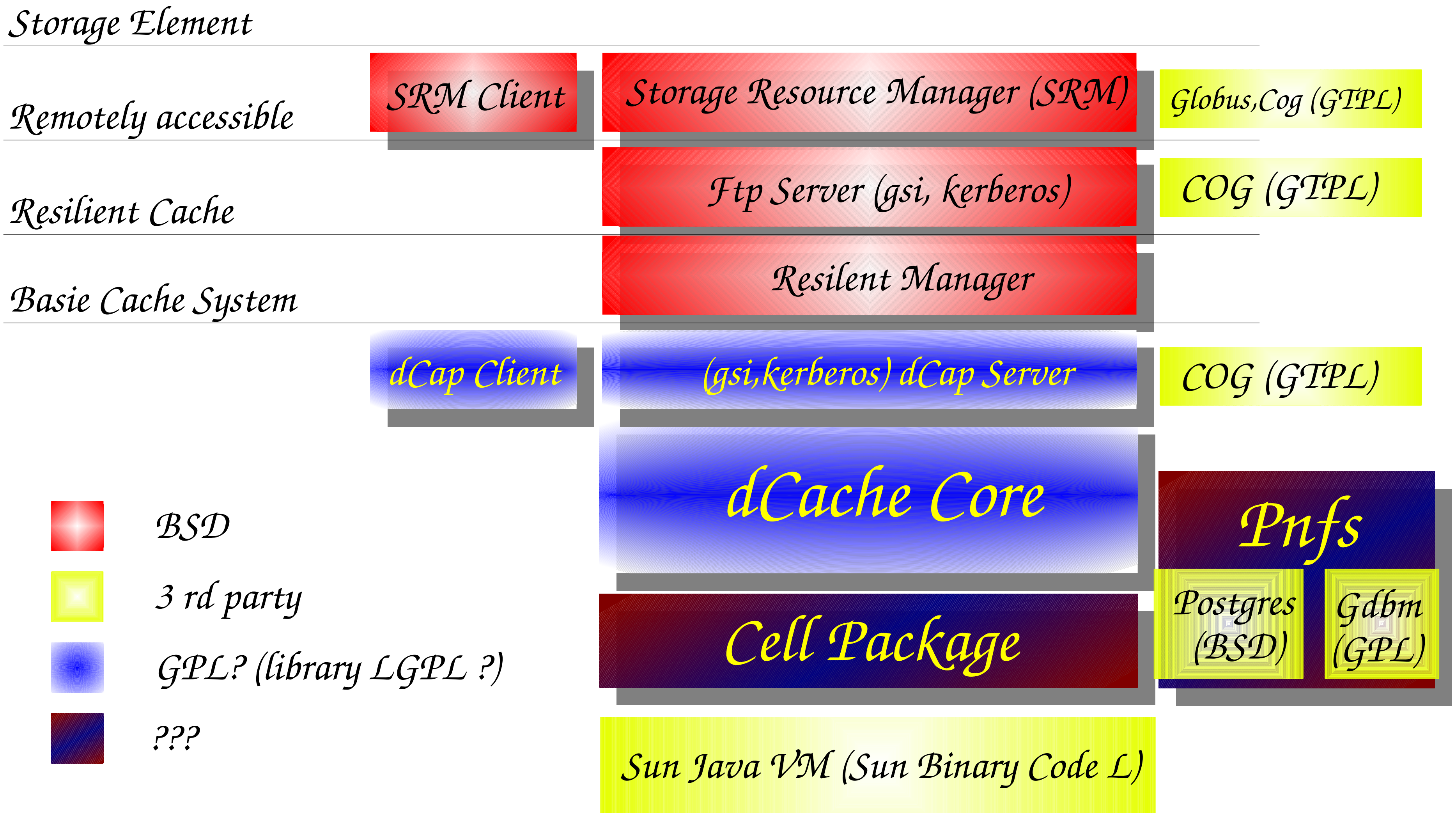
Call Center

Ticket System

Grid KA

LHC Tier I / II center

dCache Component License Model



Storage Element

Remotely accessible

Resilient Cache

Basie Cache System

SRM Client

Storage Resource Manager (SRM)

Globus,Cog (GTPL)

Ftp Server (gsi, kerberos)

COG (GTPL)

Resilent Manager

dCap Client

(gsi,kerberos) dCap Server

COG (GTPL)

dCache Core

Pnfs

Postgres (BSD)

Gdbm (GPL)

Cell Package

Sun Java VM (Sun Binary Code L)

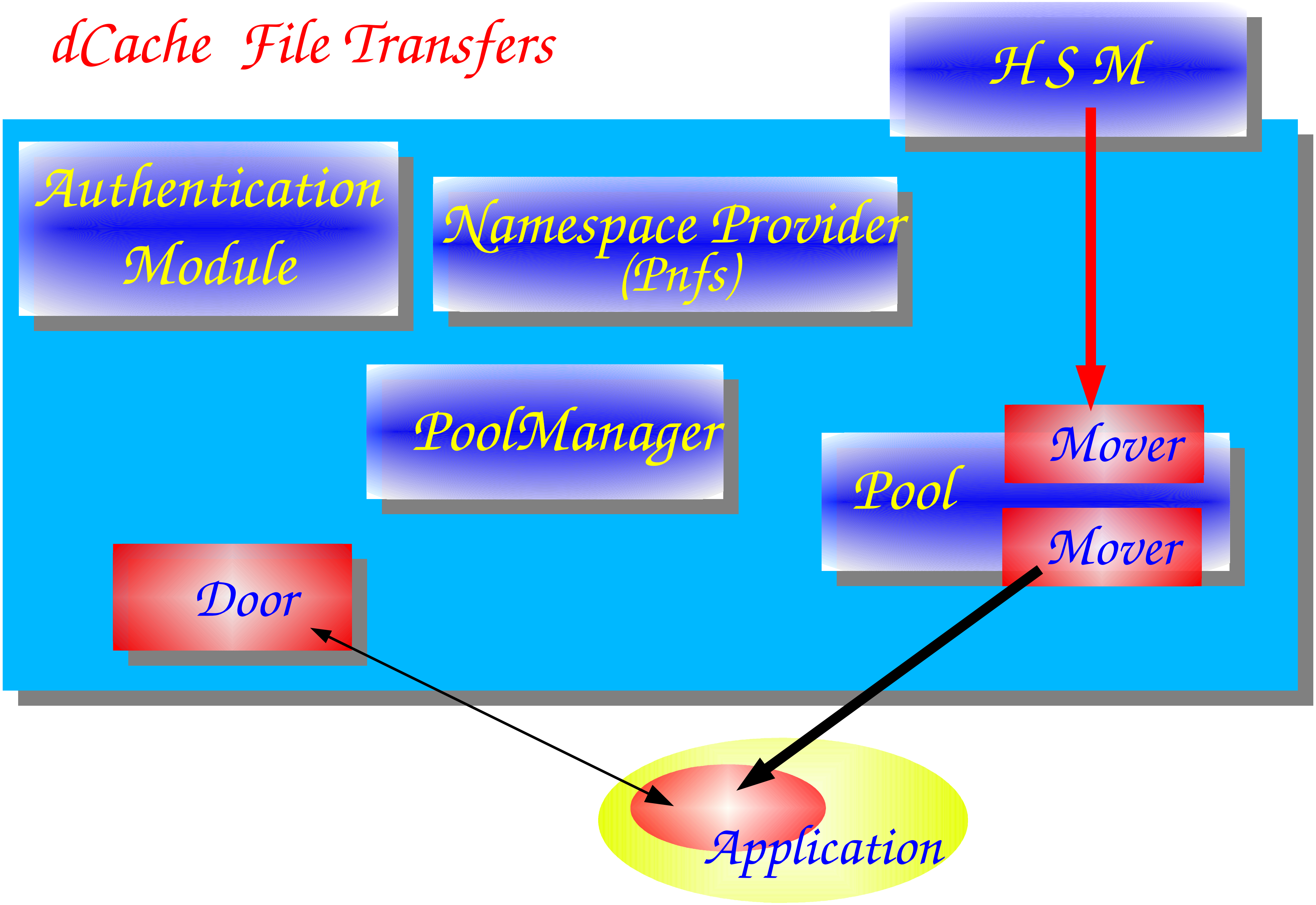
🟥 BSD

🟨 3 rd party

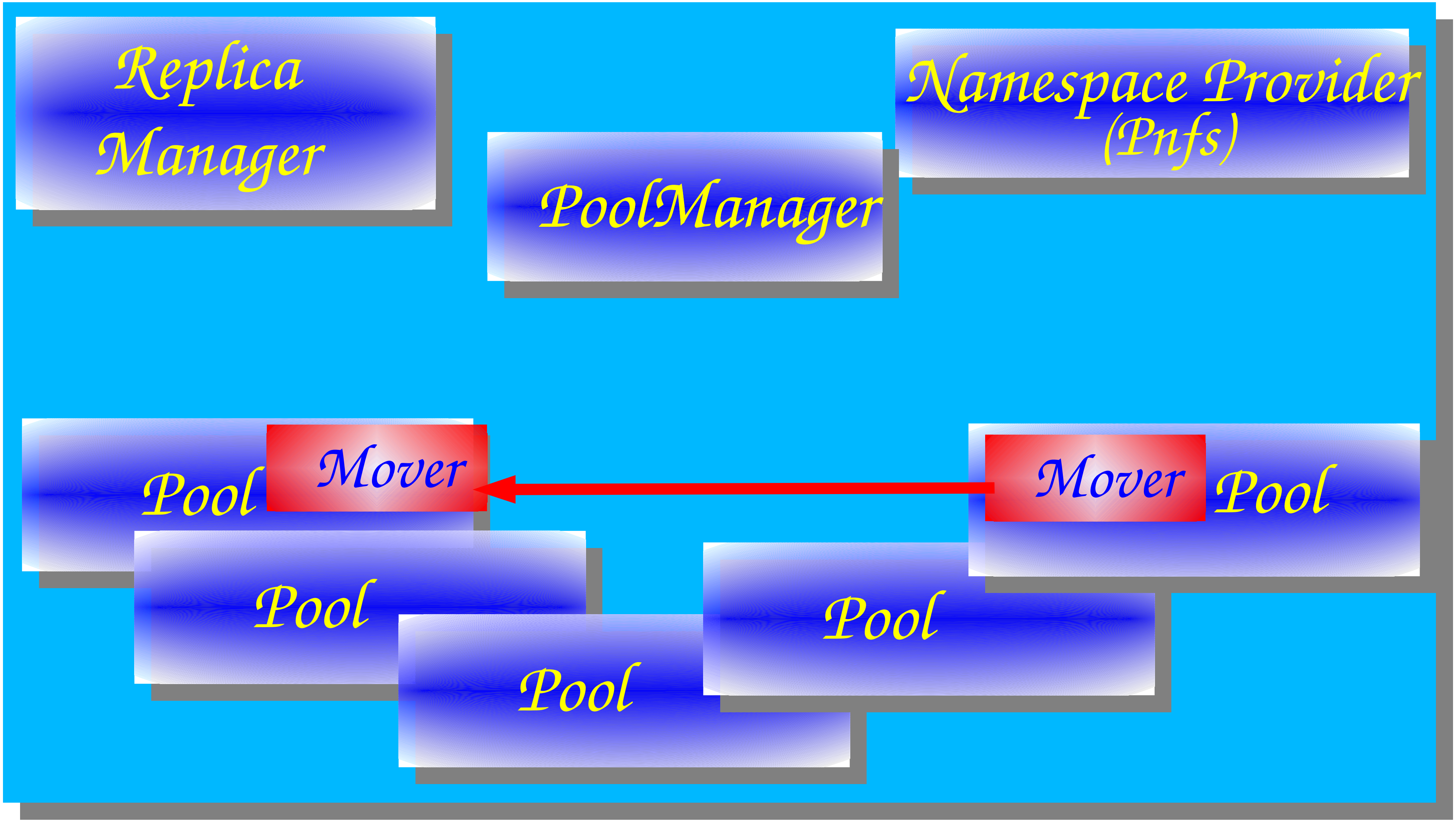
🟦 GPL? (library LGPL ?)

▨ ???

dCache File Transfers



Resilient dCache


Replica Manager
PoolManager
Namespace Provider (Pnfs)
Pool
Mover
Mover
Pool
Pool
Pool
Pool