Lustre – the inter-galactic cluster file system?

Peter J. Braam

braam@clusterfs.com

http://www.clusterfilesystems.com

Cluster File Systems, Inc

Talk overview

- Lustre is a new storage architecture
  - Object Storage
  - Storage management
  - File System
  - Locking
  - Networking

What is Lustre?

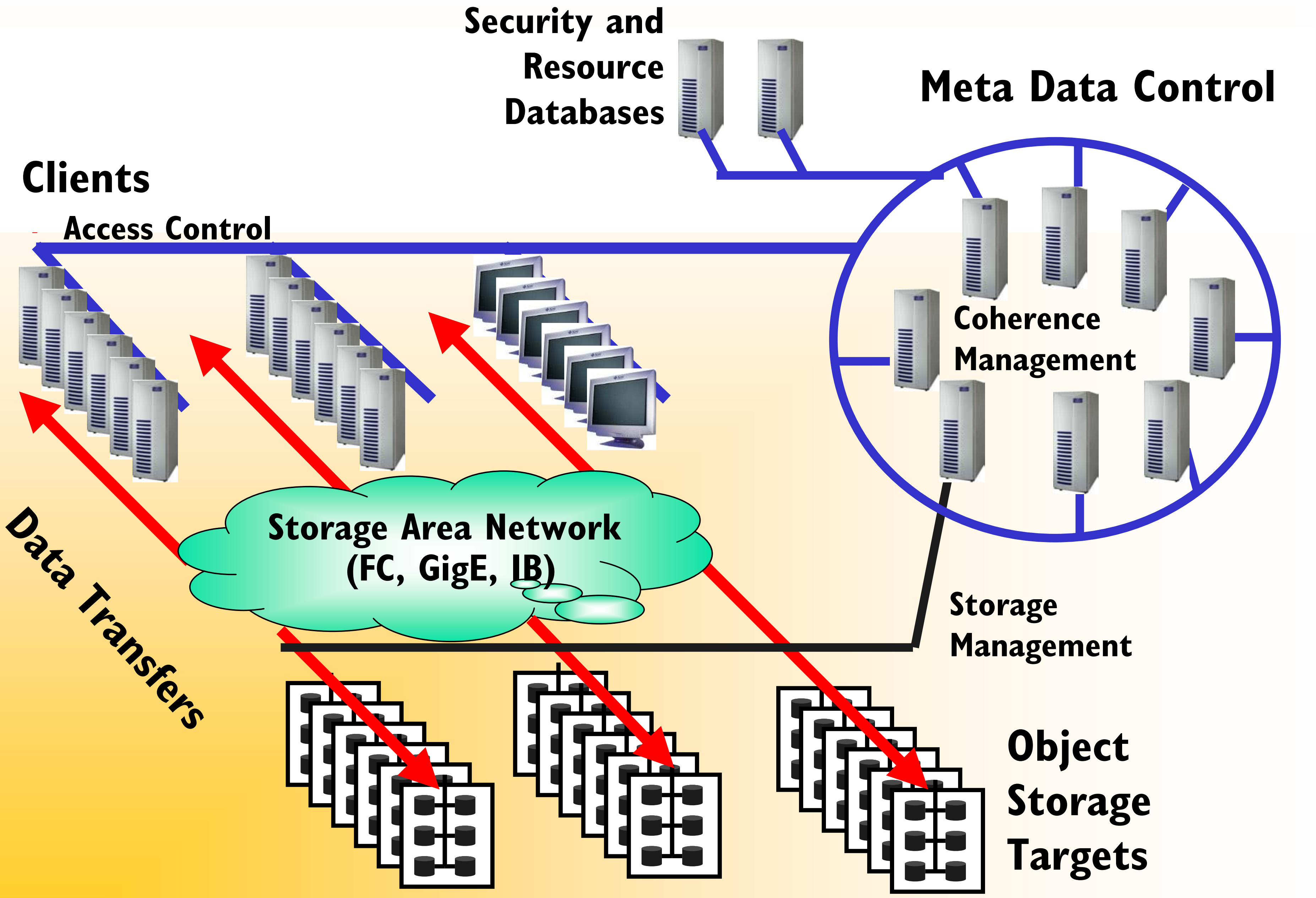
The SGS-FS challenge

- Characteristics:
  - 100's GB's/sec of I/O throughput
  - trillions of files
  - 10,000's of nodes
  - Petabytes
- First put forward 1999 Santa Fe SGPFS meeting
  - nickname: "The Inter-Galactic File System"

Project history

- Braam pursued this for 3 years:
  - 1999 CMU – Seagate – Stelias Computing
  - 2000 Los Alamos, Sandia, Livermore:
    - need new Intergalactic File System
  - 2001: Lustre design to meet the SGS-FS requirements?
  - 2002:
    - Lustre on MCR (1000 node Linux Cluster – bigger ones coming)
    - Lustre Hardware (BlueArc)
    - ASCI Path Forward contract (with HP and Intel)

Big Lustre picture



Cluster File Systems, Inc

Key issues: Scalability

- I/O throughput
  - How to avoid bottlenecks
- Meta data scalability
  - How can 10,000’s of nodes work on files in same folder
- Cluster recovery
  - If something fails, how can transparent recovery happen
- Management
  - Adding, removing, replacing, systems; data migration & backup

Outline of approach

- Critical review of existing techniques
- Extensive re-use of existing components
  - Linux file systems, like Ext2/3, JFS, XFS, ReiserFS
  - Networking: Portals from Sandia, TUX 0-copy ideas
  - Use page cache interfaces for 0 copy I/O
- Expect to contribute to the core kernel
  - To contribute a few refinements to VFS for cluster file systems
  - Storage networking and RPC interface

Ingredient 1: object storage

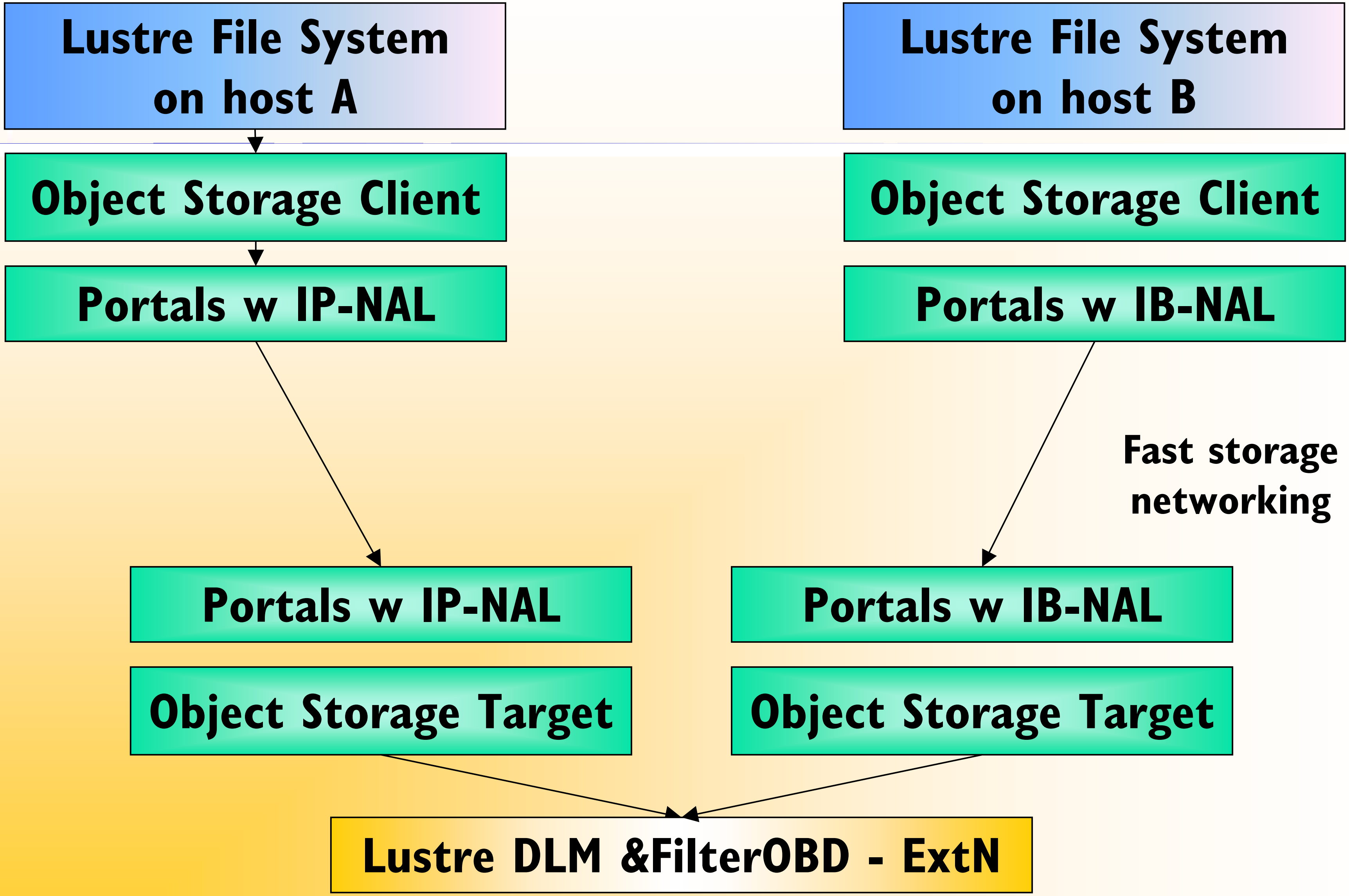
Cluster File Systems, Inc

What is Object Based Storage?

- Deal with “Objects”: think inodes/files (no file names)
  - More intelligent than block device
- Speak storage at “inode level”
  - create, unlink, read, write, getattr, setattr
  - iterators, security, almost arbitrary processing
- So…
  - Protocol allocates physical blocks, no names for files
- Requires
  - Management & security infrastructure

Components of OB Storage

- Storage Object Device Drivers
  - **class driver** – attach driver to interface
    - **Targets, clients** – remote access
    - **Direct drivers** – to manage physical storage
    - **Logical drivers** – for intelligence & storage management
- Object storage applications:
  - (cluster) file systems
  - Advanced storage: parallel I/O, snapshots
  - Specialized apps: caches, db’s, filesrv



Lustre File System on host A

Object Storage Client

Portals w IP-NAL

Lustre File System on host B

Object Storage Client

Portals w IB-NAL

Fast storage networking

Portals w IP-NAL

Object Storage Target

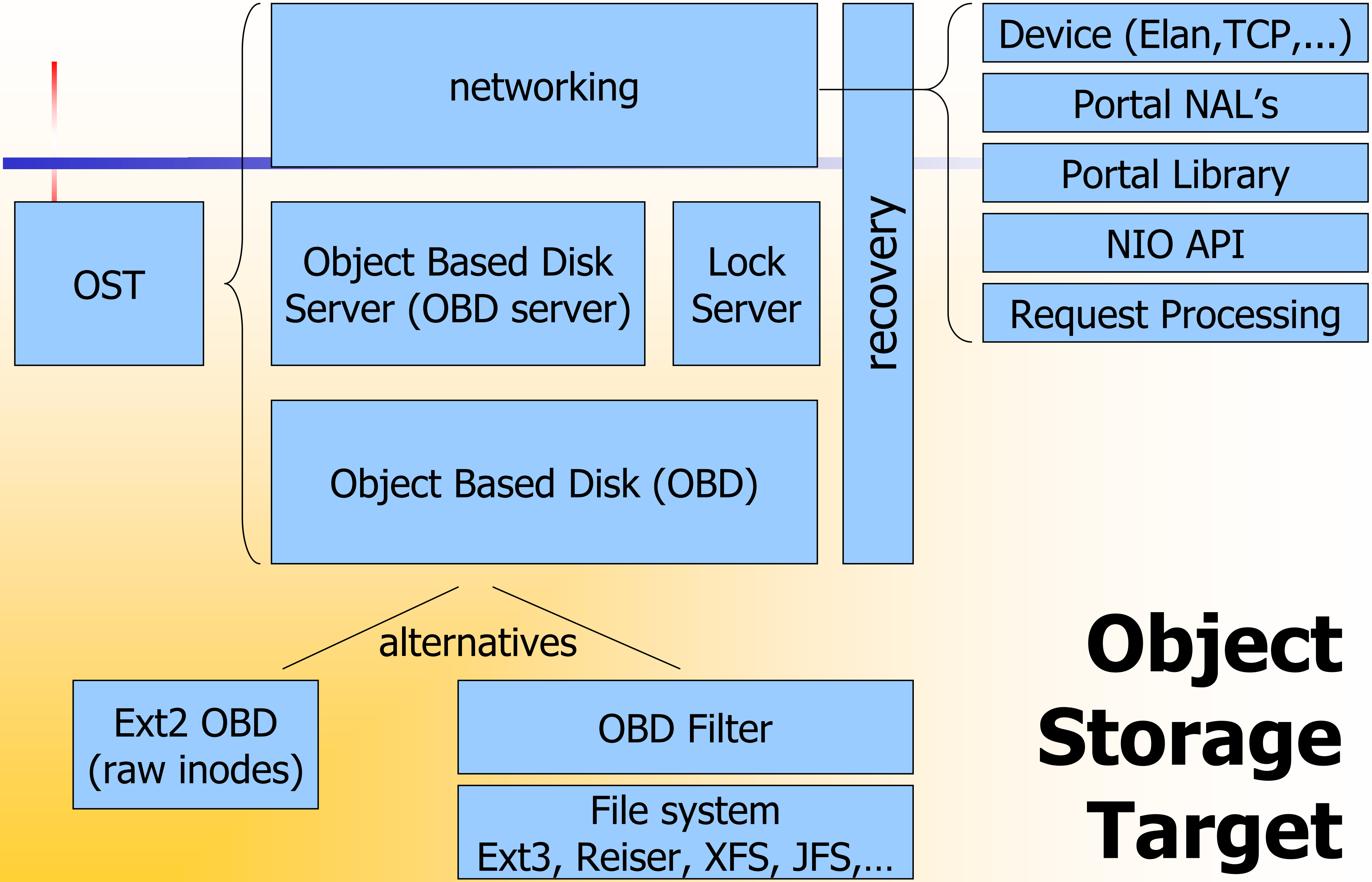
Portals w IB-NAL

Object Storage Target

Lustre DLM &FilterOBD - ExtN

Shared object storage

Object Storage Target



OST

networking

Object Based Disk Server (OBD server)

Lock Server

Object Based Disk (OBD)

recovery

Device (Elan,TCP,…)

Portal NAL's

Portal Library

NIO API

Request Processing

alternatives

Ext2 OBD (raw inodes)

OBD Filter

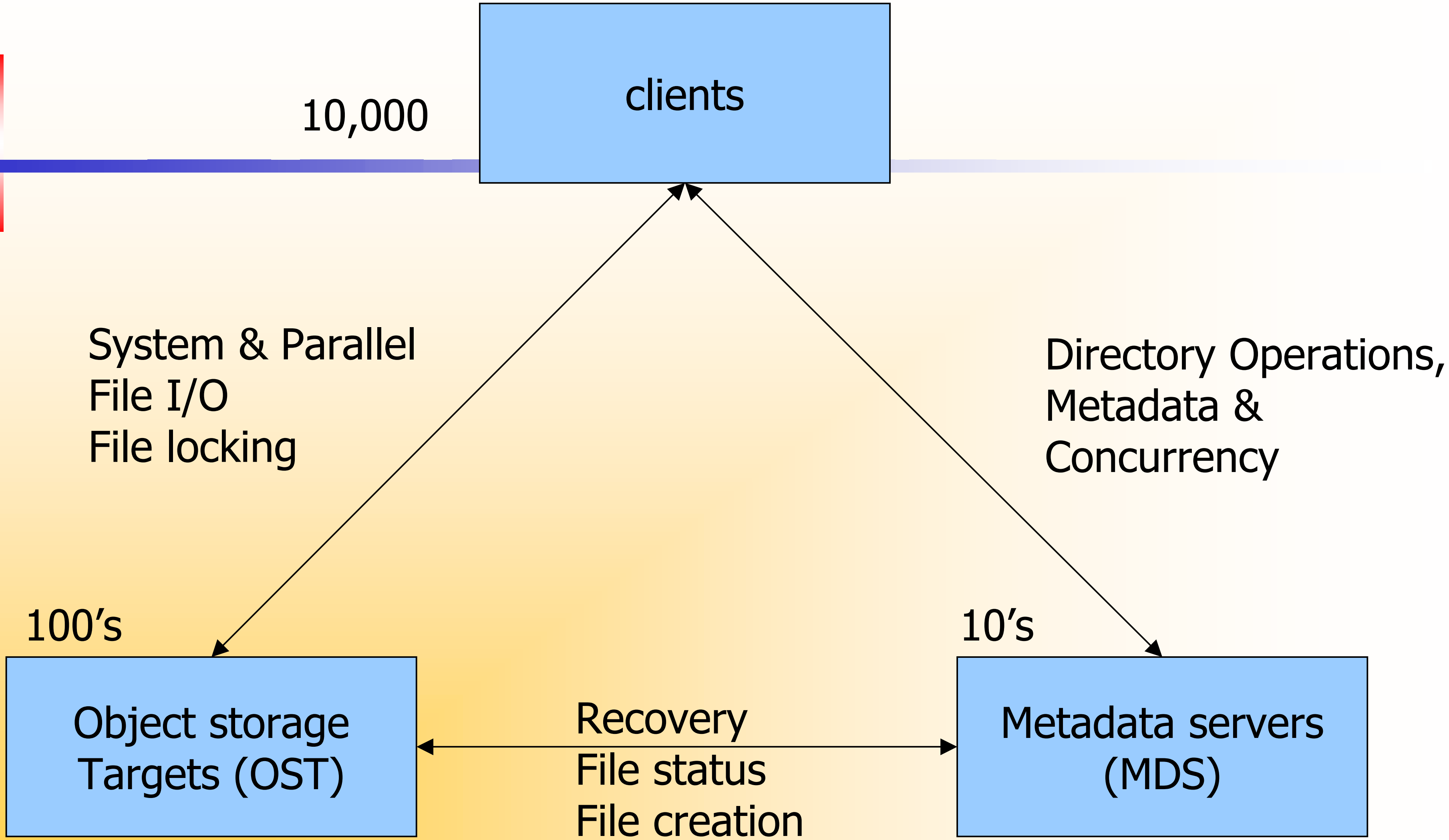
File system Ext3, Reiser, XFS, JFS,…

How does object storage help?

I/O bandwidth requirements

- Required: 100's GB/sec
- Consequences:
  - Saturate 100's – 1000's of storage controllers
  - Block allocation must be spread over cluster
  - Lock management must be spread over cluster
- This almost forces object storage controller approach

Lustre System



clients

10,000

System & Parallel
File I/O
File locking

Directory Operations,
Metadata &
Concurrency

100's

Object storage Targets (OST)

Recovery
File status
File creation

10's

Metadata servers (MDS)

File – I/O

- Open file on metadata system
- Get obtain information
  - What objects on what storage controllers store what part of the file
  - Striping pattern
- Establish connection to storage controllers you need
  - Do logical object writes to OST
  - From time to time OST updates MDS with new file sizes

Ingredient 2: Storage Management

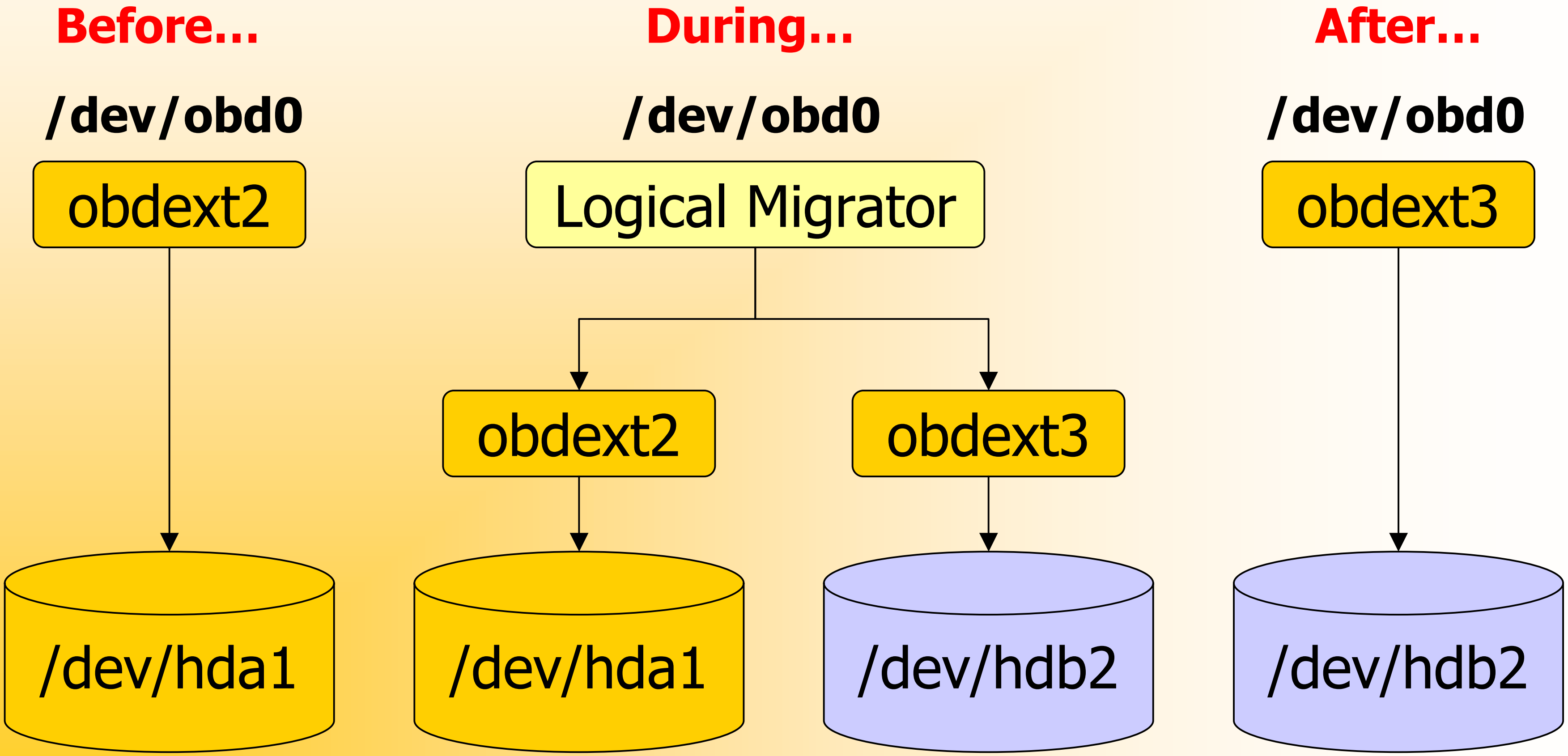
The cost of storage management routinely exceeds that of the hardware by 300%

Examples of logical modules

- Storage management:
  - System software, trusted
  - Often inside the standard data path,
  - also involves iterators
  - Eg: security, snapshots, versioning data migration, raid
- Lustre offers active disks
  - almost arbitrary intelligence can be loaded into OST driver stack

Example of management: hot data migration:

Key principle: dynamically switch object device types



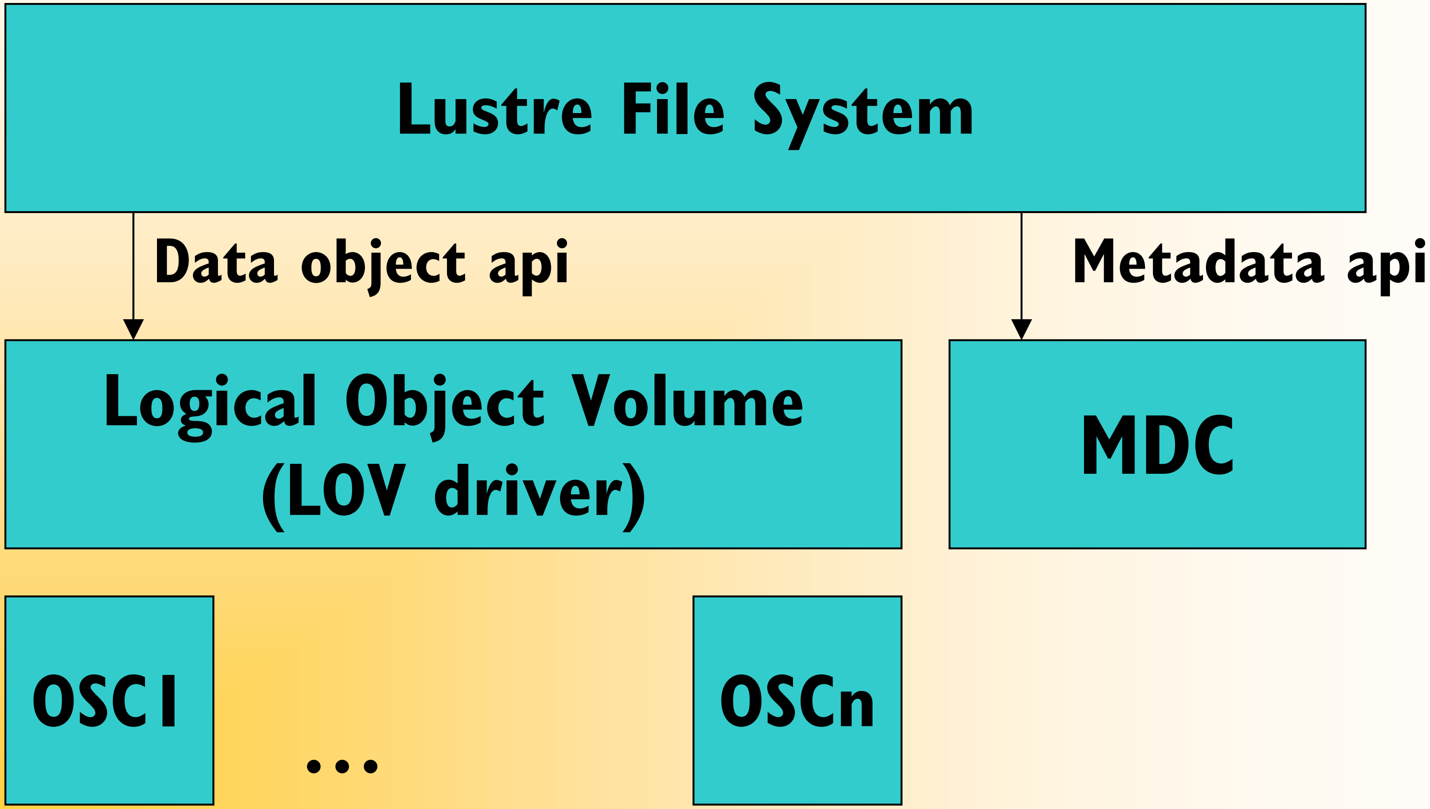
Cluster File Systems, Inc

Ingredient 3: metadata handling

Lustre File System

- Each file identified by an inode
  - inode stored on the MDS cluster
    - data for directories on MDS
    - data for file inode stored in data objects on OST's
- File inode metadata
  - Includes data object descriptor in extended attribute
  - Stored on MDS
  - Includes: striping descriptor and object id's

Stripes



Lustre File System

Data object api

Metadata api

Logical Object Volume (LOV driver)

MDC

OSC1

…

OSCn

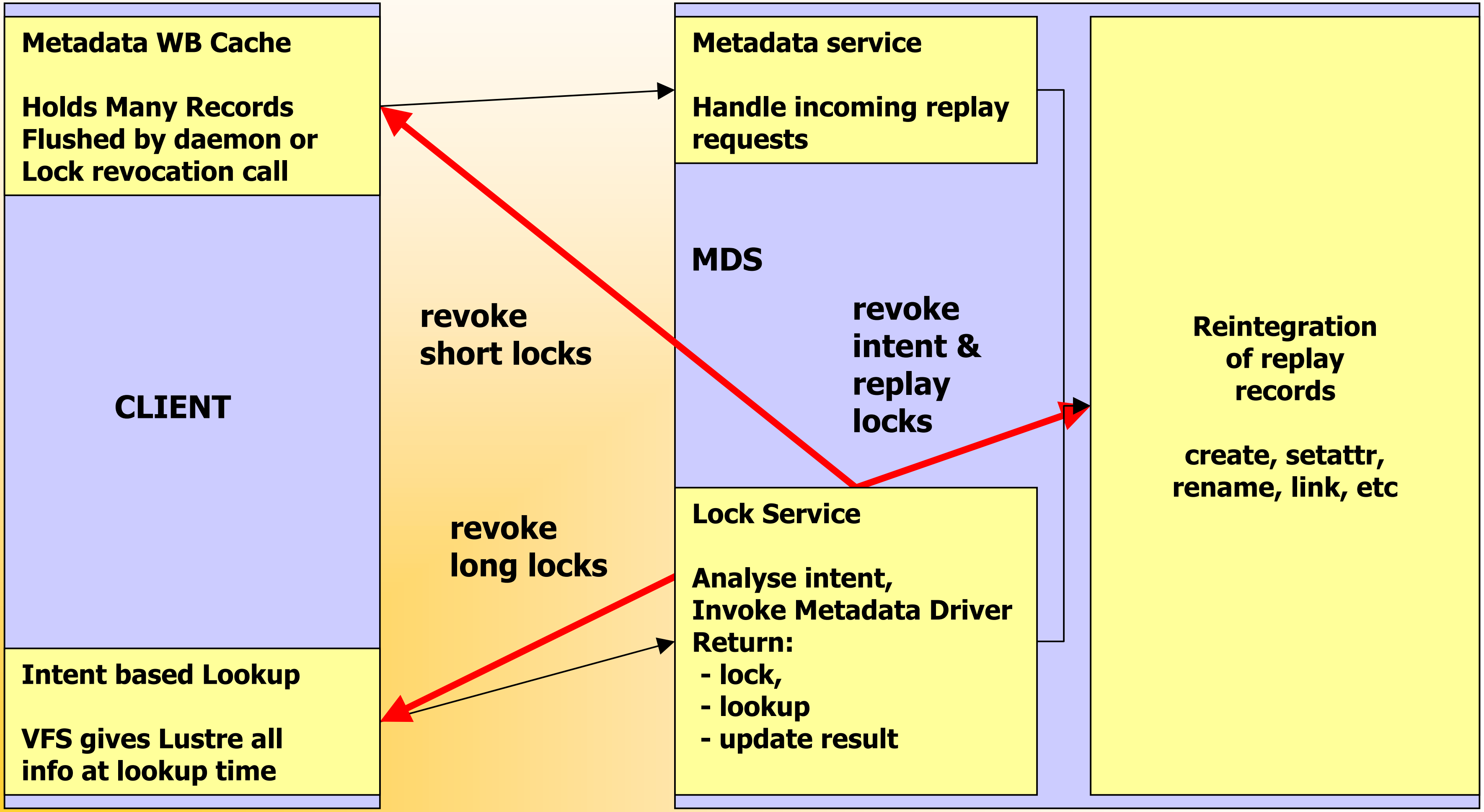
Intent based locks & Write Back caching

- Protocol adaptation between clients and MDS
- Low concurrency - write back caching
  - On client in memory updates with delayed replay on MDS
- High concurrency
  - Single network request per transaction, no lock revocations
  - Intent based locks – lock includes all info to complete transaction

Two types of metadata locks:

- Long locks –
  - Lock whole pathname, help with concurrency
  - e.g. locking the root directory is BAD
    - so lock /home/peter & /home/phil separately
- Short Locks
  - Lock a directory subtree -help for delegation
  - e.g. a single lock on /home/phil is GOOD

Metadata updates



Metadata WB Cache

Holds Many Records
Flushed by daemon or
Lock revocation call

CLIENT

Intent based Lookup

VFS gives Lustre all
info at lookup time

revoke
short locks

revoke
long locks

Metadata service

Handle incoming replay
requests

MDS

revoke
intent &
replay
locks

Lock Service

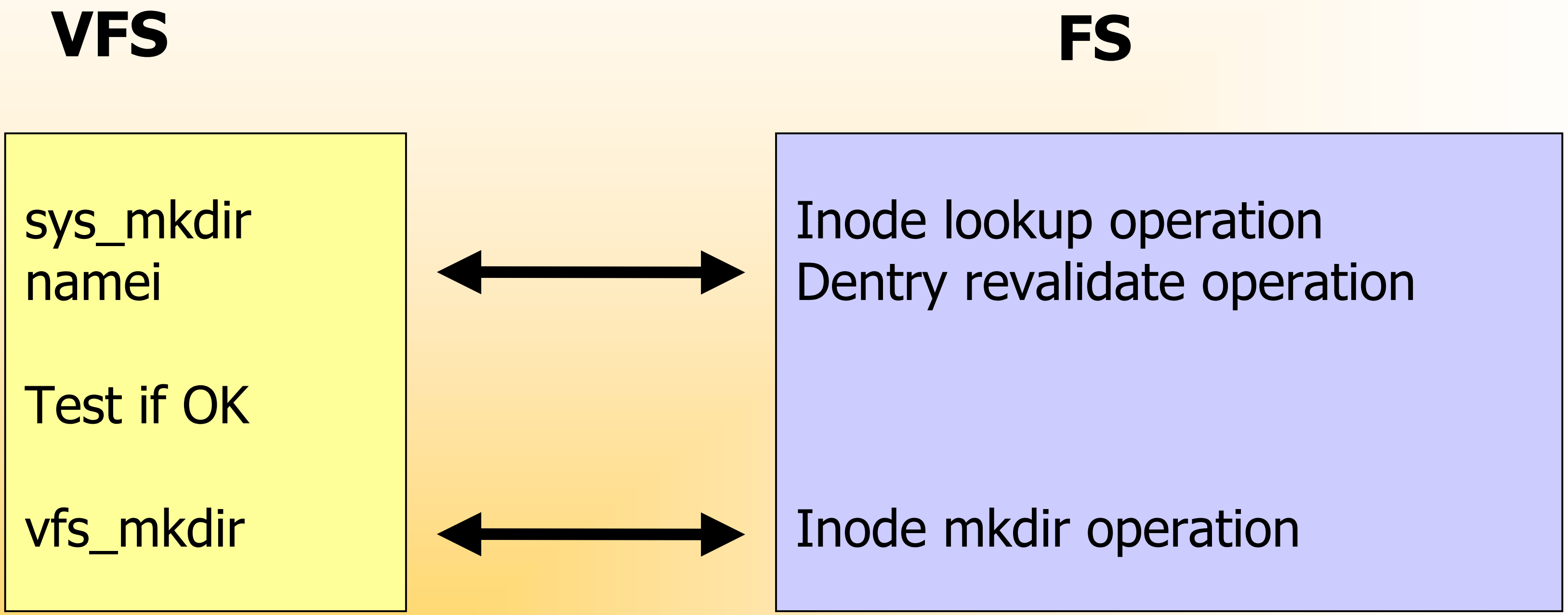
Analyse intent,
Invoke Metadata Driver
Return:
- lock,
- lookup
- update result

Reintegration
of replay
records

create, setattr,
rename, link, etc

Cluster File Systems, Inc

Current Linux VFS



VFS

FS

sys_mkdir
namei

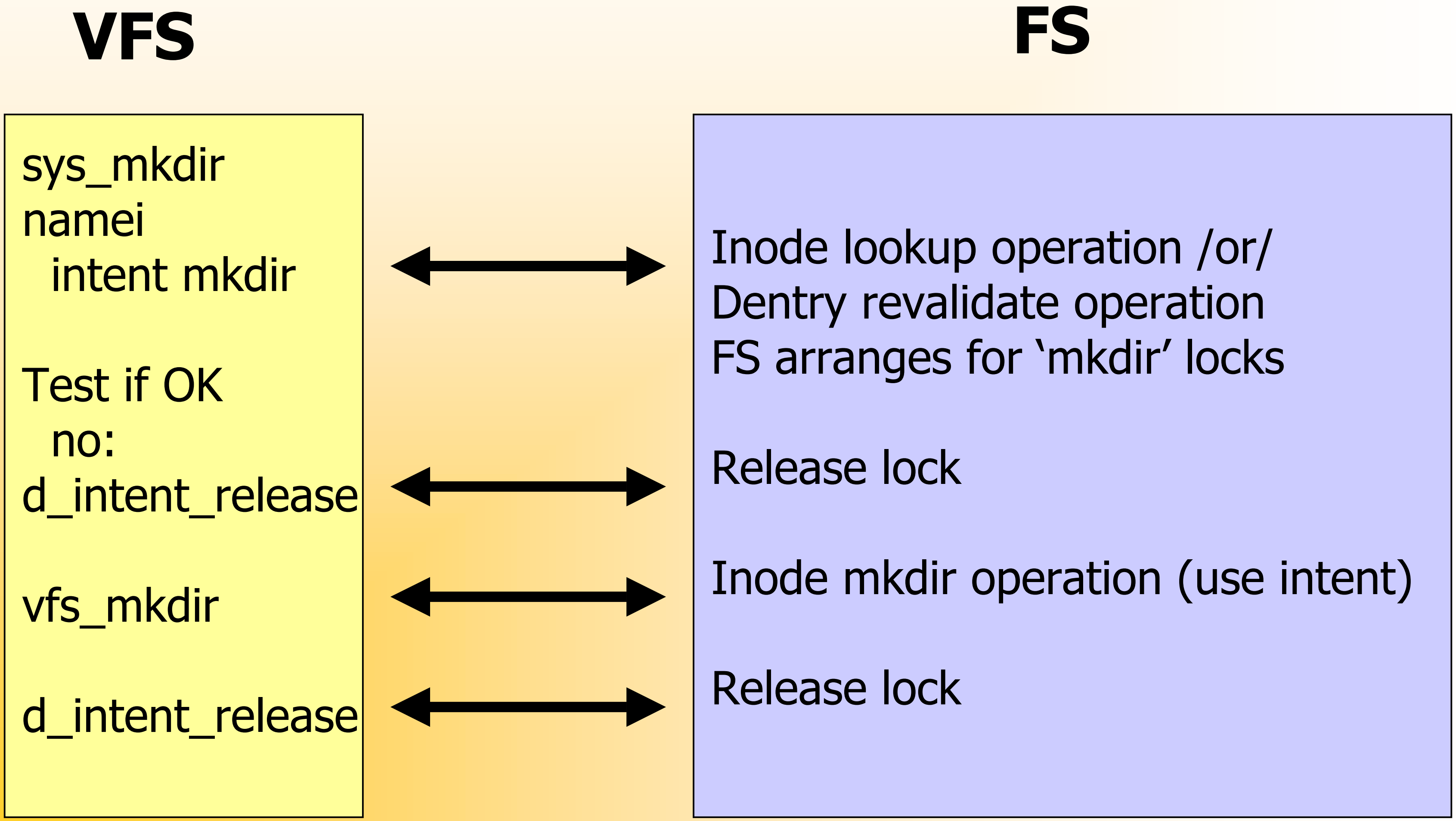
Test if OK

vfs_mkdir

Inode lookup operation
Dentry revalidate operation

Inode mkdir operation

We added “intents” to lookups



VFS

FS

sys_mkdir
namei
  intent mkdir

Test if OK
  no:
d_intent_release

vfs_mkdir

d_intent_release

Inode lookup operation /or/
Dentry revalidate operation
FS arranges for ‘mkdir’ locks

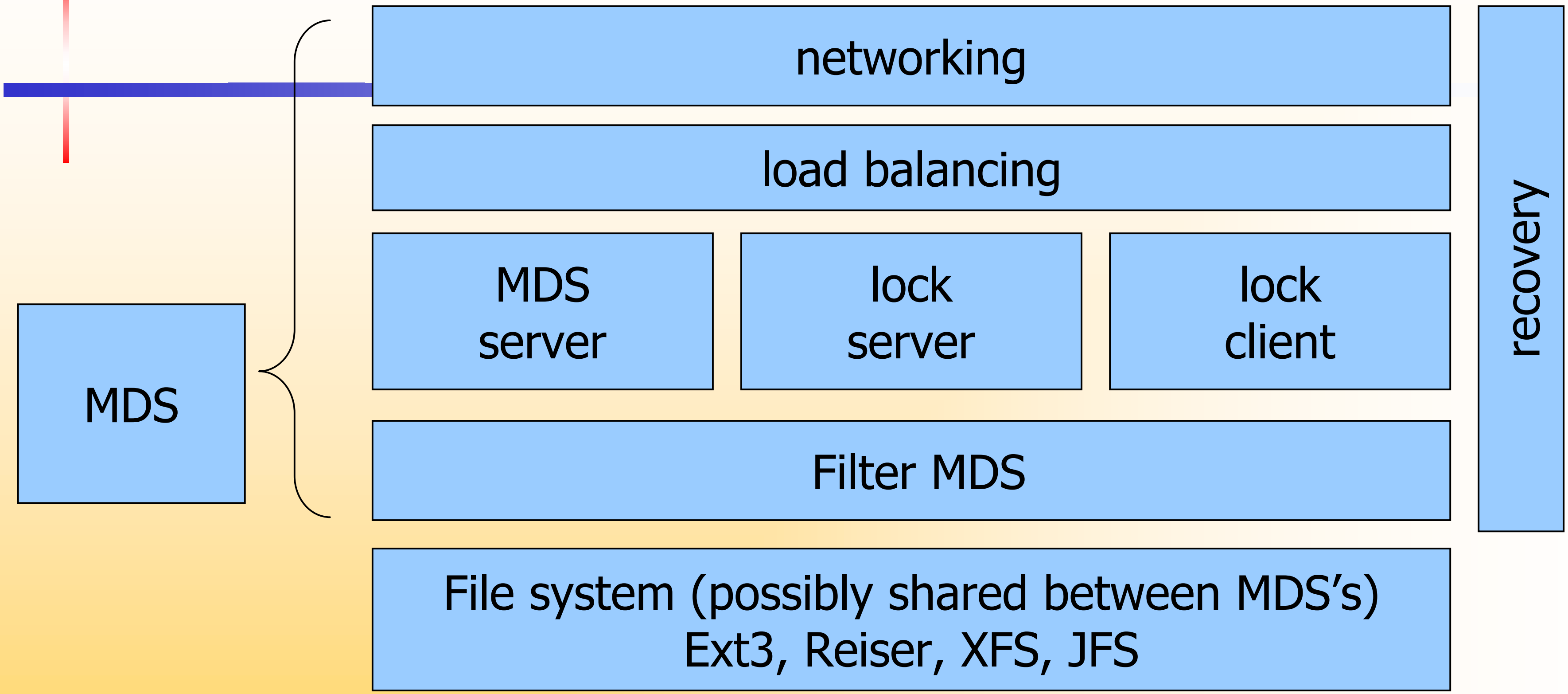
Release lock

Inode mkdir operation (use intent)

Release lock

Subdivision of metadata across cluster

- Directories:
  - hash by name
  - assign hash values to MDS cluster nodes
- Inodes:
  - Assign 16GB ext3 block groups to MDS cluster nodes
- Result:
  - many ops can proceed in parallel
  - Journaled metadata file system at the core



Metadata Server

Recovery

- Client – MDS updates
  - Deals with lost replies, requests & disk updates
  - Replay mechanism
- Locks
  - Forcefully revoke locks from dead clients
  - Re-establish existing locks with recovering services
- Recovery Interaction with storage targets
  - Preallocation of objects
  - Orphaned inodes and data objects

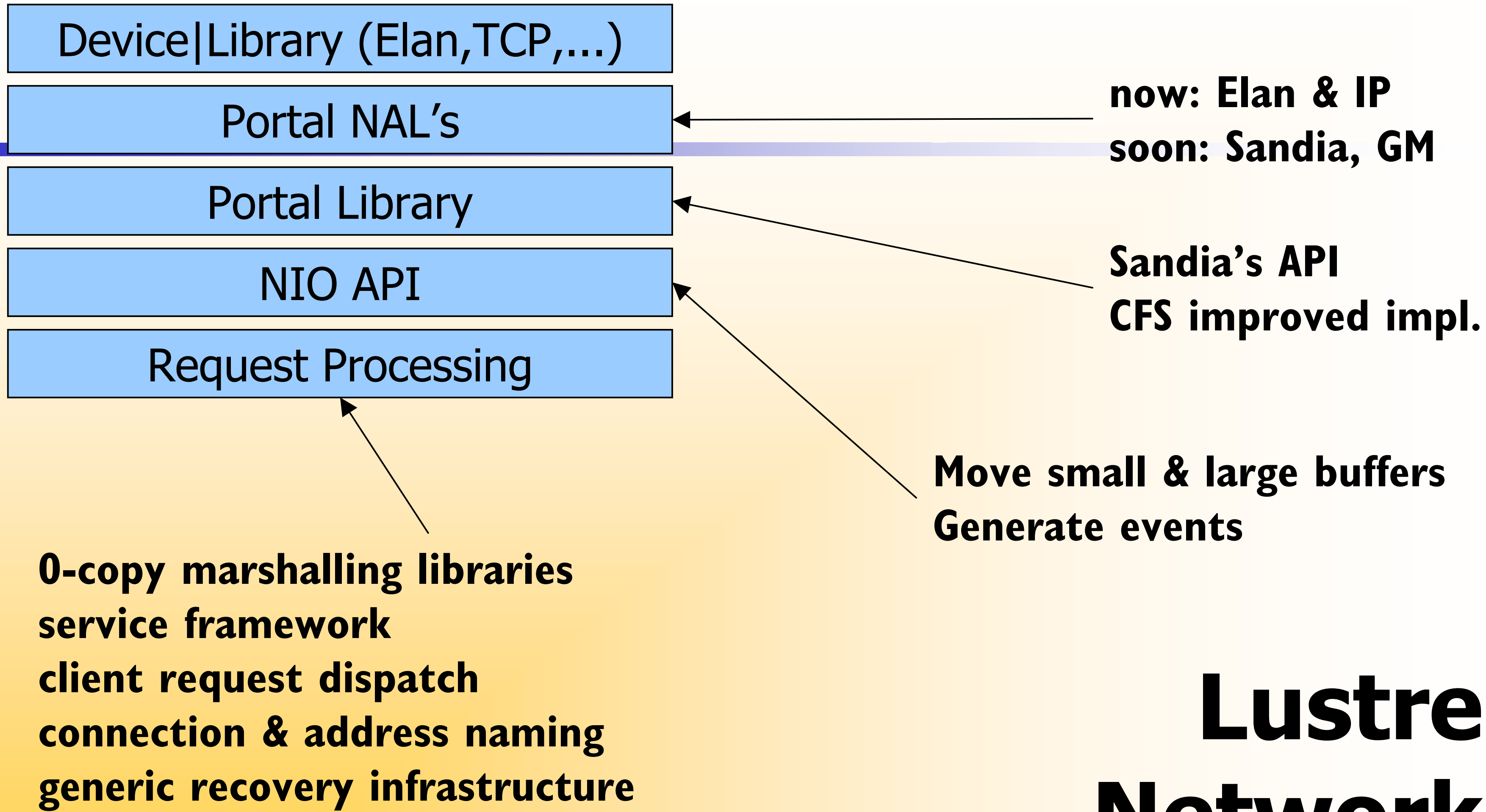
Metadata odds and ends

Logical Metadata Drivers

- We have not forgotten about:
  - Local persistent metadata cache, like AFS/Coda/InterMezzo
  - Replicated metadata server driver
  - Remotely mirrored MDS

Ingredient 4: Storage Networking

Lustre Network Stack



Device|Library (Elan,TCP,...)

Portal NAL's

Portal Library

NIO API

Request Processing

now: Elan & IP
soon: Sandia, GM

Sandia's API
CFS improved impl.

Move small & large buffers
Generate events

0-copy marshalling libraries
service framework
client request dispatch
connection & address naming
generic recovery infrastructure

Lustre networking

- Currently runs over
  - TCP,
  - Quadrics
  - Myrinet (almost)
- Other networks we are looking at:
  - SAN's
  - I/B
  - NUMA interconnects (@ GB/sec)
  - SCTP

Portals

- Sandia Portals message passing
  - simple message passing API
  - support for remote DMA
  - Network Abstraction Layers: pluggable device support

Initial network performance figures

- IP
  - Server throughput 40,000 requests/sec
  - Data movement 110MB/sec over Gige
  - Single client up to 45MB/sec
- Quadrics Software Elan3
  - Server throughput 20,000 requests/sec
  - 240 MB/sec bulk movement
- Tested up to 25 nodes, 6,000 client threads
- We have no definitive answers on best design of the NALs yet

File I/O

- Single client
  - Quadrics 80MB/sec
  - Gige 40MB/sec
- Cluster: should scale nicely, not measured yet

The real world…

Lustre & SAN's

- From the galaxy to a 4 node Linux cluster
- Exploit SAN's – retain OST/MDS
  - TCP/IP: to allocate blocks, do metadata
  - SAN: for file data movement
- Shared ext3 file system
  - Merge MDS & OST: export one file system

Project status

Lustre Mandatory Features

Lustre Lite	**Lustre Lite Performance**	**Lustre**
2002	2003	2004
Single Failover MDS	Metadata cluster	Metadata cluster
Basic Unix security	Basic Unix security	Advanced Security
		Storage management
Intent based metadata	Writeback metadata	Load balanced MD
	Parallel I/O	
POSIX compliant		Global namespace

Cluster File Systems

- Small scale service company
  - contract work for Government labs (all OSS but defense contracts)
  - some consulting and collaboration with industry
- Extremely specialized and extreme expertise
  - we only do file systems and storage
- Investments etc
  - Please visit “Save the Children”
  - no thank you – it’s perfectly possible to go forward without