



The Evolution of Object-Based Storage to Enable Large Scale Distributed Research Computing

Jacob Farmer, CTO
Cambridge Computer

- My Definition of Object
 - A Broad Definition for Object-based storage
 - Object storage is enabled by “Big Data” – “Big Metadata”
- How We Got Here: The Evolution of Storage Networking
- The Original Object Stores: OST-T10, Lustre, Panasas
- Object Stores Can Serve Any Layer of the Storage Stack
- Content Addressing
- Cloud-like Object Stores for High Capacity and Resiliency
 - Petabyte-scale growing pains
 - Geographic distribution of data
 - Object Mirroring and Erasure Codes
- Layering Object Models

Objects Represent a Different Way to Address Data



Block	Blocks are addressed by Device ID and sequential block number.
File	Files are addressed by UNC paths: <code>\\MyServer\MyFolder\MyFile.doc</code>
Object	Objects are addressed by an ID that is unique to the storage system. <ul style="list-style-type: none">- Sequentially assigned number- Randomly assigned number- A hash derived as a function of the objects content- A combination of things



What is an "Object"



- An object is a unit of data that can be individually addressed and manipulated
- Ideally an object should not be bound to a physical location
 - Each object has a unique logical address
 - The object might exist in multiple locations for accessibility or redundancy
 - An object might be *content addressable* meaning that the address of the object is derived from a hash of the objects data payload
- Objects might have metadata associated with them
 - Metadata refers to additional descriptive properties that give additional meaning to the object



Examples of Objects – Any Discernable Chunk of Data



• Email

- An email message is a chunk of data
 - An email attachment is a chunk of data
 - An email message along with its attachments could be treated as a single chunk of data.
- A file could be an object
 - A zip file containing many files could be an object
 - A file can be made up of several chunks of data, each of which could be an object
 - A block could be stored as an object
 - A grouping of blocks, making up a disk array volume
 - Chunk, chunklet, page, extent





CAMBRIDGE
Computer
ARTISTS IN DATA STORAGE

Big Metadata

- Storage systems typically require a lot of metadata
 - POSIX file systems – metadata is the limiting reagent
 - RAID /disk systems – typically use sequential addresses which simplify metadata operations
- Object stores require metadata to navigate and decipher
 - Big object stores need a lot of metadata
- Big, fast metadata is enabled by
 - Solid state storage
 - Modern database technologies
 - Many of which are arguably object stores in and of themselves

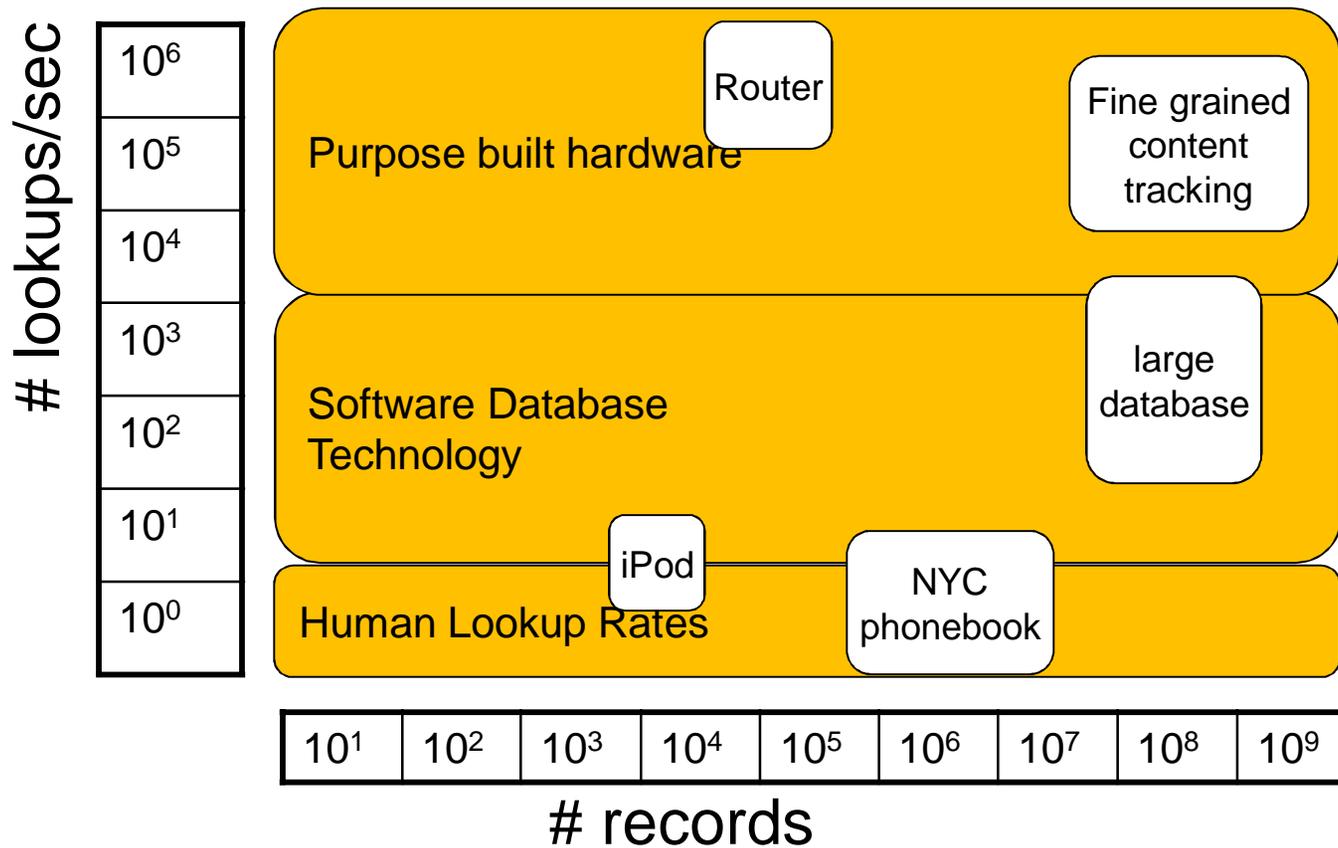
Indexing Small Segments with Large Capacities is a Real Challenge



- There are 2.5 Billion 4K segments in a Terabyte
- There are 250 Billion 4K segments in 100 Terabytes
- If you are doing random I/O processing in an enterprise SAN you could easily be demanding 20,000 IOPS.
- What kind of database can perform 20,000 lookups per second across 250 billion records?
- That is a significant indexing challenge!
 - Of course, no one says you have to do use 4K objects or that you can't do some sequential addressing



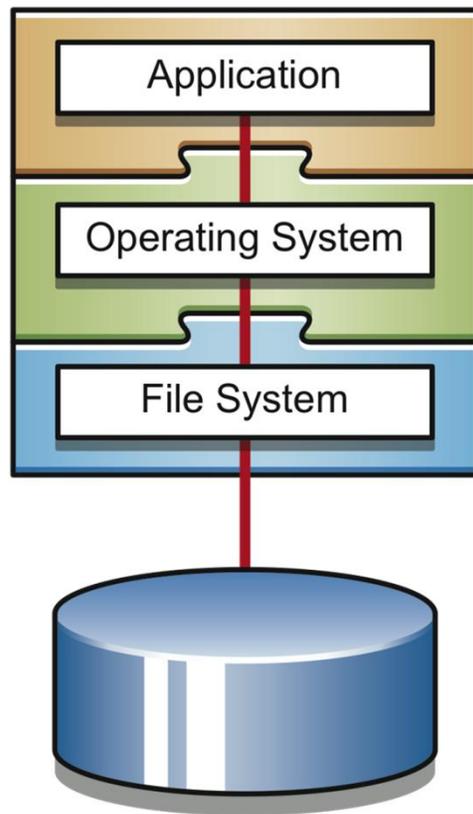
Dedupe Indexing Compared with Other Common Indexes





CAMBRIDGE
Computer
ARTISTS IN DATA STORAGE

The Evolution of Storage: The Traditional Storage Stack



Storage has a layered architecture, very much like a network stack.

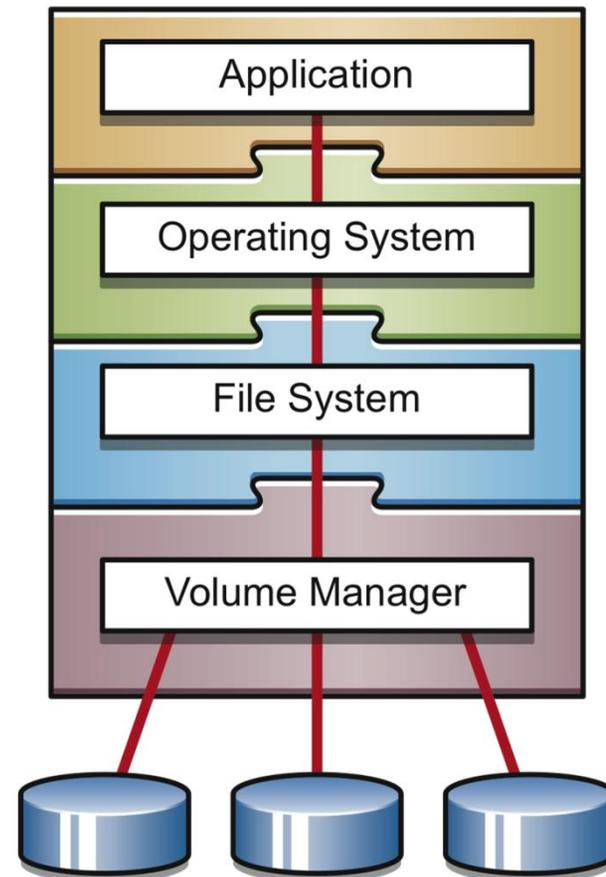
Disk drives store data in blocks. Each block has a unique numerical address.

Disk devices (hard drives, RAID systems, etc.) are like “block servers”, meaning you ask them to perform operations on specific blocks.

Block-Level Abstraction in a Logical Volume Manager (LVM)

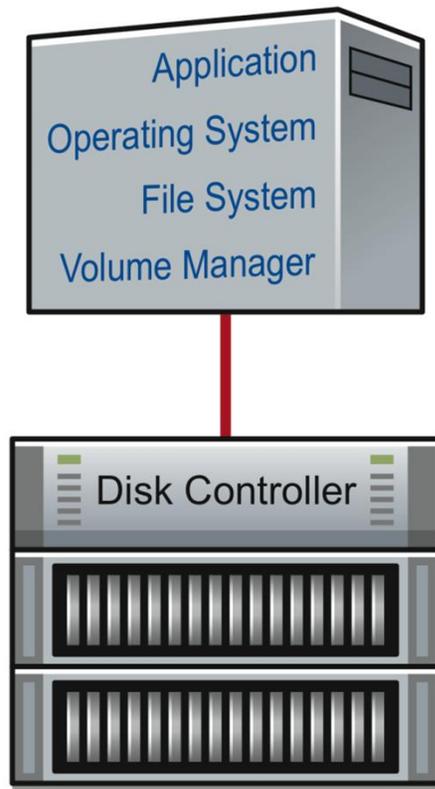
Abstraction of the physical disk

- Software RAID
 - Hard Drive Fault Tolerance
 - Spindle Aggregation
- Solid State Disk Caching
- Host-based mirroring
- Proxy Backups
- Volume-level snapshots
- Volume-level replication

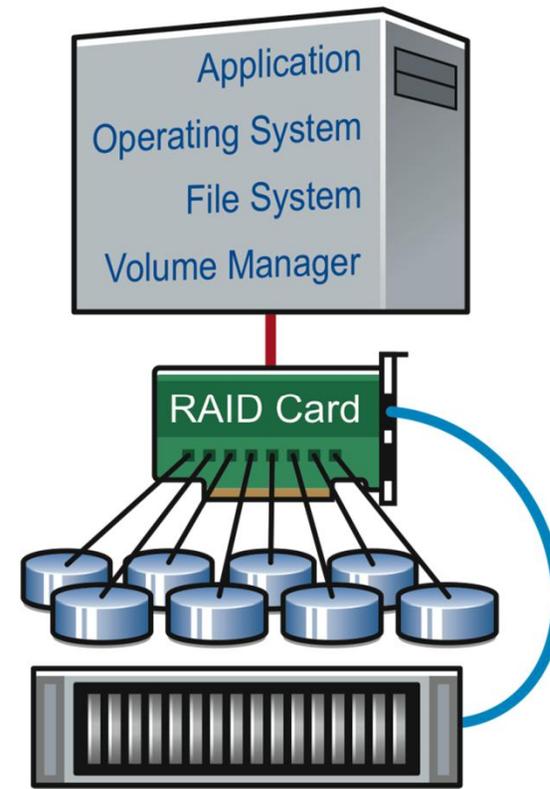


RAID Arrays = Hardware-Enabled Block Abstraction

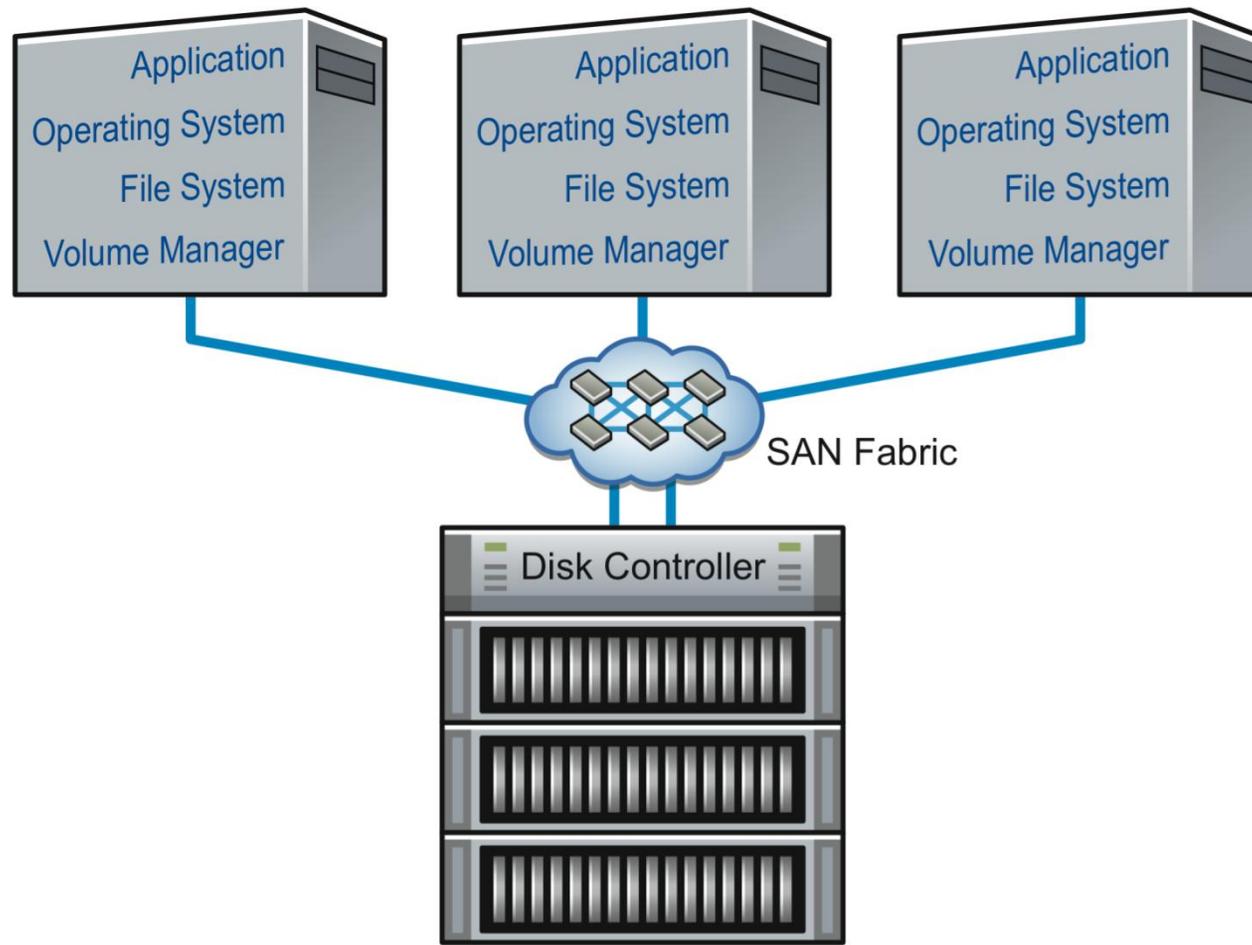
External Array



Internal Array



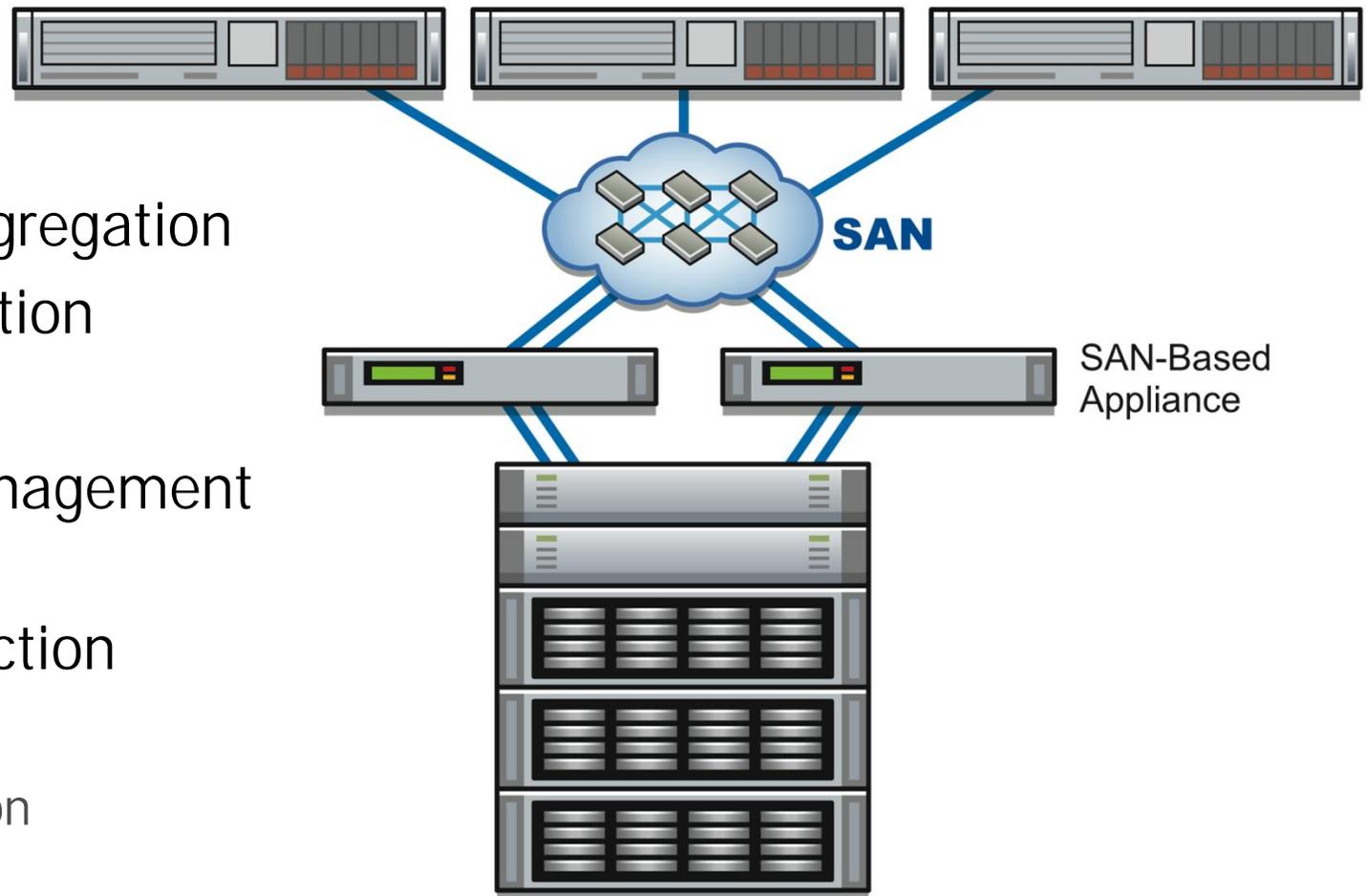
SAN Array = Centralized, External Block Abstraction



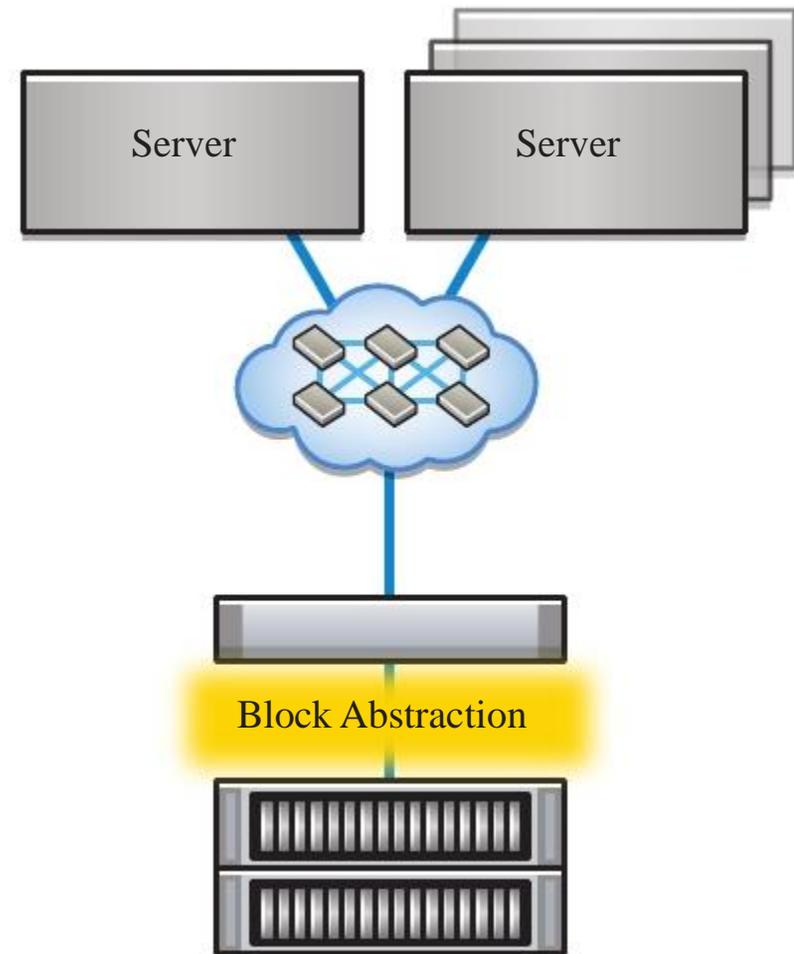
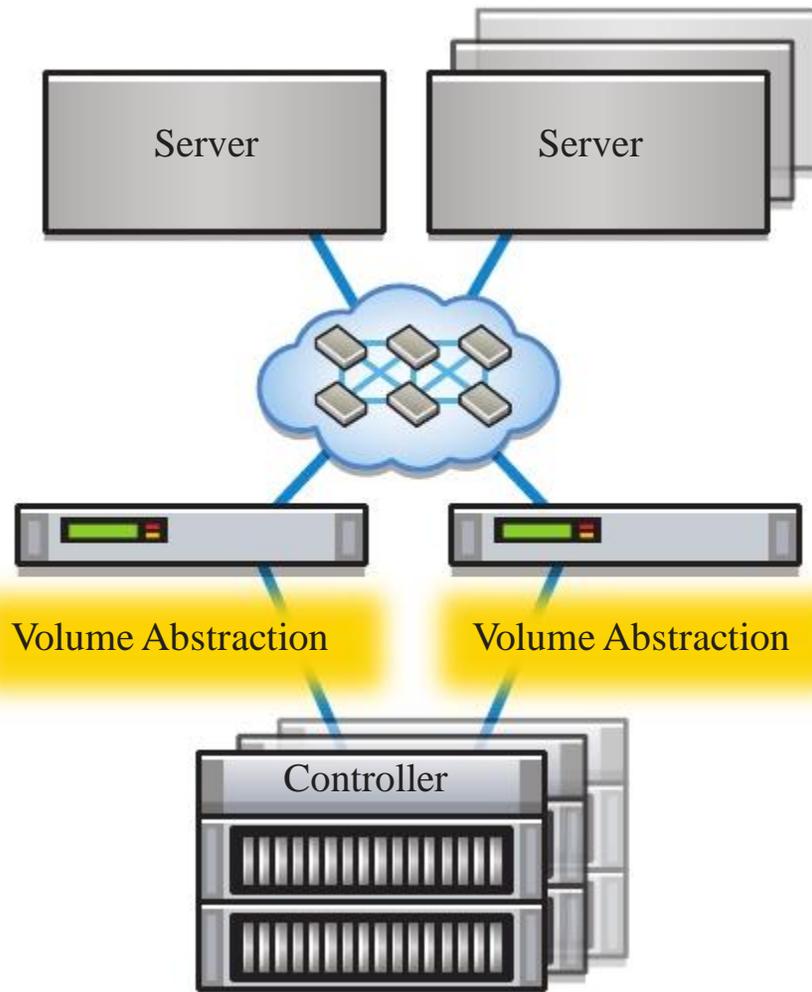
In-Band SAN Volume Virtualization



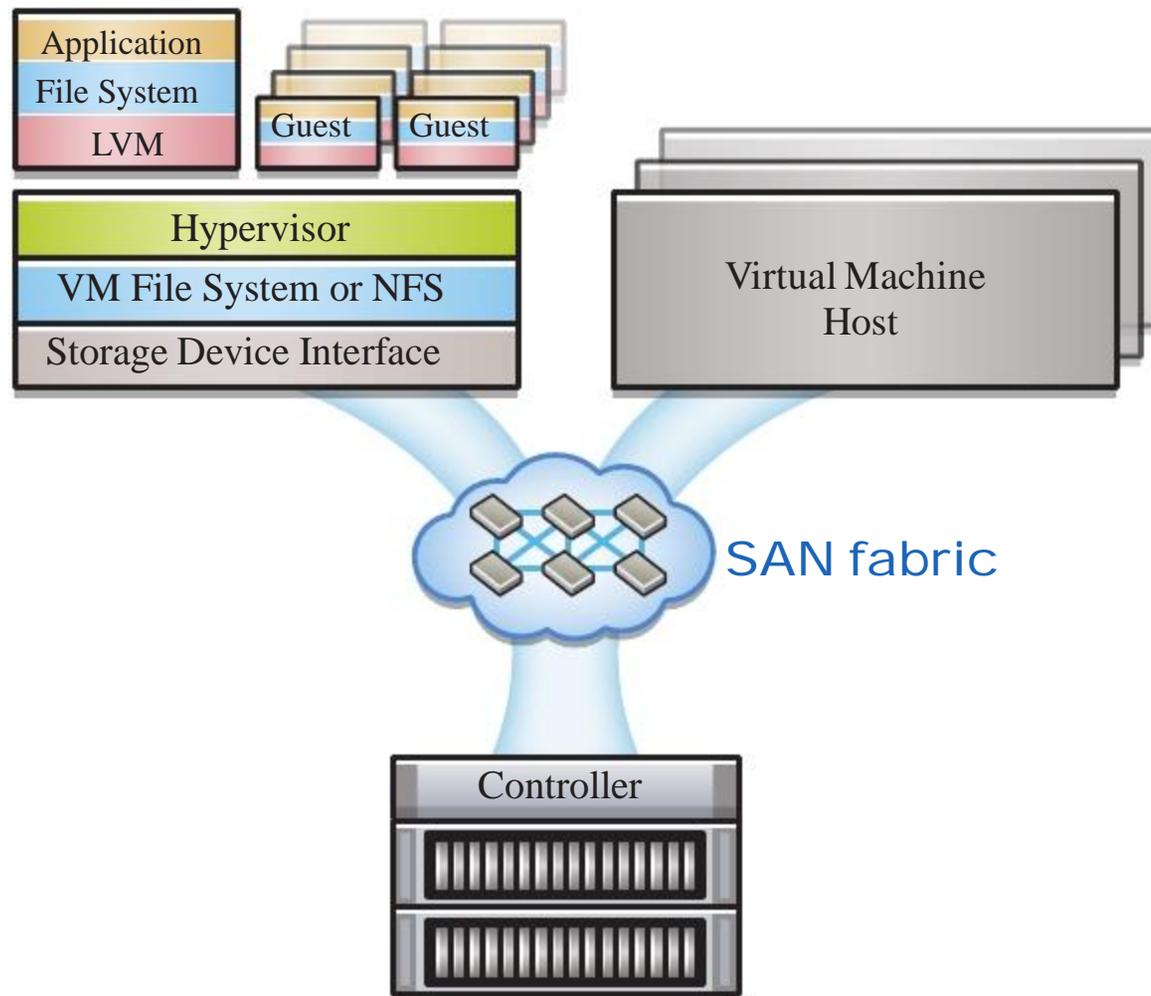
- Spindle Aggregation
- Data Migration
- Caching
- Unified Management Console
- Data Protection
 - Snapshot
 - Replication
 - CDP



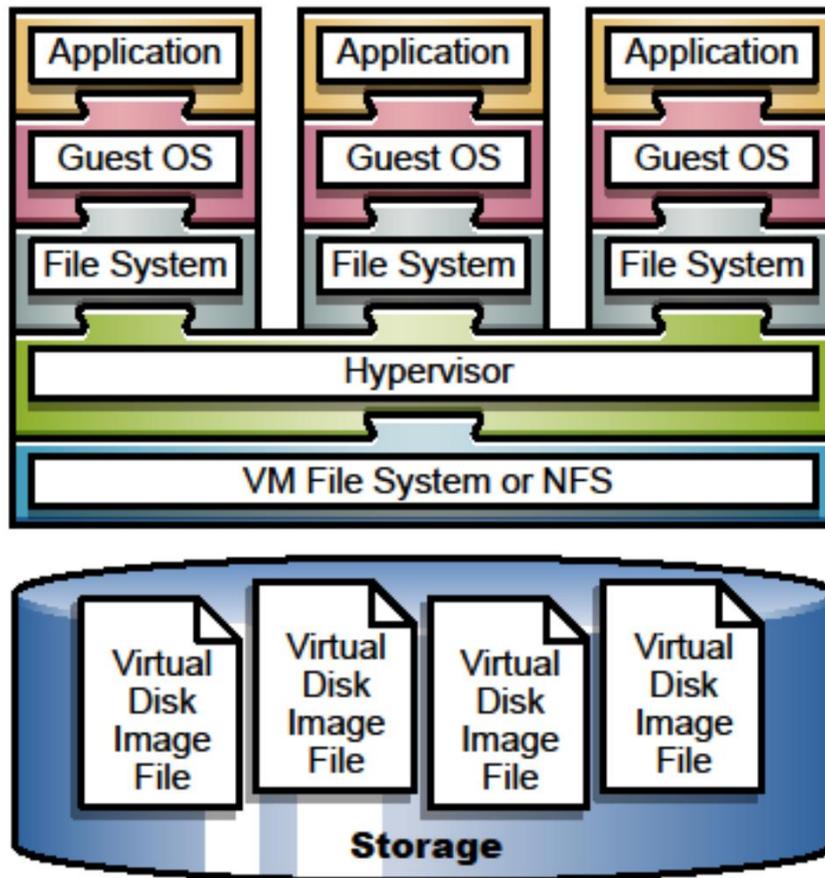
“Virtualization in the SAN” versus “Virtualization in the Array”



The Hypervisor Sits Directly In the Storage I/O Path

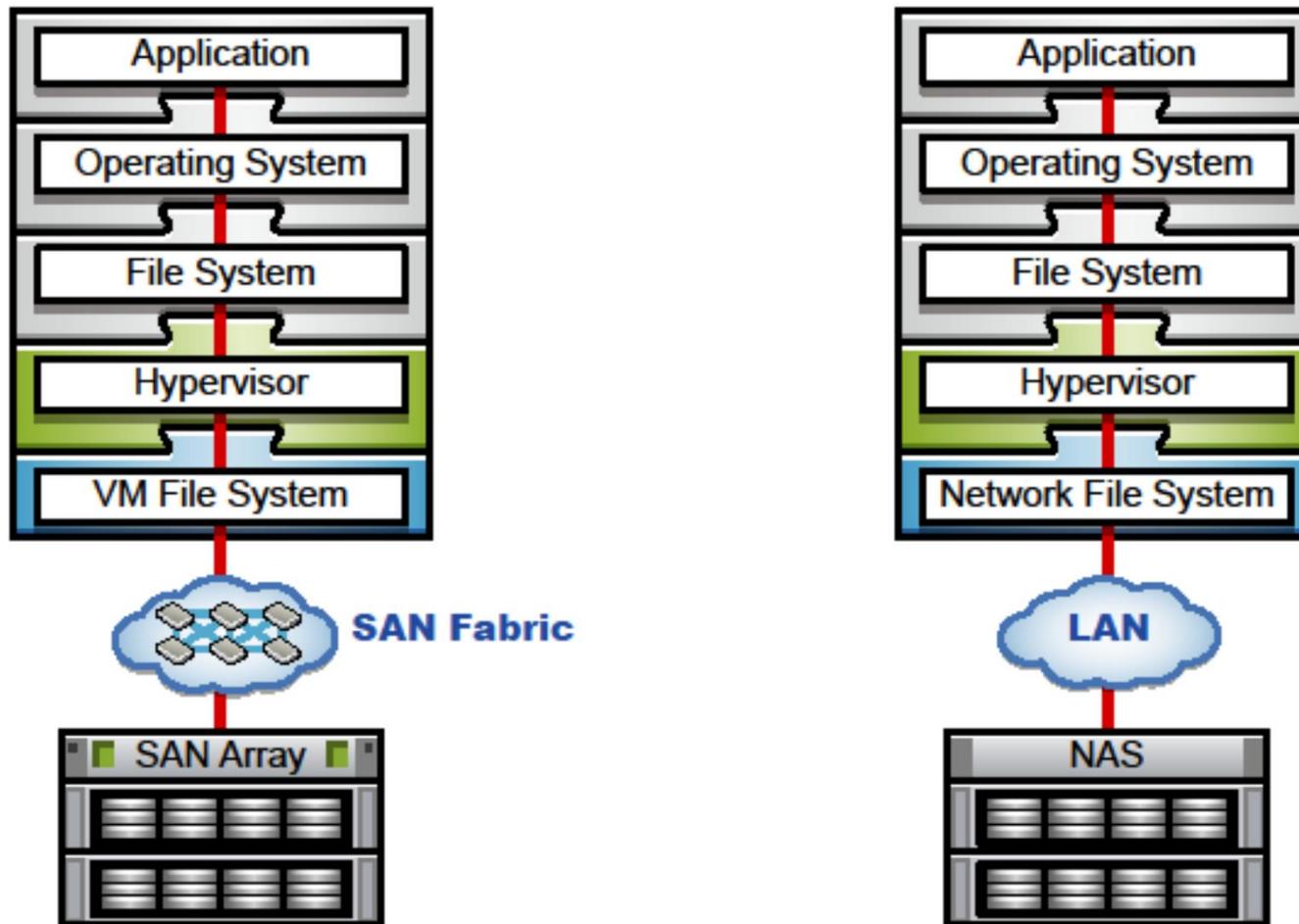


The Hypervisor is a Storage Virtualization Layer

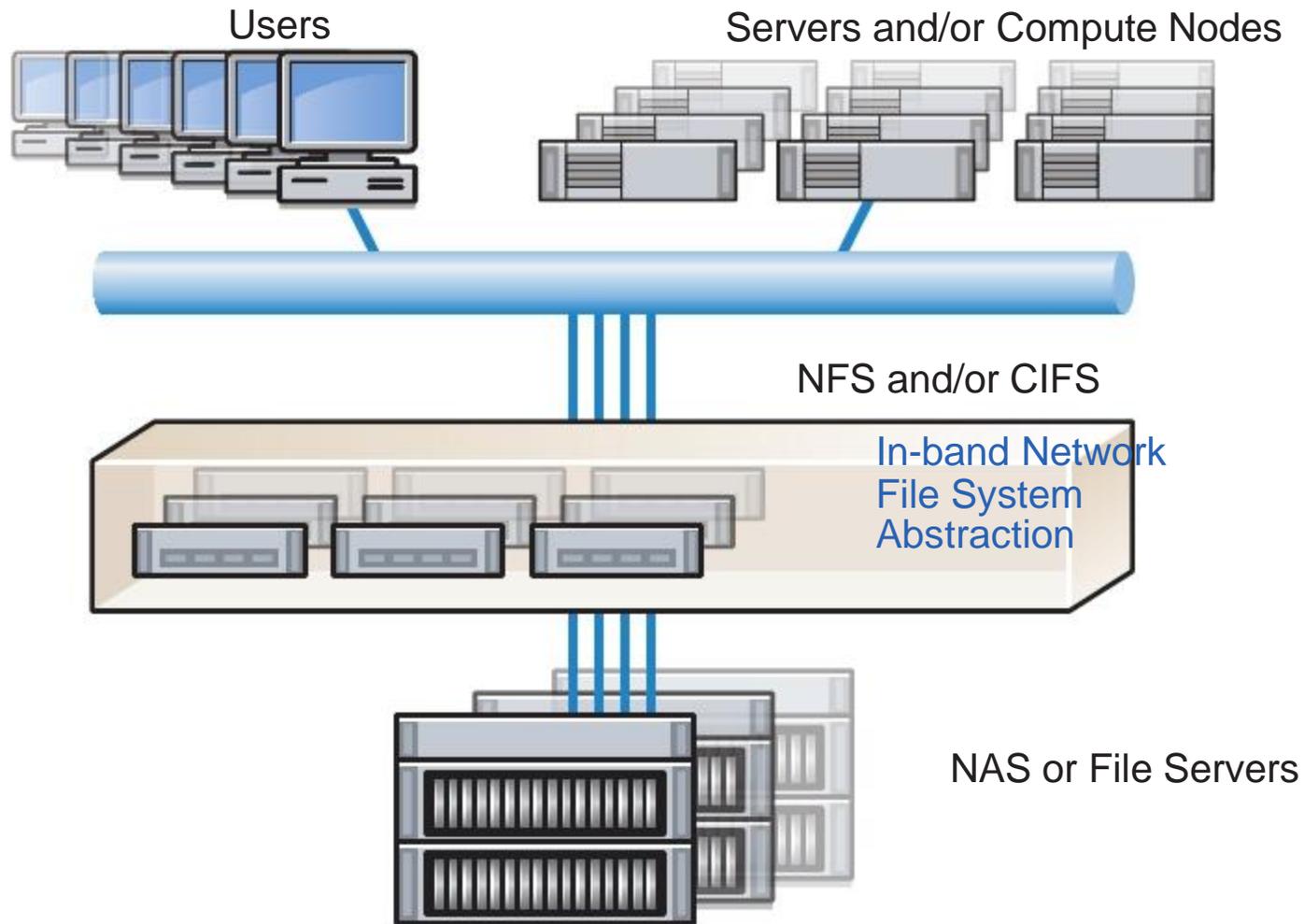


- Logical disk volumes are stored as single files
 - Non-disruptive storage migration
 - Automated site recovery
 - Granular backups and replication based on snapshot deltas

SAN or NAS: The Guests Don't Know or Care



In Band File System Abstraction



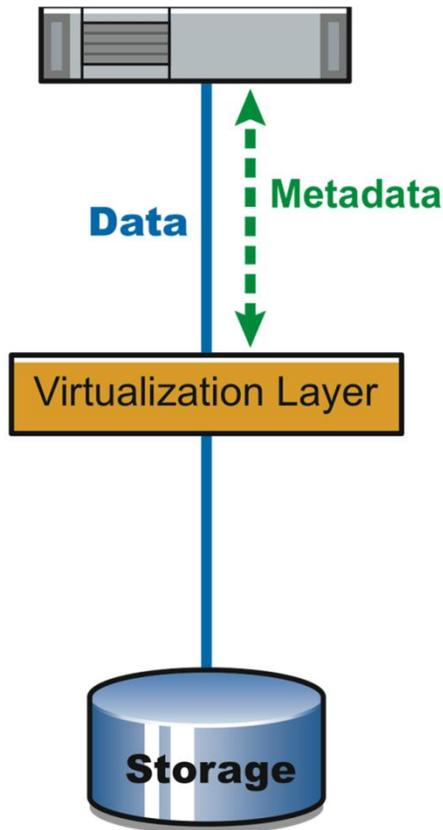
LAN-Resident Storage Abstraction



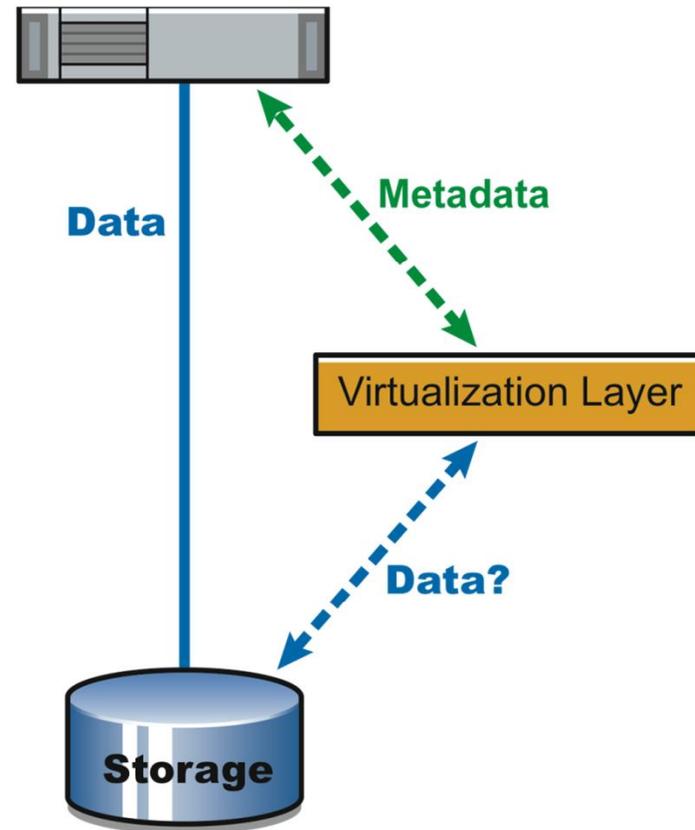
- In the last few years a number of devices have come to market that sit in-line between file servers and their clients, offering features like:
 - WAN Acceleration
 - Auditing & Encryption
 - Capacity Optimization
 - Compression
 - De-Duplication
 - Application Failover
 - Global Name Space
 - Tiered Storage
 - Caching
- File management logic can also sit out-of-band on the IP network.



Storage Abstraction Logic Can Sit In-band or Out-of-band

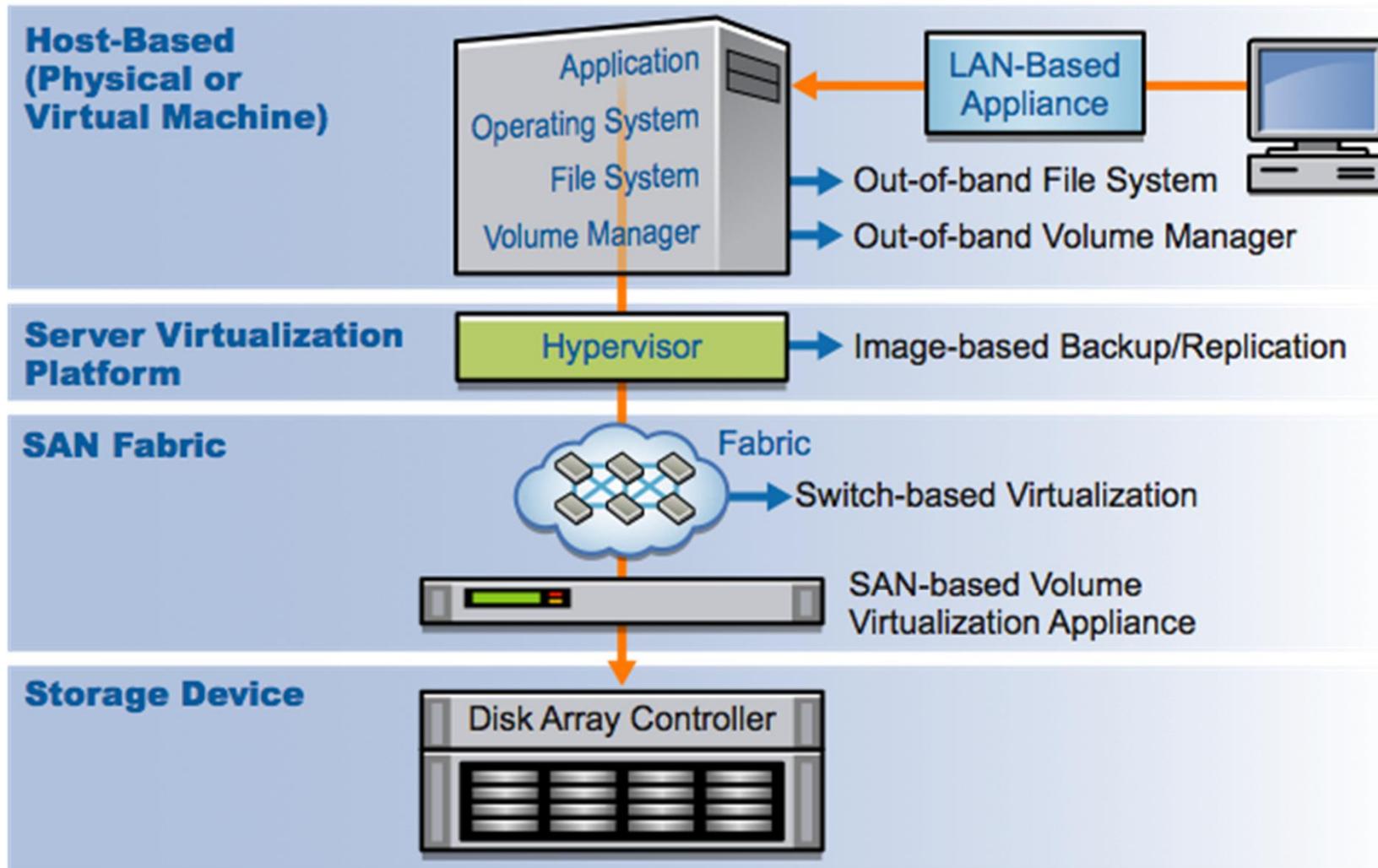


In-band Virtualization



Out-of-Band Virtualization

Fully Expanded Storage I/O Path



Further Evolution of Storage will Happen in One of Two ways



- People are going to figure out new ways to insert clever new logic into the storage I/O path
 - Hot areas of innovation include:
 - Abstraction of physical hard drives
 - File systems
 - Devices on the LAN sitting between file servers (NAS) and clients
- Collapsing of some or all layers of the traditional “storage stack” into more intelligent components
 - Intelligent file systems like ZFS or OneFS
 - Object storage models based on flexible addressing and/or content addressing



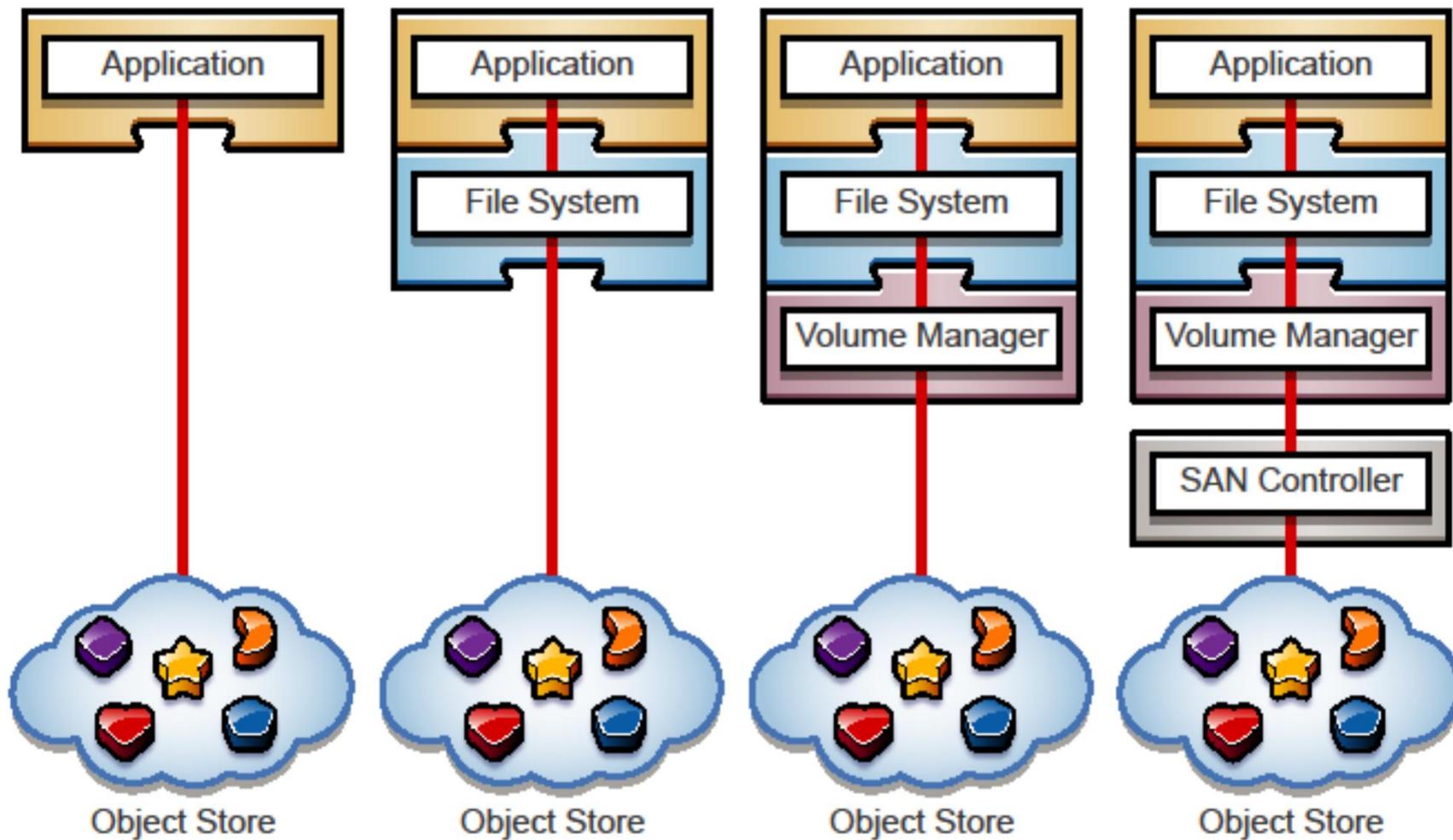
Object Storage: An Alternative Way to Address Stored Data



- An object is a chunk of storage with a specific set of properties that give it definition.
 - One of the key properties is an address scheme that abstracts the physical location of the object.
 - Traditional storage is addressed either as a sequential block of raw storage or a UNC path that points to a file
 - Often object addresses are self-describing, meaning that the address of the object could be the result of a hash
- Objects can interact at all levels of the storage I/O path
 - Applications can interact directly with objects
 - Individual files can be represented as objects
 - Files can be made up of a bunch of fine-grained objects
 - Disk volumes can be made up of a bunch of fine-grained objects



Collapsing the "Stack" with Object Stores



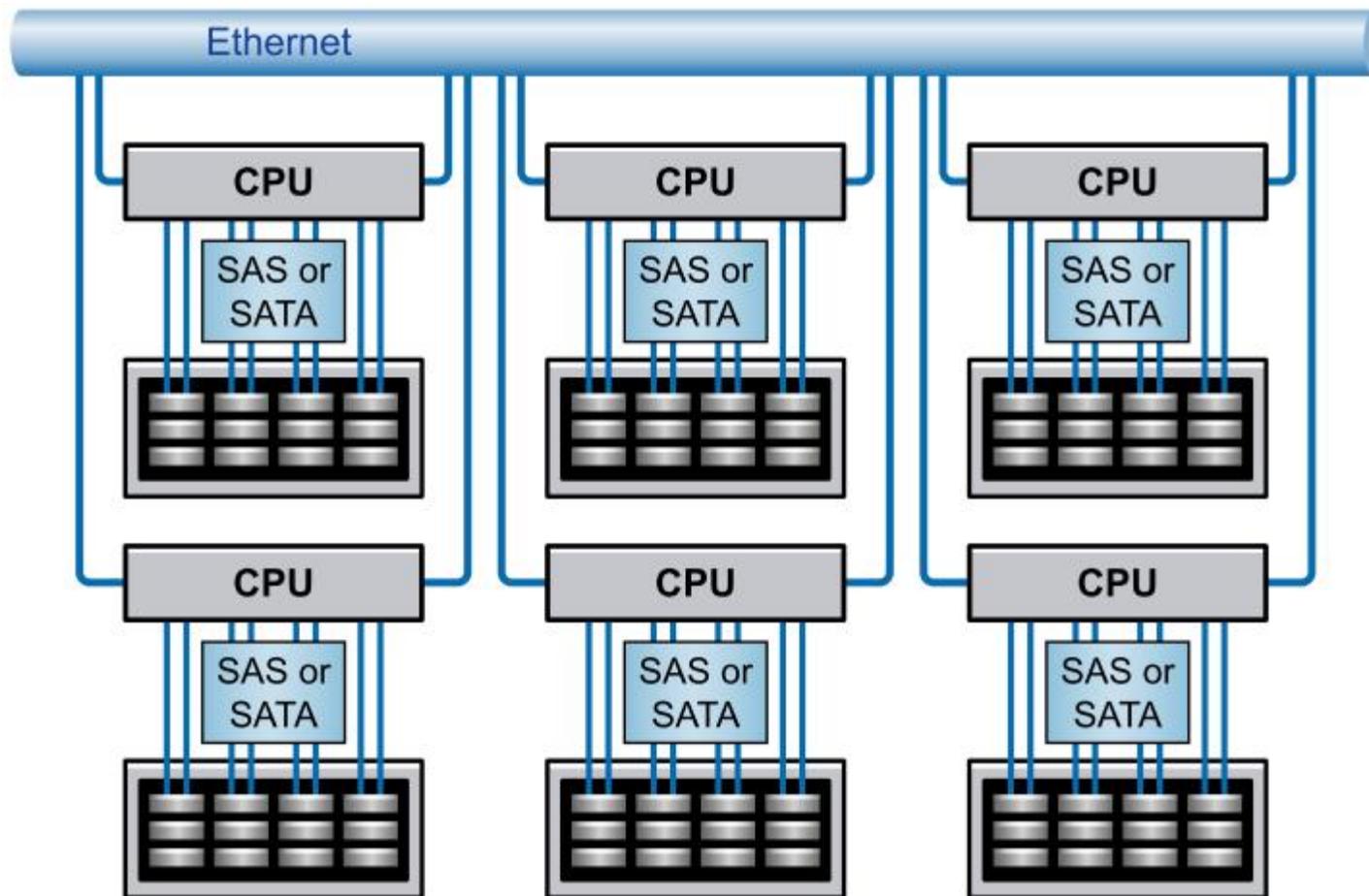
Seagate Kinetic – Object Store on a Hard Drive



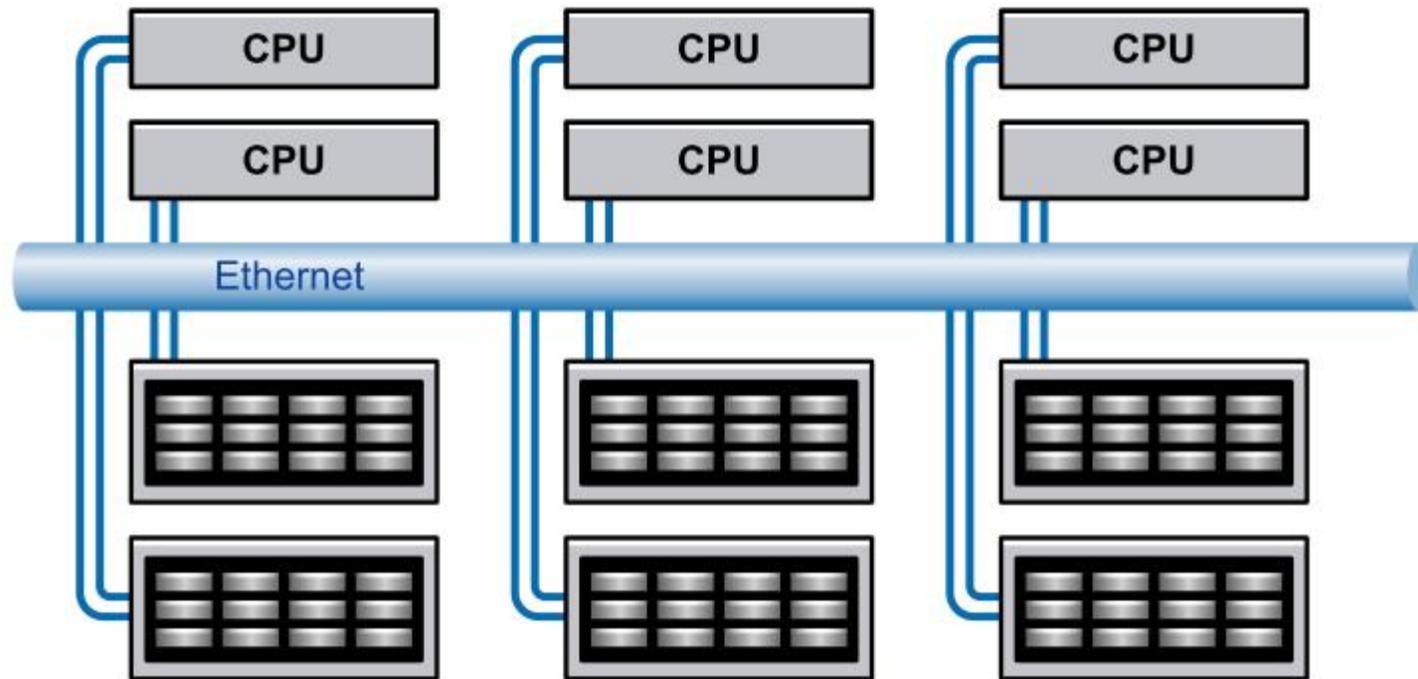
- Kinetic is a new product from Seagate that will begin shipping in the fall of 2015
- Each Kinetic hard drive is a self-contained object store
 - Instead of SATA or SAS connection, each drive has dual 1Gb Ethernet ports
 - Instead of SCSI or ATAP block protocol, the drives have a RESTful API to an internal key-value store
 - Drives can copy objects between one another
- Kinetic simplifies the storage stack, especially for SMR drives.



Before: Traditional Object Servers



After: Seagate Kinetic





CAMBRIDGE
Computer
ARTISTS IN DATA STORAGE

Object Storage for HPC

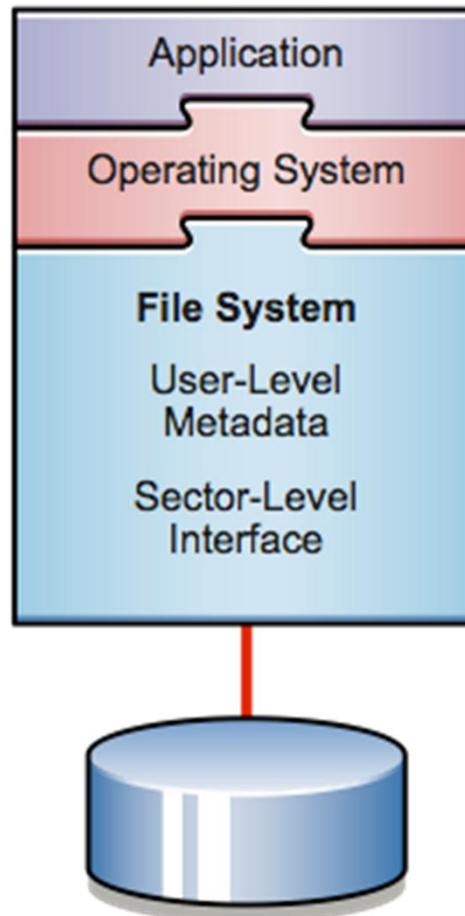
“Object File Systems” and OSD T-10



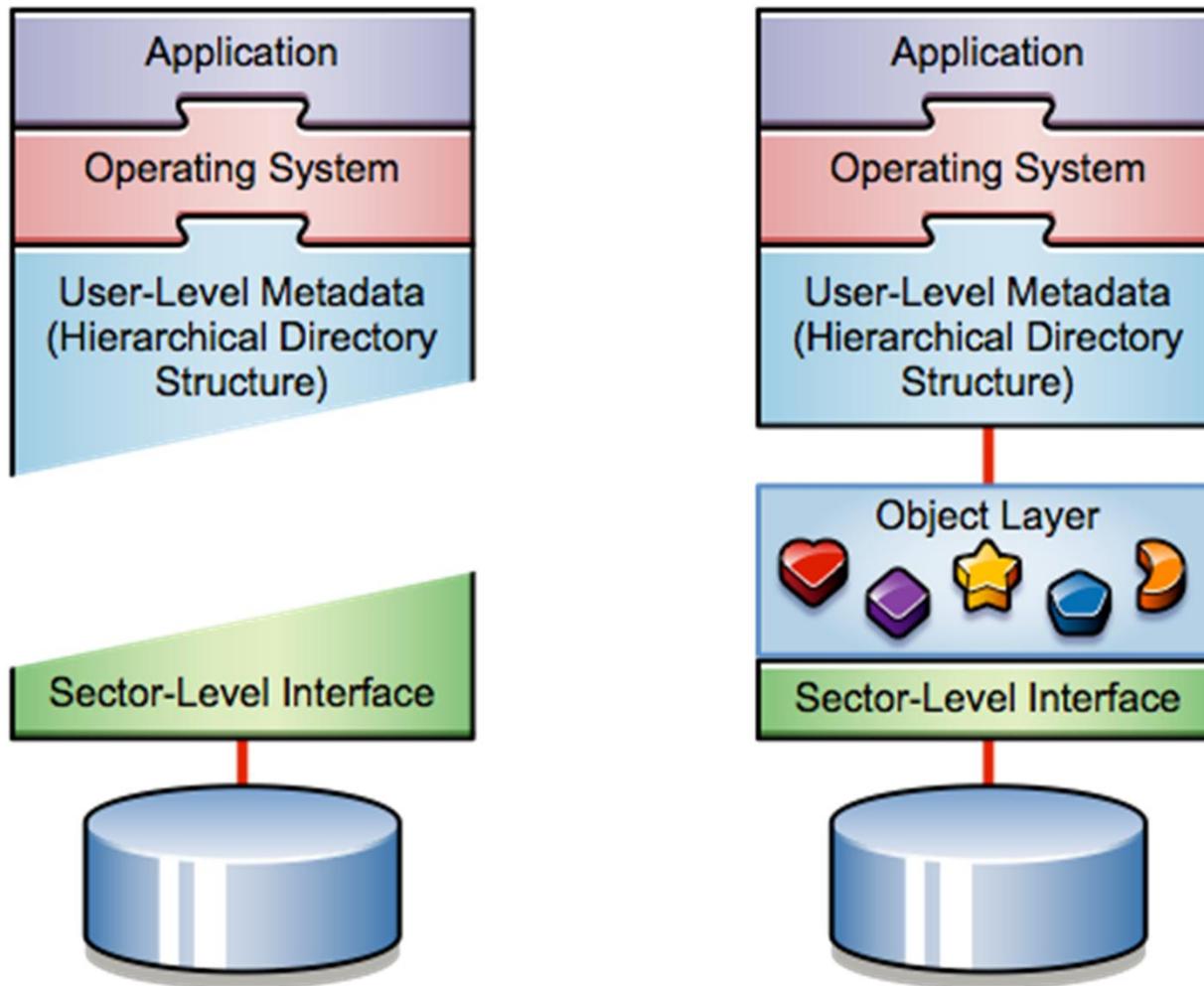
- OSD T-10 is an effort to replace blocks with an object model.
 - Allows block storage devices to be smarter
 - Enables security for raw storage
 - Enables file systems to span across storage arrays without traditional fears of losing device control
- SCSI specification amended to accommodate object command set.
- NOTE: Very few systems use OSD T-10, but the overwhelming number of papers describe “object storage” in this way.



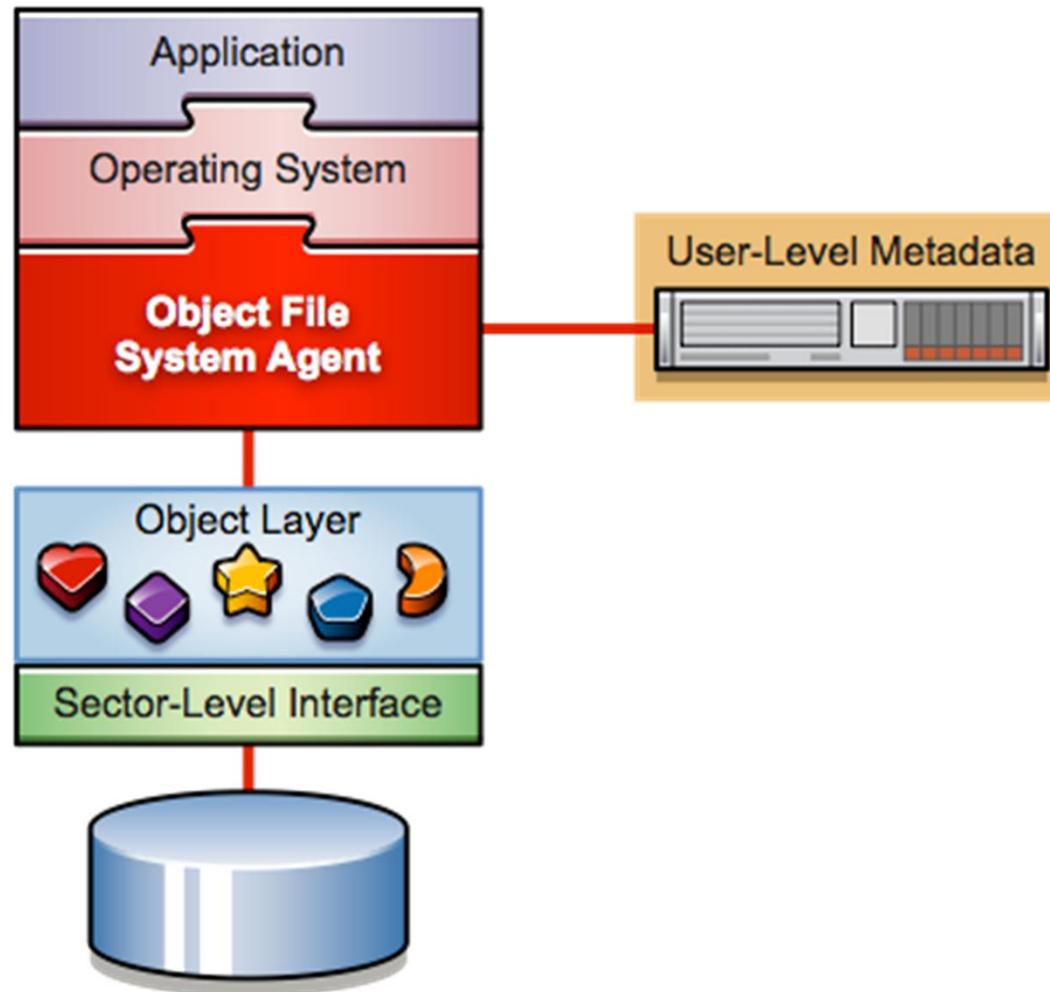
A Traditional Disk File System Has Two Very Distinct Roles



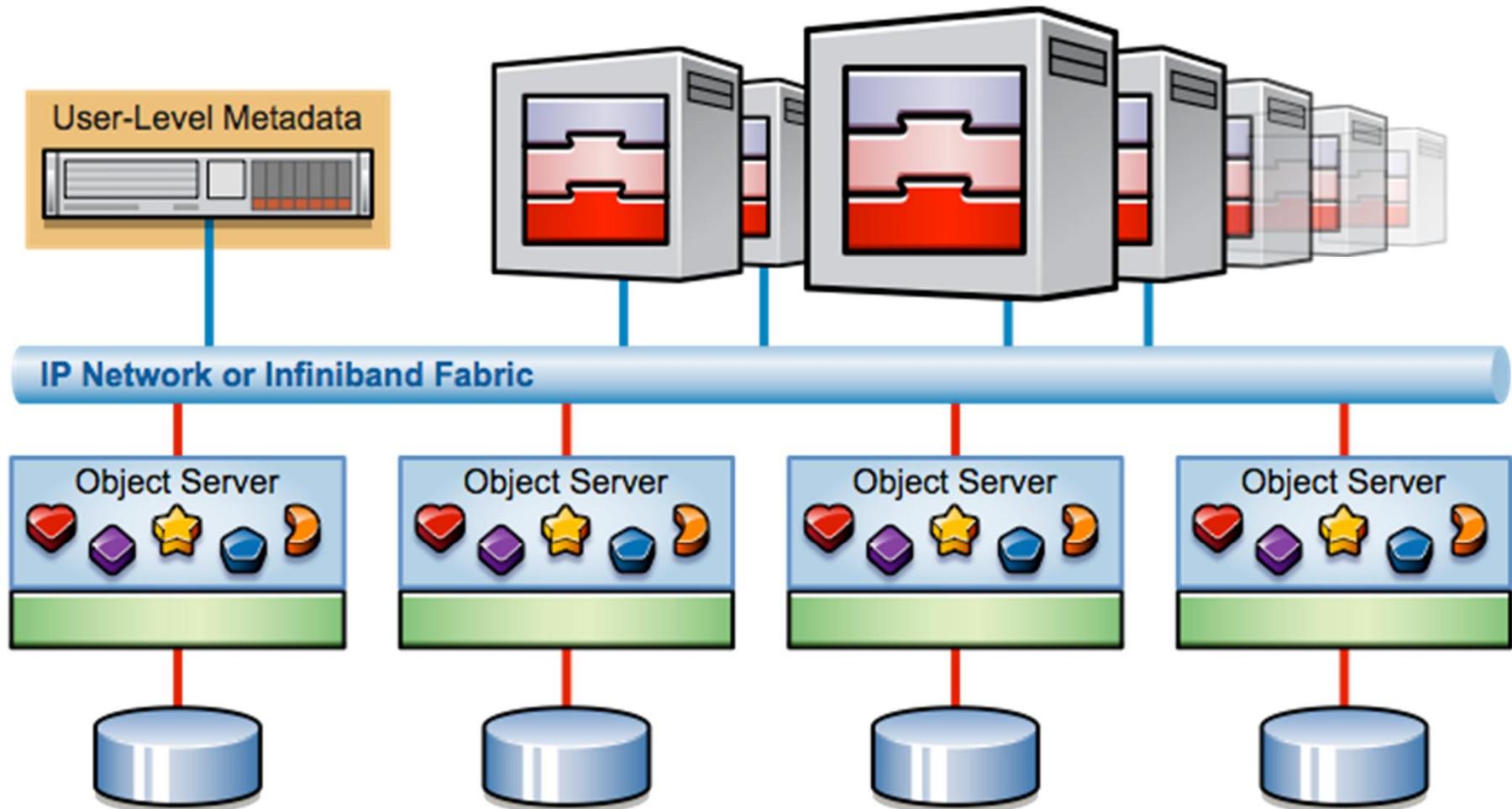
Separate the "What" from the "Where"



Out-of-Band Metadata for Locking, Access Controls, Directory Structure



Scale Out Capacity and Performance With Object File System



What Problems Were We Trying to Solve



- Build big with small parts based on commodity storage
 - Avoid heavily-engineered sub-systems
 - This requires a storage network that is fast, reliable, and inexpensive
- Build a storage network that is purpose-built for HPC and does not require storage channel protocols between clients and object stores.
 - Achieve lower latency
 - Achieve lower costs
 - Ability to address lots of devices
- Did it / does it work as advertised?





CAMBRIDGE
Computer
ARTISTS IN DATA STORAGE

Object Stores for High Capacity and Resiliency

Petabyte Scale Growing Pains

Challenges as You Grow Big



- Scalability of the individual file system
 - How much capacity can you have in a single file system?
 - How many files?
- Backup and Restore
 - How do you back up a Petabyte?
 - How long is going to take to restore?
- Fault tolerance
 - How do you make it fault tolerant?
- Disaster recovery
 - How do you get the data off site? How do you get it back?
- Hardware refresh
 - How do you replace old hardware with new?
 - Especially hard if you have replicated a large file system
- Namespace issues
 - Search, indexing, workflow become highly desirable



Moving a PB is Heavy Lifting



Data Rate	Example	Total Time (Approximate)
140MB/Sec	LTO-5 tape drive at full tilt without factoring in compression	82.5 days
1GB/Sec	A beefy Virtual Tape Library A dedicated 10Gb Ethernet	11 days
1.5mb/Sec	A dedicated T-1	176 years
156mb/Sec	An OC3	640 days
2488mb/Sec	An OC48	40 Days



Bigger Hard Drives: Friend or Foe?



- The Good News: As drives grow bigger we can achieve more capacity with fewer devices
 - Fewer devices = higher density, lower power consumption, fewer device failures
- The Bad News
 - MTBF not growing as fast
 - Bandwidth into device not growing as fast
 - Consequences
 - Unreliability (per bit) growing
 - Accessibility of data (per bit) shrinking
 - Drive rebuild times are longer, which increases overall risk of data loss
 - Rebuilding failed drives has a heavier impact on performance



RAID Rebuilds Take Too Long



- RAID 5 rebuilds take too long
 - On the order of 36 hours per TB
 - 4TB drive could take a week to rebuild
- RAID 6 (double parity offers some protection)
 - But what happens when we have 8TB drives?
- The more stuff you have the higher the chance of failures.
 - If you have 1PB or more, something will always be broken



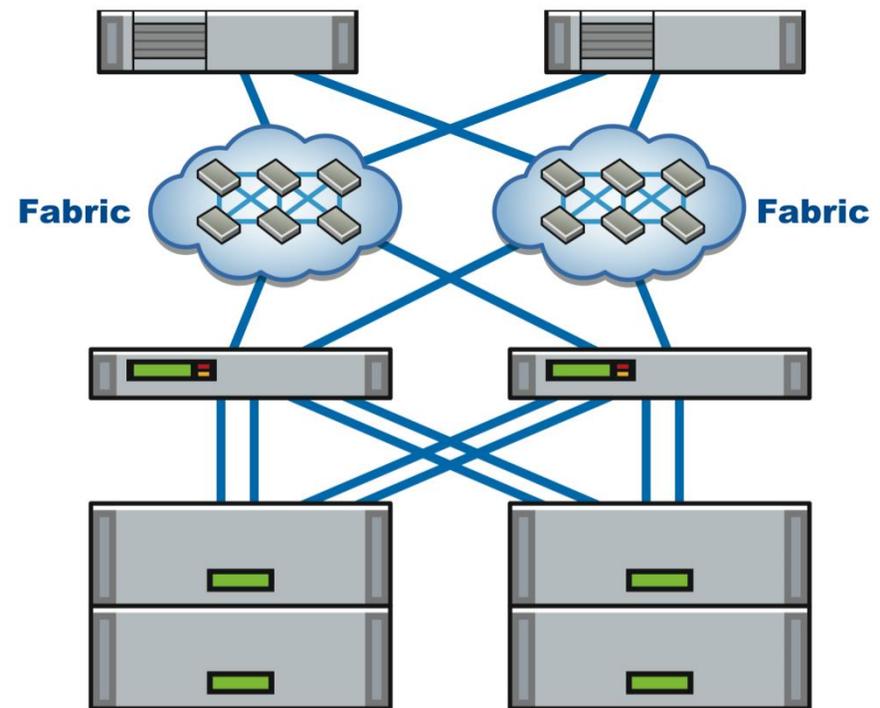
Modern Petabyte-Scale RAID Array

- Drives are striped in groups of 10 using RAID 6.
- Drives can be striped vertically across cabinets.
- An entire cabinet can fail without losing data.



Redundancy Between Cabinets: Can You Have Too Much Redundancy?

- Is this really a good idea?
- How long will it take to re-mirror a 14TB RAID 6 stripe?
- Is there a better way to protect against a device failure?
 - Replication?
 - Backup?
 - Mirroring at a different level of abstraction?



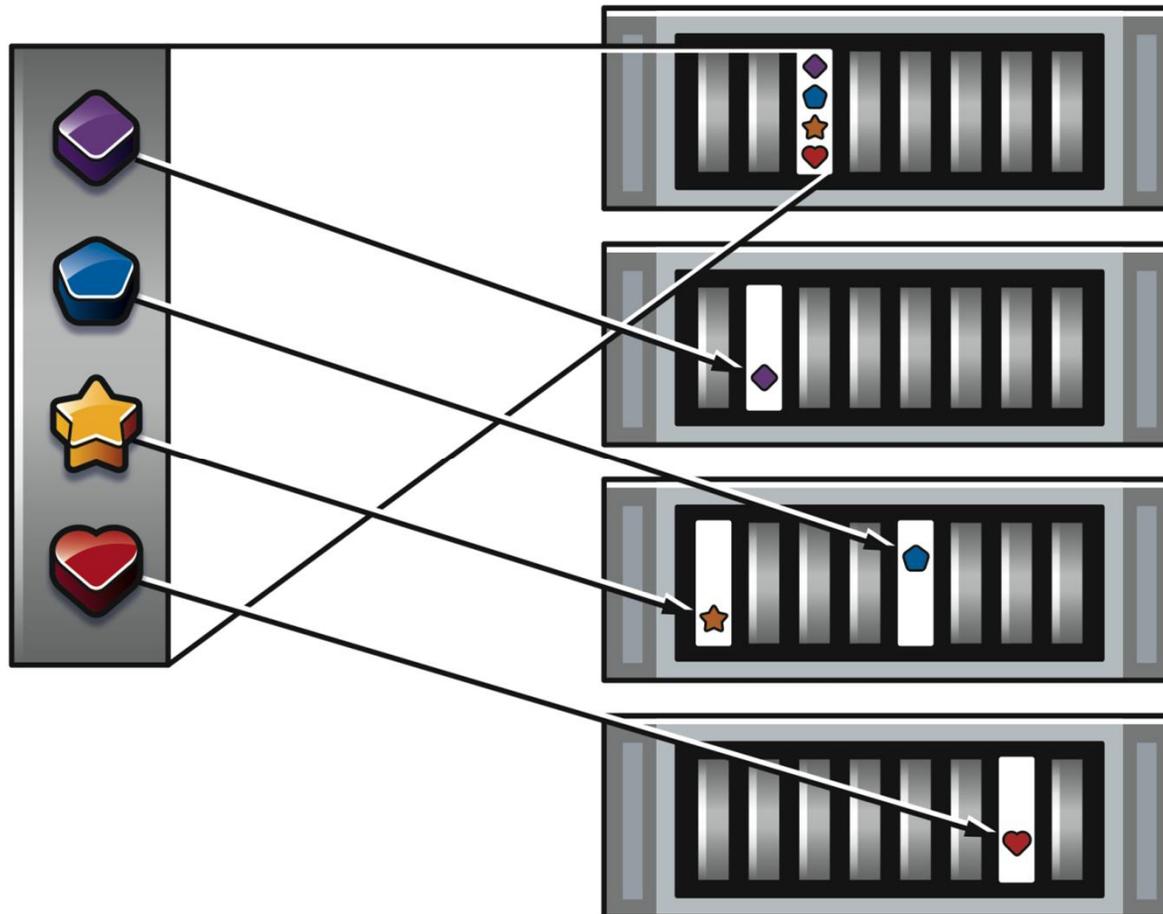
How Big is the Building Block?



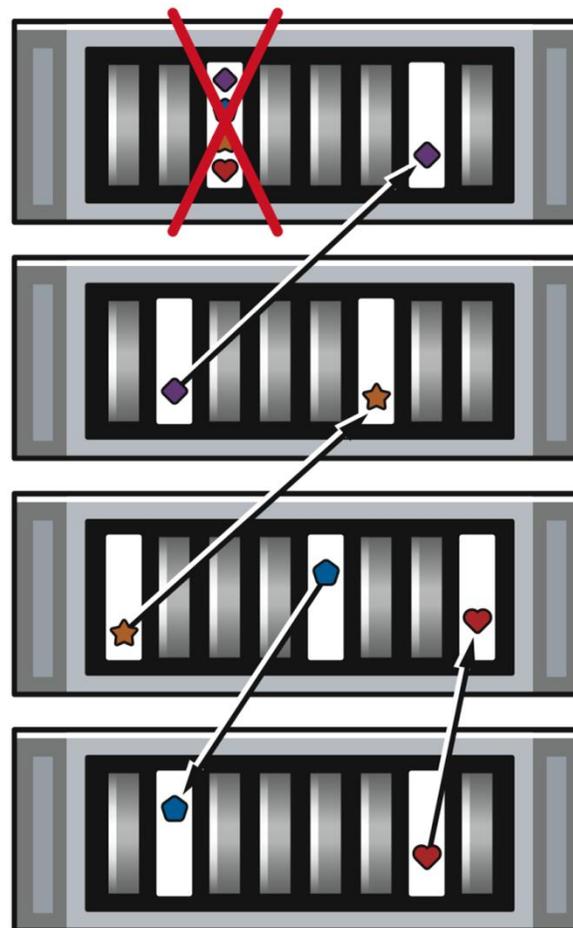
What Are You Building?	What Size Building Block?
An outhouse?	Brick
The foundation for a new house?	Cinder Block
A pyramid?	Boulder
A parking garage?	Grains of Sand (Concrete)



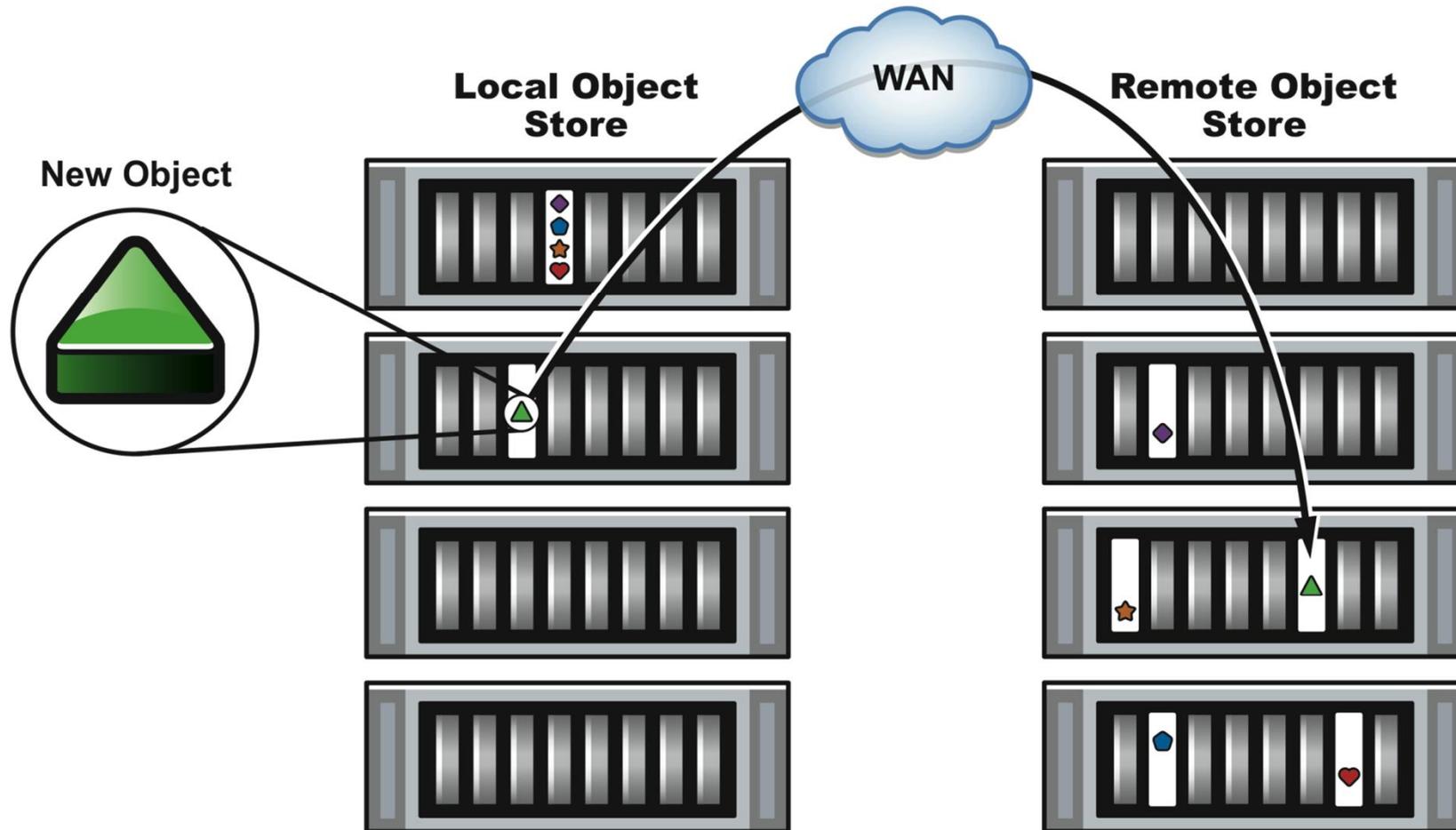
Basic Object-Level Redundancy: An Alternative to RAID and Mirroring



Redundant Objects Propagate on Device Failure



Object Mirroring Across a WAN



Erasure-Coded Data Protection: An Alternative to Parity-Based RAID



New object



Object is broken into k segments



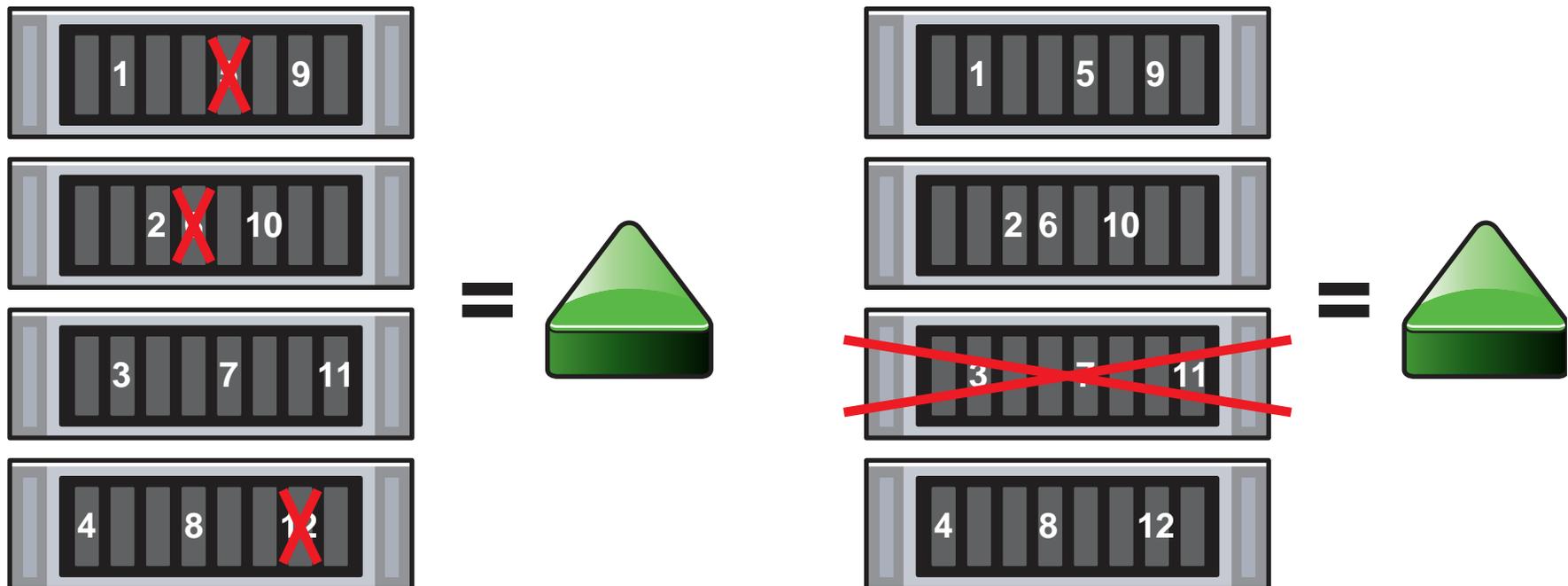
m parity segments are generated



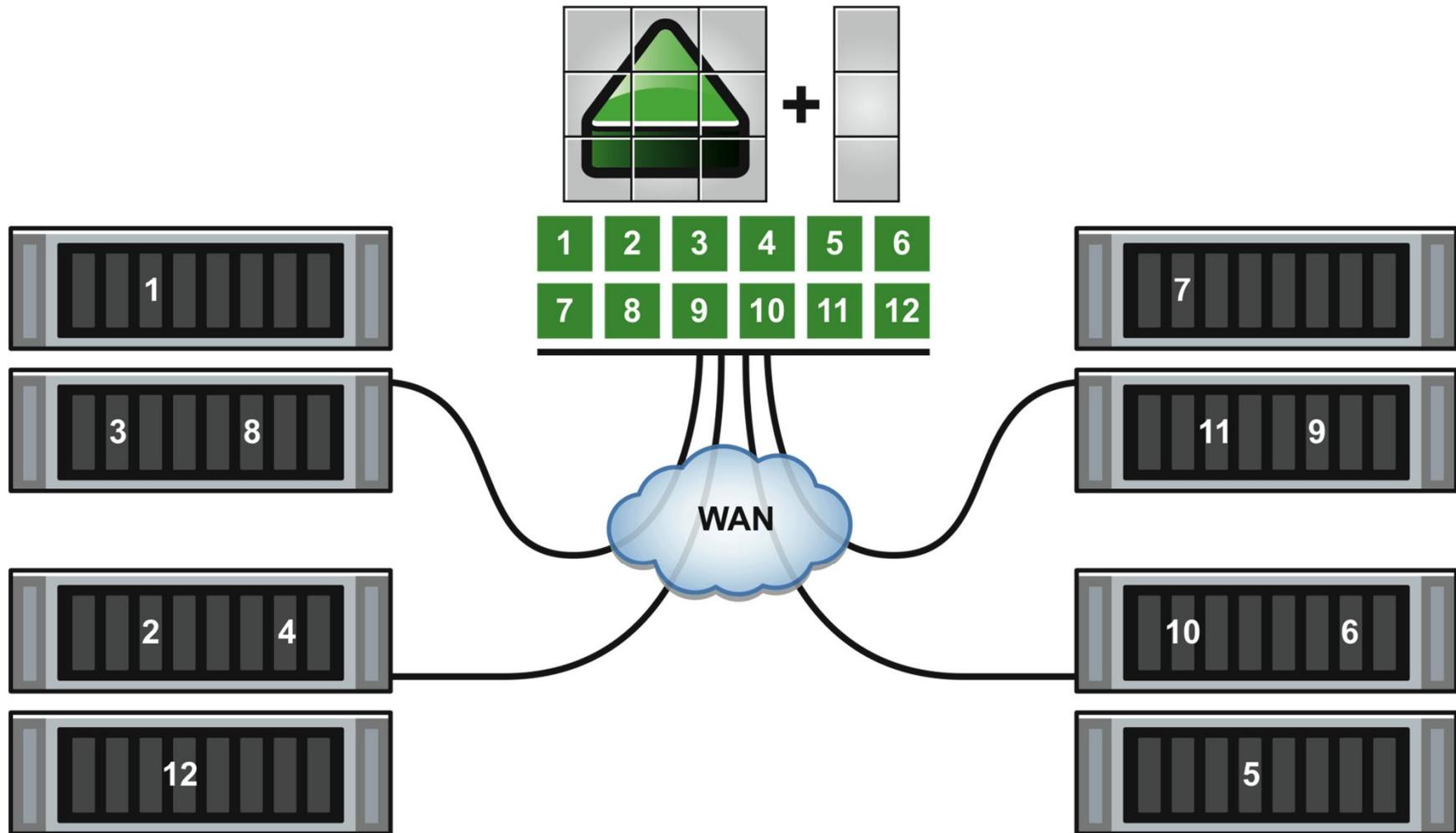
n elements are dispersed across devices



You Can Lose M/N of Your Storage Without Losing Data



"Dispersed" Storage: Erasure Coded Storage Across the WAN



Summary: What Problems Are We Trying to Solve?



- FAST – CHEAP – RELIABLE = The Elusive Trifecta
- Massive capacity
- Fault tolerance
 - We rebuilt drives super-fast
 - We can tolerate intermittent failures
 - We can abandon broken drives or deal with them later
- We can use lousy hardware
- Geographic dispersal
- Eliminates the need for POSIX interface
 - But that's not always a good thing





CAMBRIDGE
Computer
ARTISTS IN DATA STORAGE

Content Addressing and Deduplication

- Content addressing refers to the process of deriving the object's ID by hashing the object itself.
 - This is sometimes referred to as "self-describing"
 - There can never be a mismatch of the object address and the actual data stored within the object
 - With conventional file storage you can have two files with the same name and different contents
 - With conventional file storage you can have two files with identical contents and different names
- **Content Addressable Storage**
 - A term coined and popularized by EMC to describe the inner workings of their Centera storage device.
 - Many other systems work on this same principle.

Content Addressing and **Locality Independence**



- A single object can be physically stored in multiple locations without creating confusion
- Redundancy
 - Objects can be stored redundantly such that one object can stand in for the other in the event of a data loss or corruption
- Parallelism
 - A single object could be stored in multiple locations to allow for parallel processing of that object
- Local processing affinity
 - Objects can be permanently stored or cached in proximity to where they are being processed
- Cache coherency



Content Addressing and Data Integrity Assurance



- Data integrity is a big concern with large scale storage systems
 - The more storage you have, the higher the probability of data corruption
 - The larger your hard drives, the higher the probability that you will encounter unrecoverable errors
- Data integrity is a key concern for digital preservation and for regulatory compliance.
- Content addressing is an easy way to ensure data integrity:
 - Step 1: Retrieve an object by its content address
 - Step 2: Open the object and calculate a hash against its content
 - Step 3: Compare the hash to the content address. If they are not the same, then your object has been tampered with or corrupted.



Content Addressing and Deduplication



- Content addressing is fundamental to deduplication
- Any two objects with the same address are the same object, even if they have different names
 - Step 1: Encounter a new object
 - Step 2: Calculate a hash of the object
 - Step 3: Look up the hash in a table or index of existing objects
 - Step 4: If the object already exists, store a reference. If the object does not exist, store it and make an entry in the index.

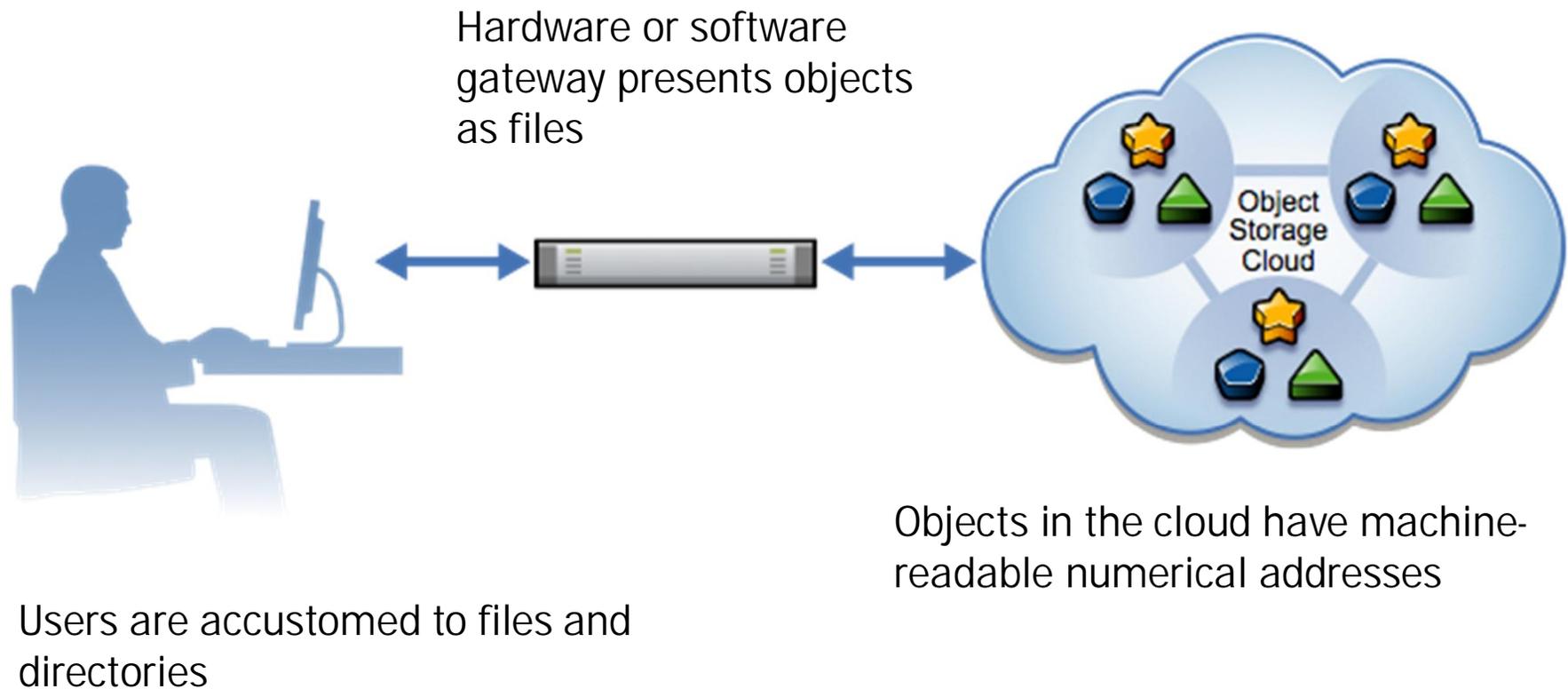




CAMBRIDGE
Computer
ARTISTS IN DATA STORAGE

Layering Object Models on Top of One Another

The Cloud is Accessed Through SOAP/REST Interface



Object Granularity: Fine-Grained v. Coarse-Grained Objects



Fine-Grained

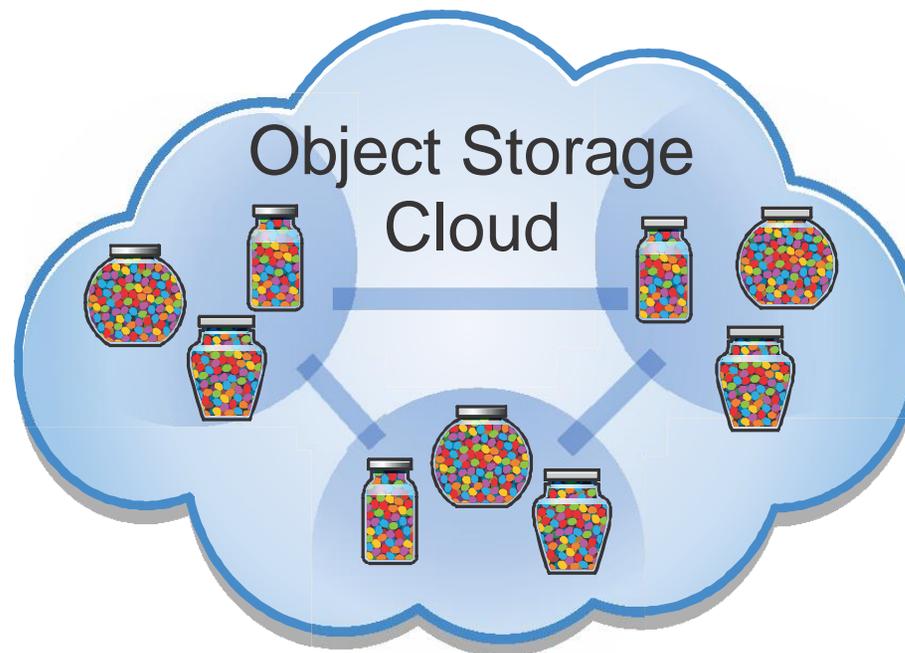
- Object is a portion of a file, akin to a block
 - But might be variable in size
- Objects are opaque – individually they are just blobs of data
- Very friendly to caching and distribution over a WAN
- Might be friendly to sub-file-level deduplication

Coarse-Grained

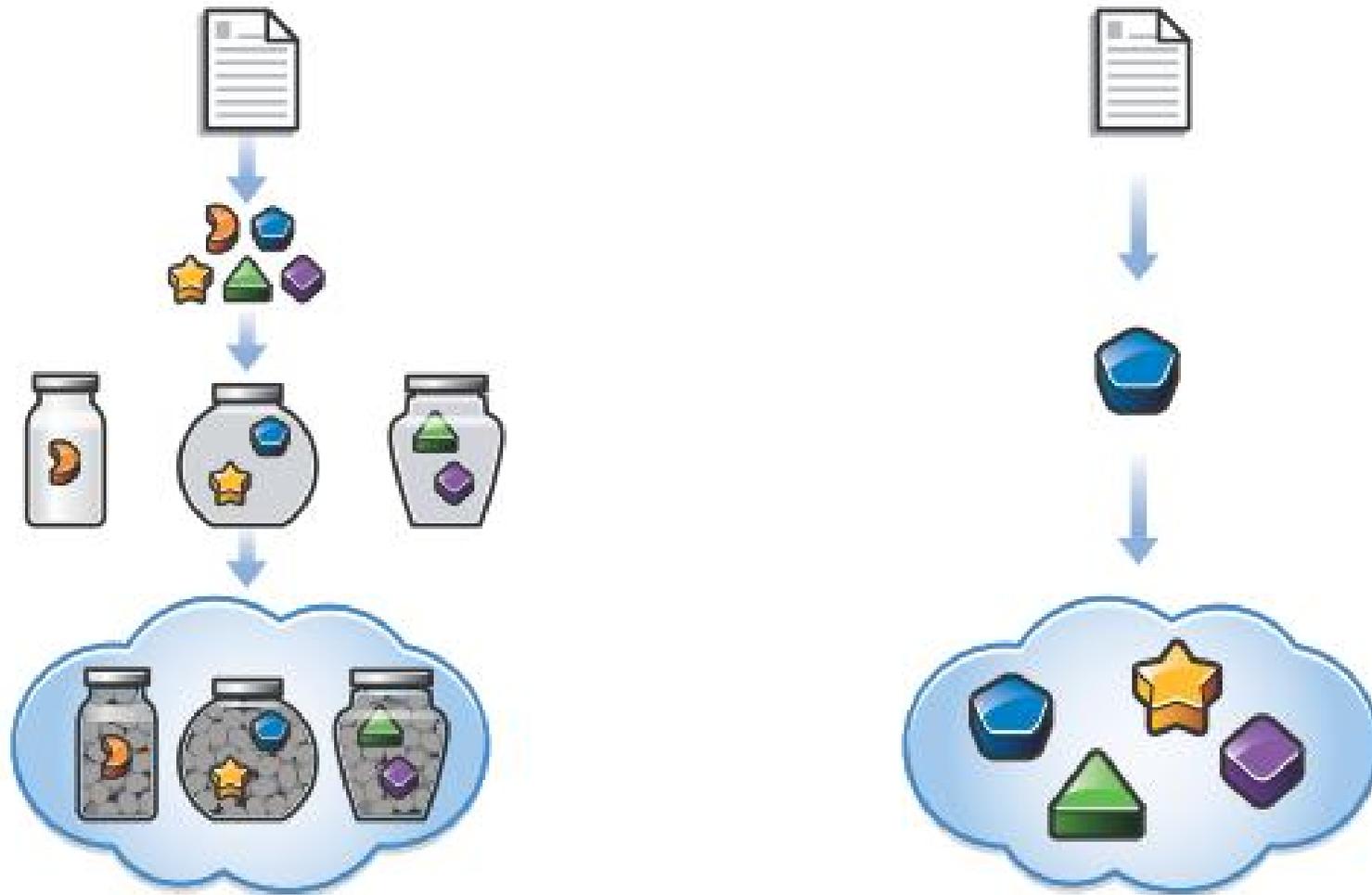
- Object is a whole file or some kind of “container”
- Changes made to the file might generate a whole new object
 - Deltas between versions can be stored as objects that reference a parent object
- Often have additional properties (metadata) associated with them



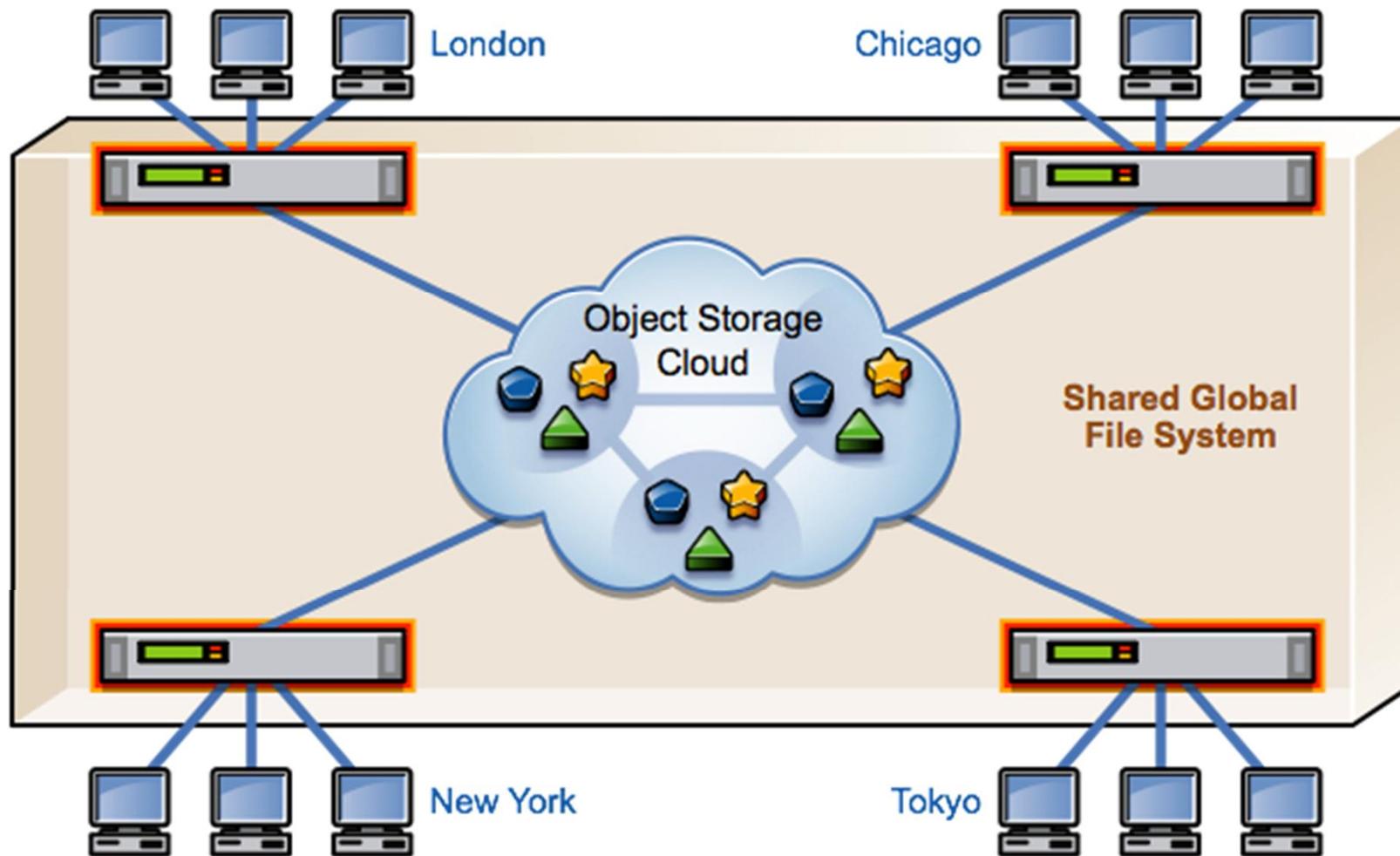
Coarse-Grained Objects Can Contain Fine-Grained Objects



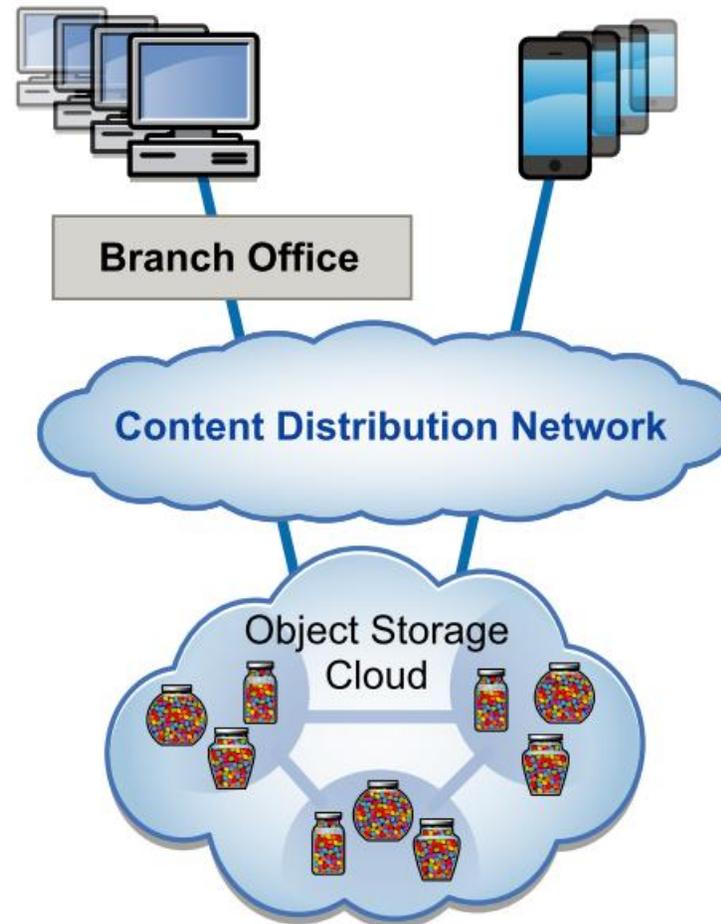
Mapping Files to Objects In a Cloud-like Object Store



Shared File System Leveraging a Cloud-based Object Store



Multi-level Caching of Object-based File System

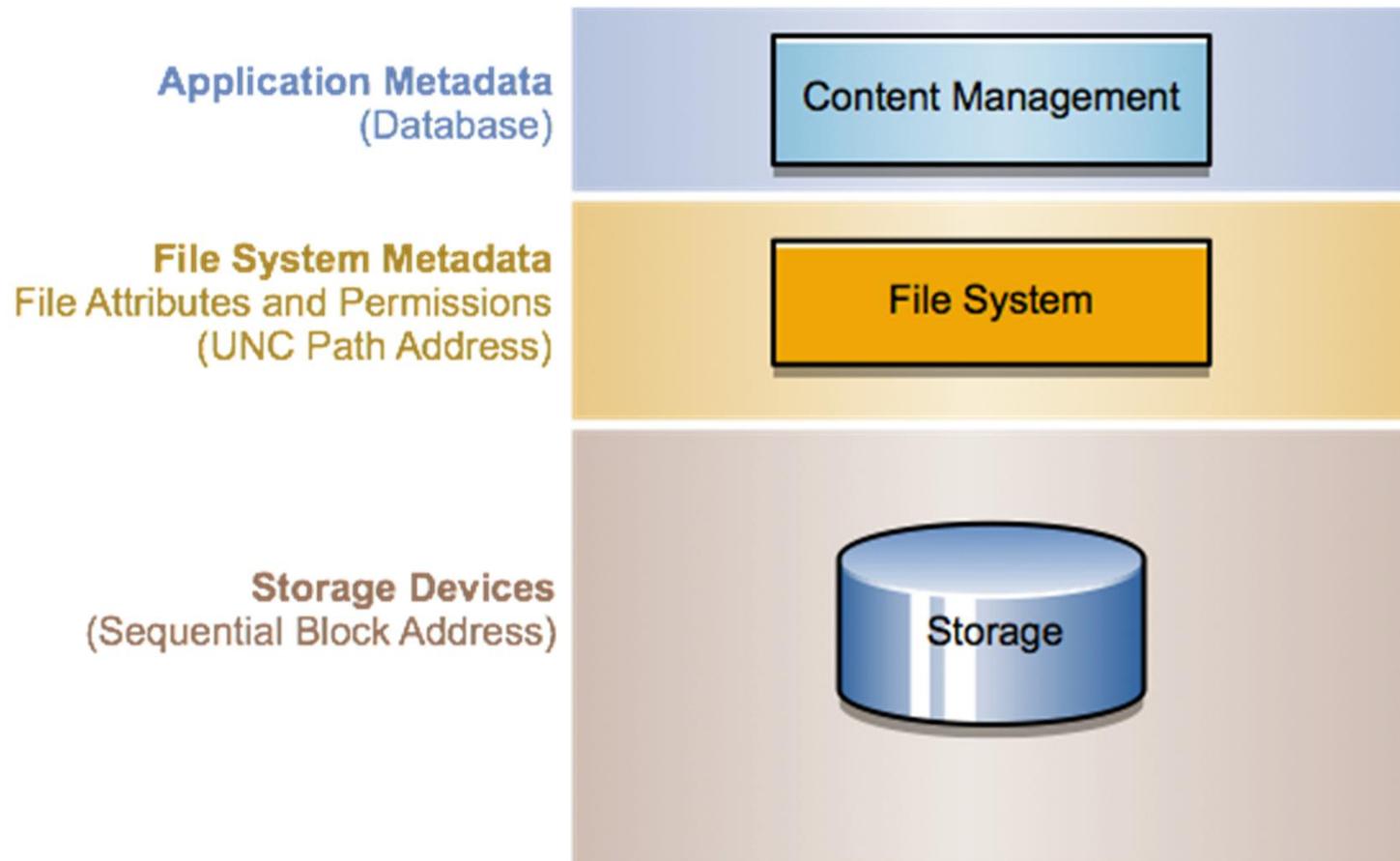




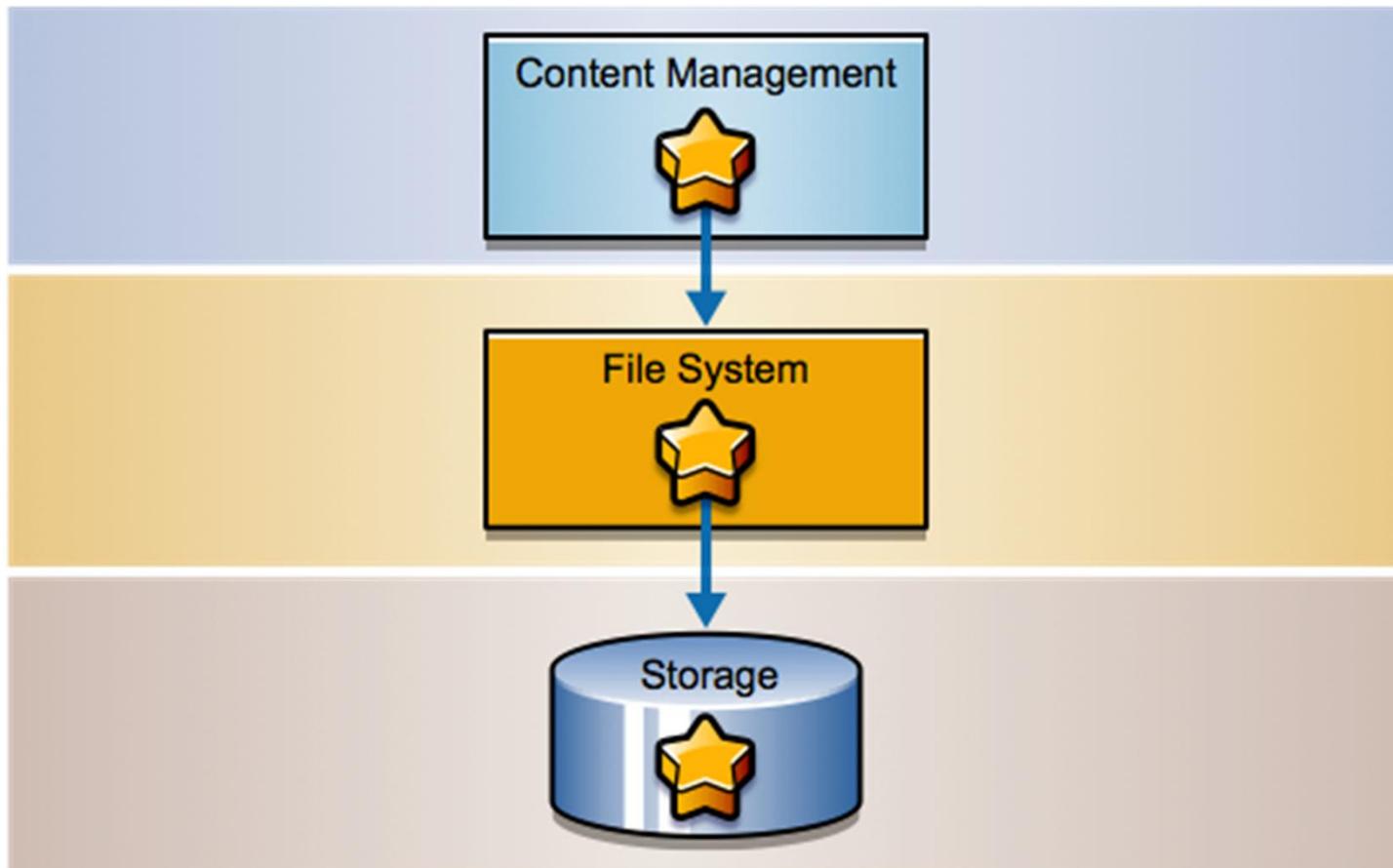
CAMBRIDGE
Computer
ARTISTS IN DATA STORAGE

File System Middleware for Digital Object Management

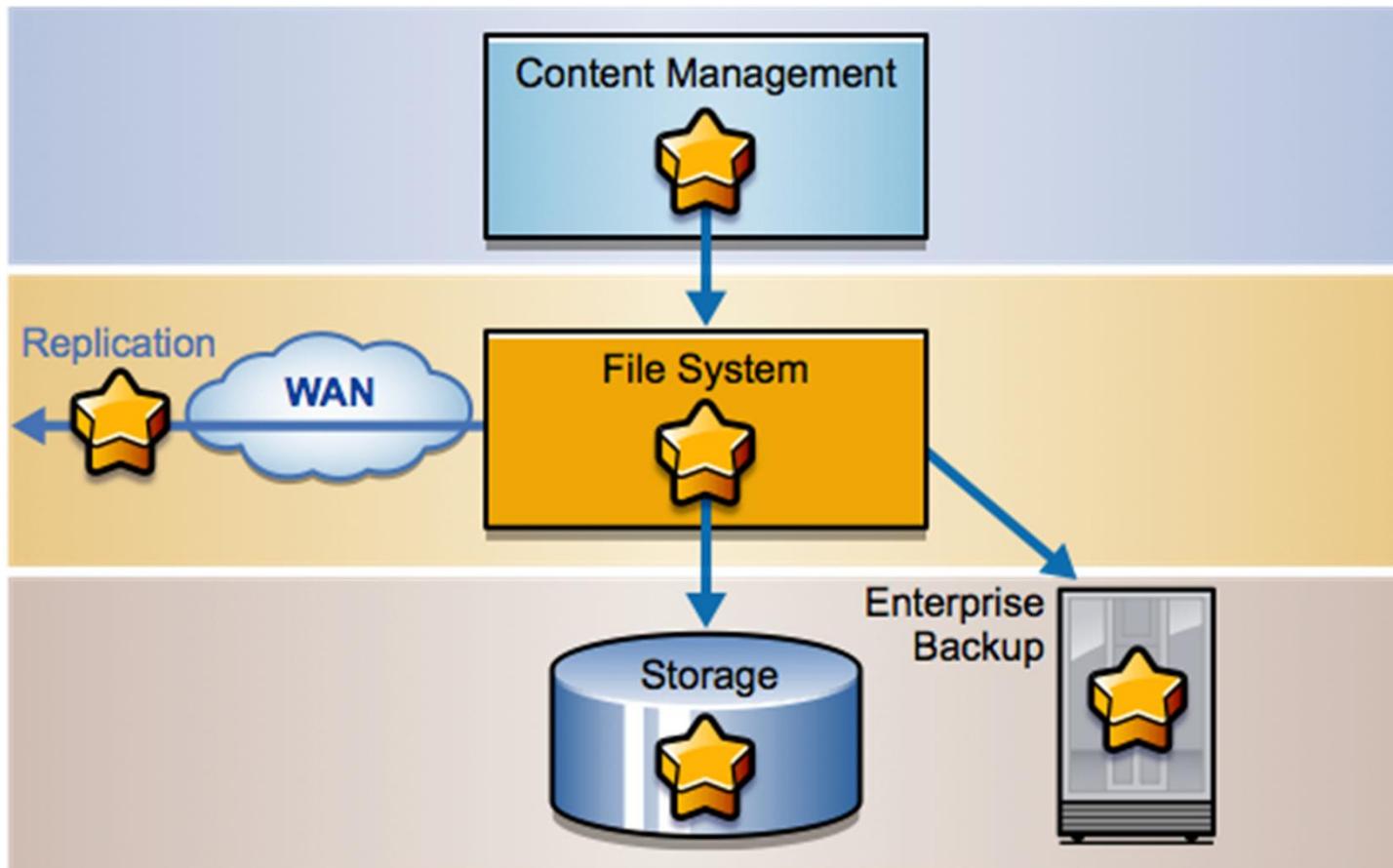
Typical Content Management "Stack"



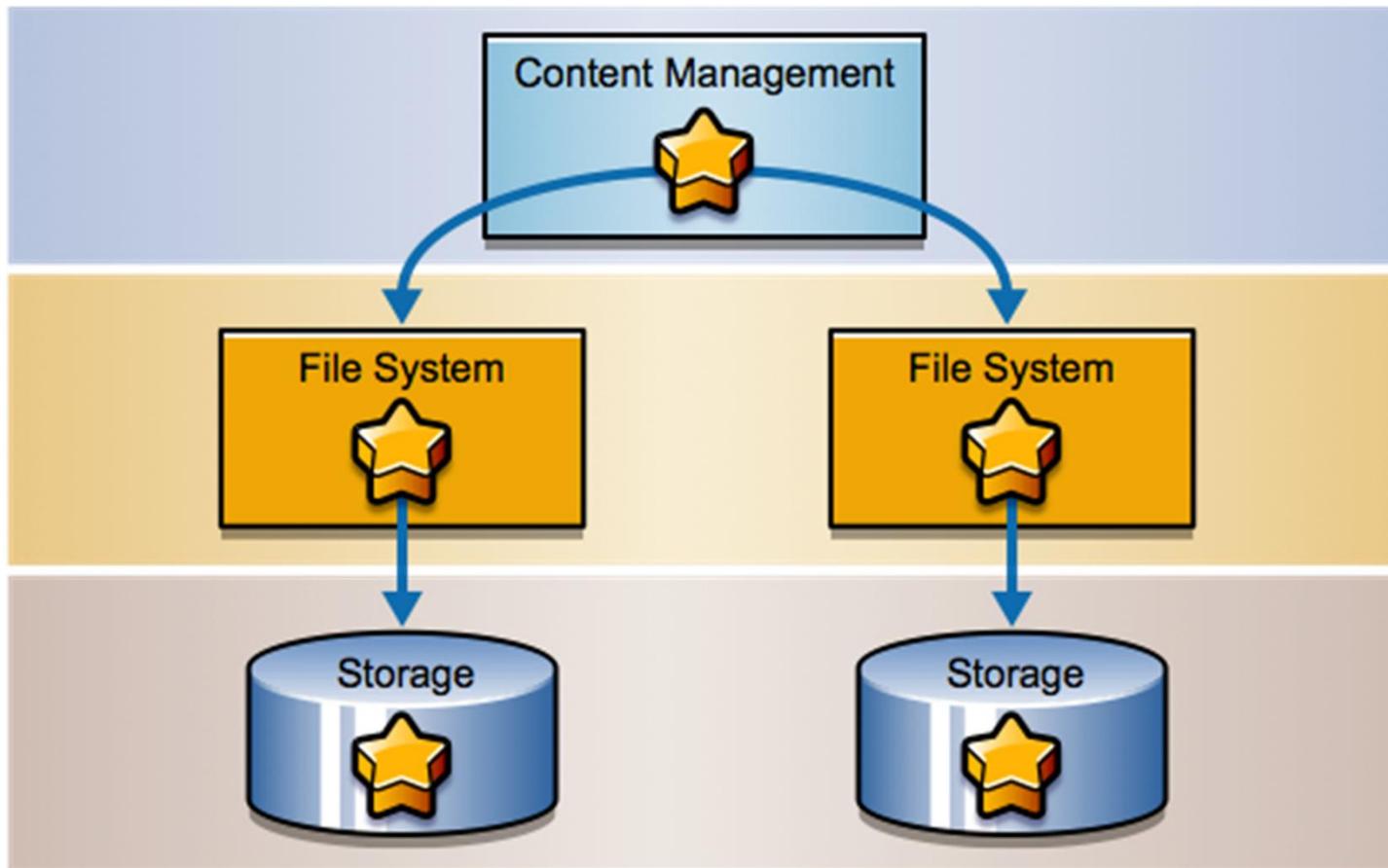
An Object Flows from Application to Disk



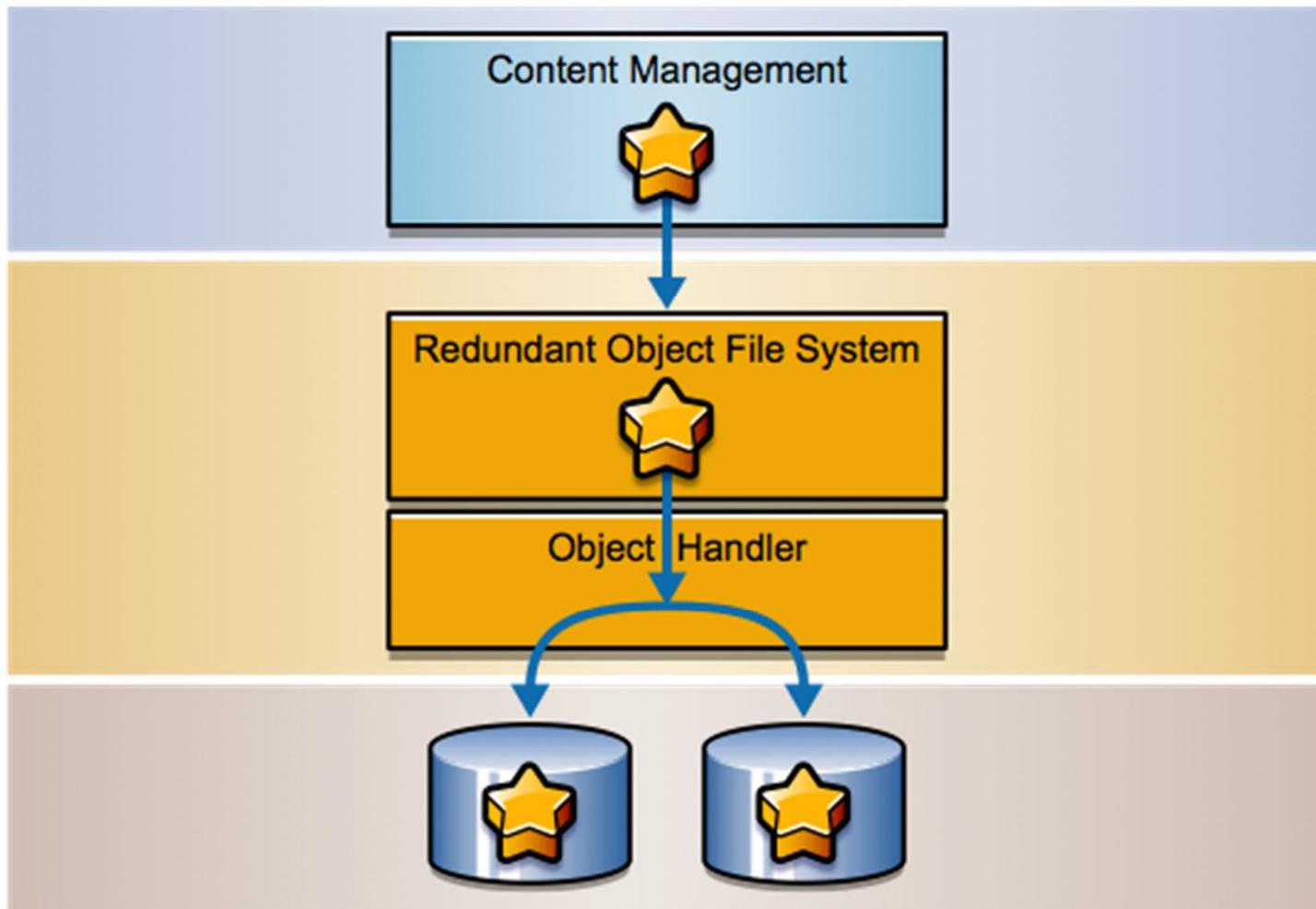
Conventional Enterprise Data Protection



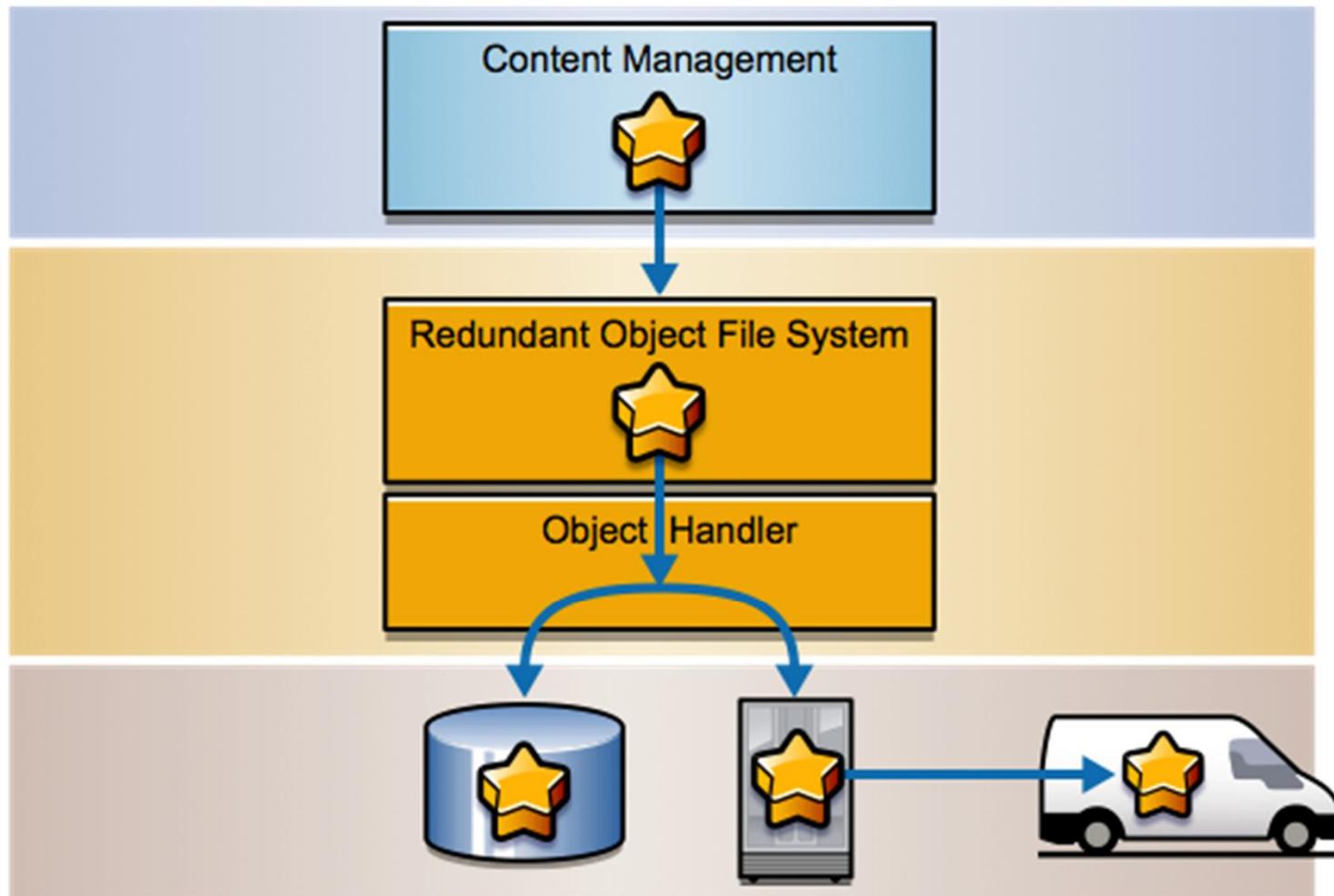
Application Manages Redundant Copies of Object



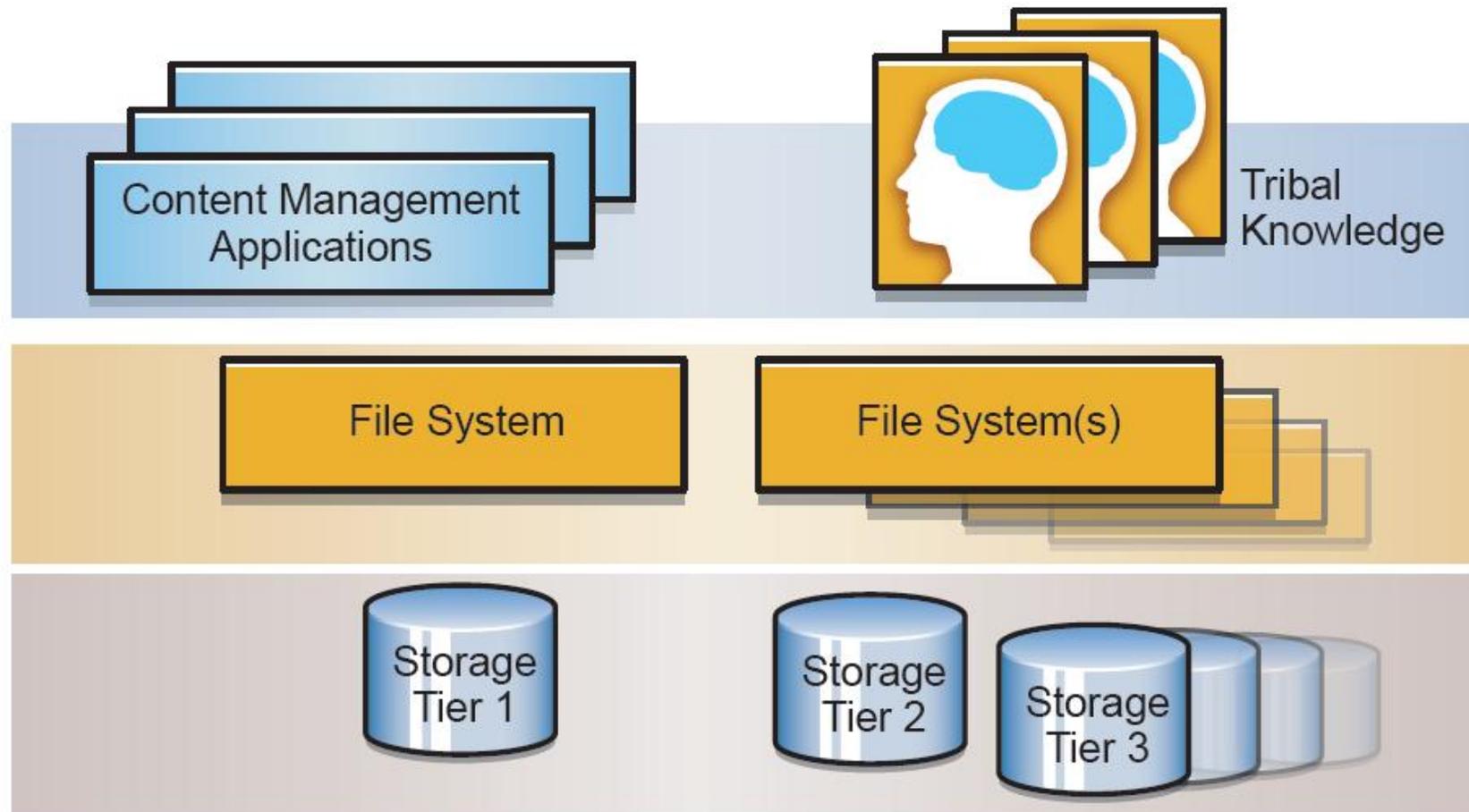
Disk-Based Object Store Enabling Redundancy



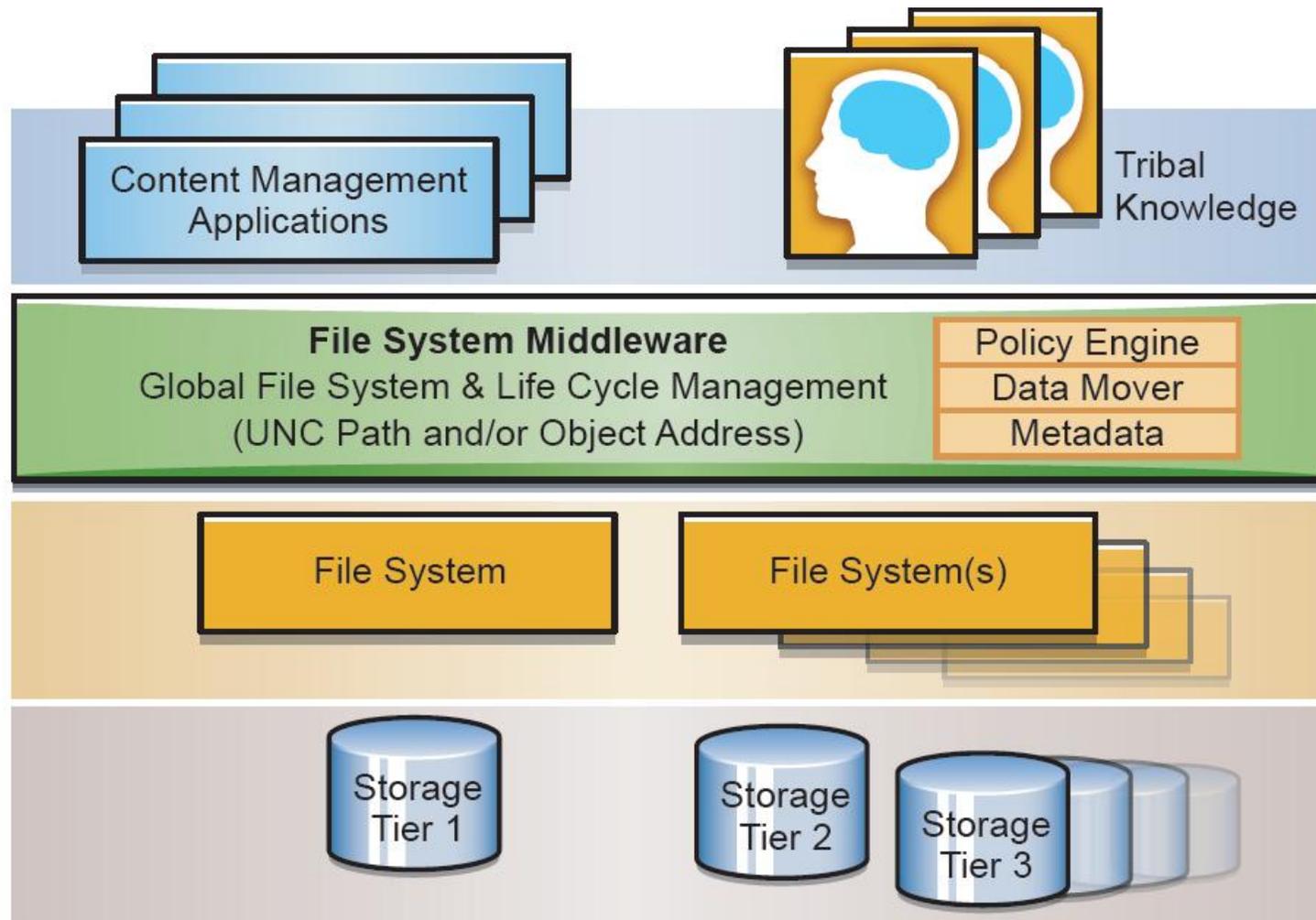
Automated Backups With Object-Based Archival Storage System



Data Management "Stack" In a Diverse Institution



Adding Middleware to the Data Management Stack



- SRB – Storage Resource Broker
- IRODS – Integrated Rules Oriented Data Service
- Applications that play the role of a file system while treating each file as a digital object
 - Extensible Metadata is associated with files to give them more meaning
 - Files are organized into virtual containers, breaking free of traditional file system limitations
 - Redundant copies of files can be served as a single logical files
 - Rules engine takes action based on file system metadata and extensible metadata
 - Files are served to other applications via API or file system gateway



CAMBRIDGE
Computer
ARTISTS IN DATA STORAGE

Starfish – A Cambridge Computer Software Initiative

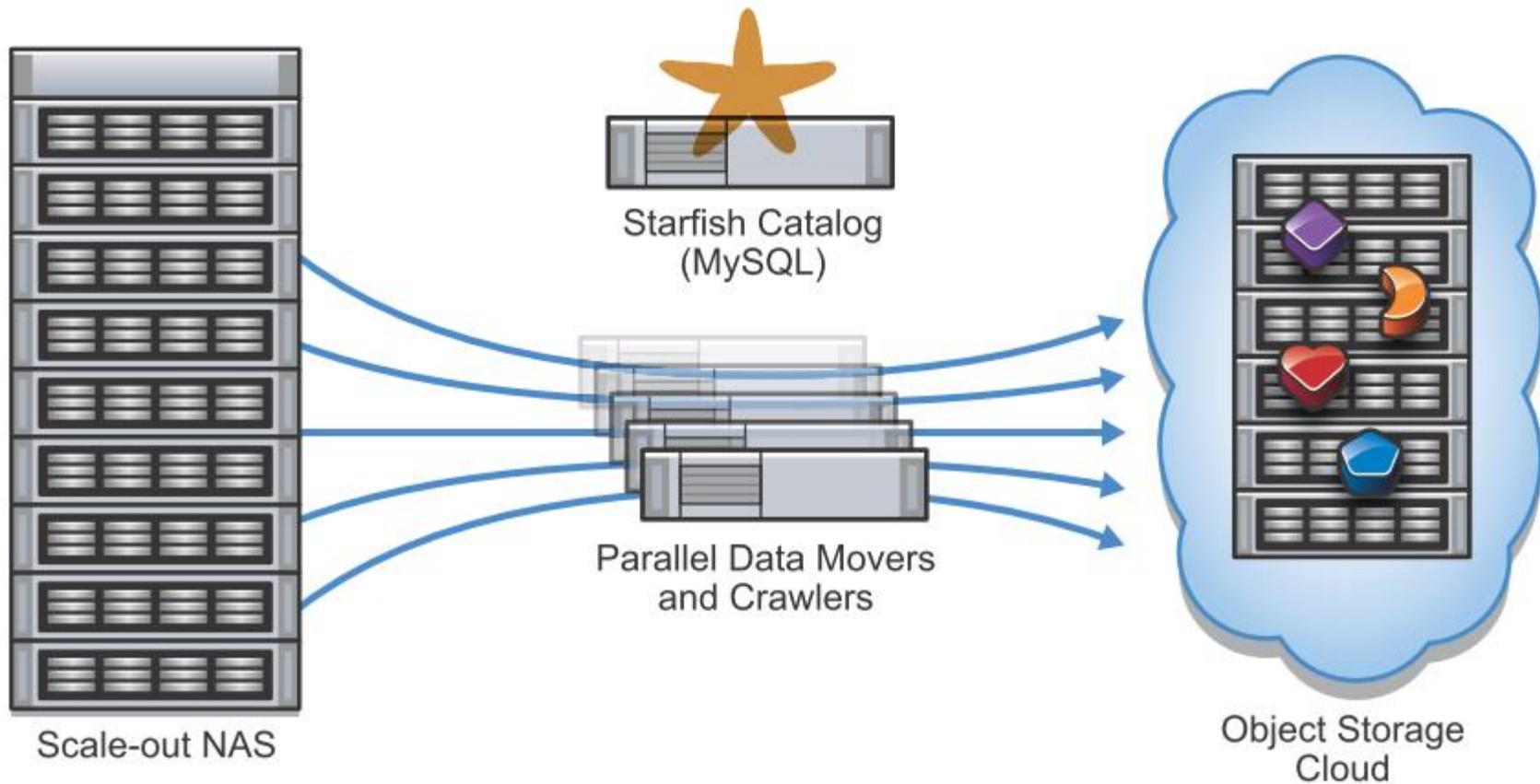
Starfish Overview - Treat Files Like Objects



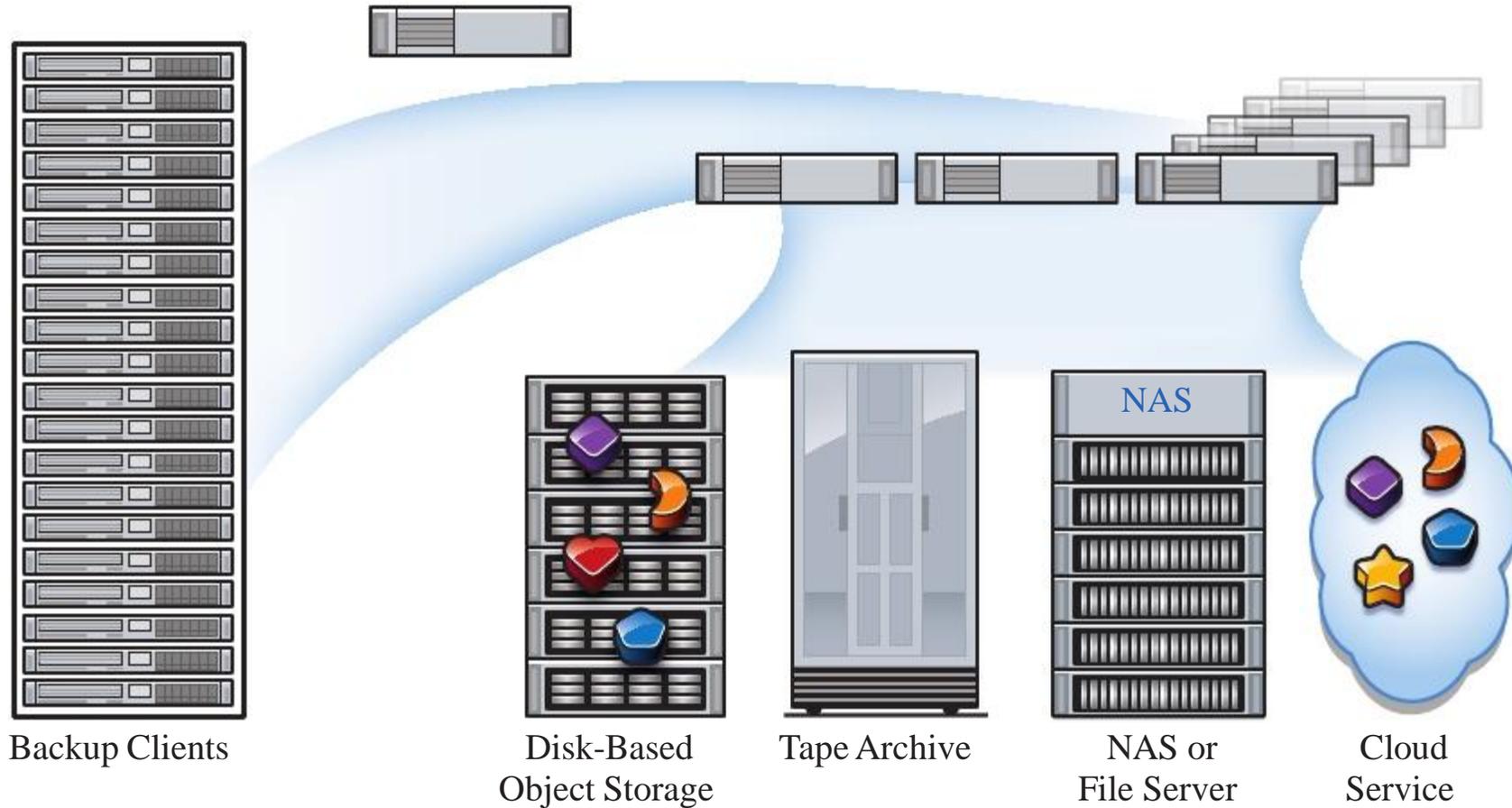
- Virtual Global File System
 - *FS = StarFS = Starfish
- We make (and maintain) a big database that stores a record for each file
 - We capture permissions, and attributes
 - Users (via GUI) and applications (via API) can associate metadata with applications and directories
 - Reporting framework (canned reports, custom reports, SQL query)
 - Rules based management – copy, move, delete, other operations
- Built for scale – Petabytes and billions of files
- Designed to operate out-of-band
- Initial target markets: Scientific Data Management and Digital Preservation



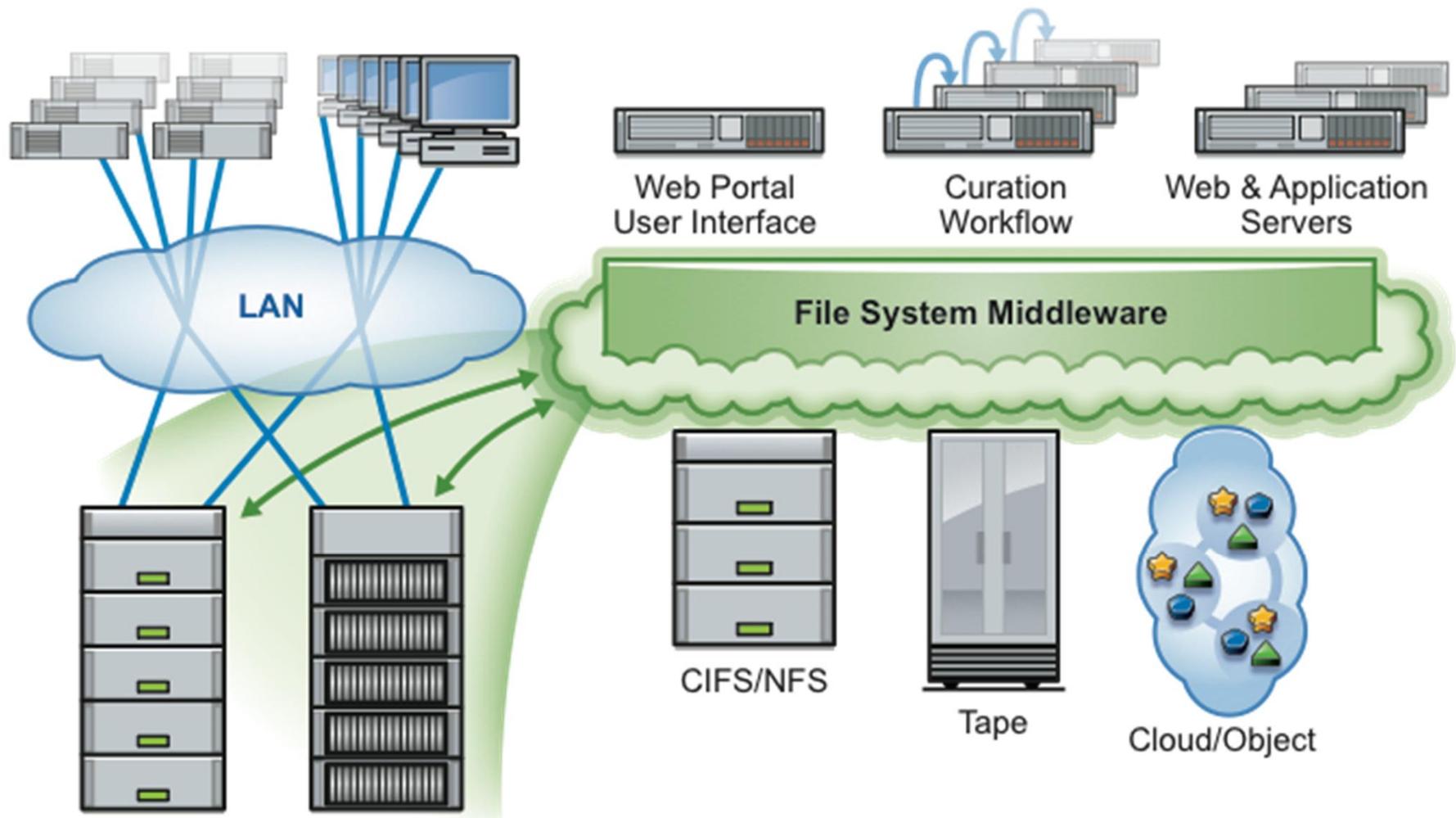
Policy-Based Data Mover and/or Massive Backup System



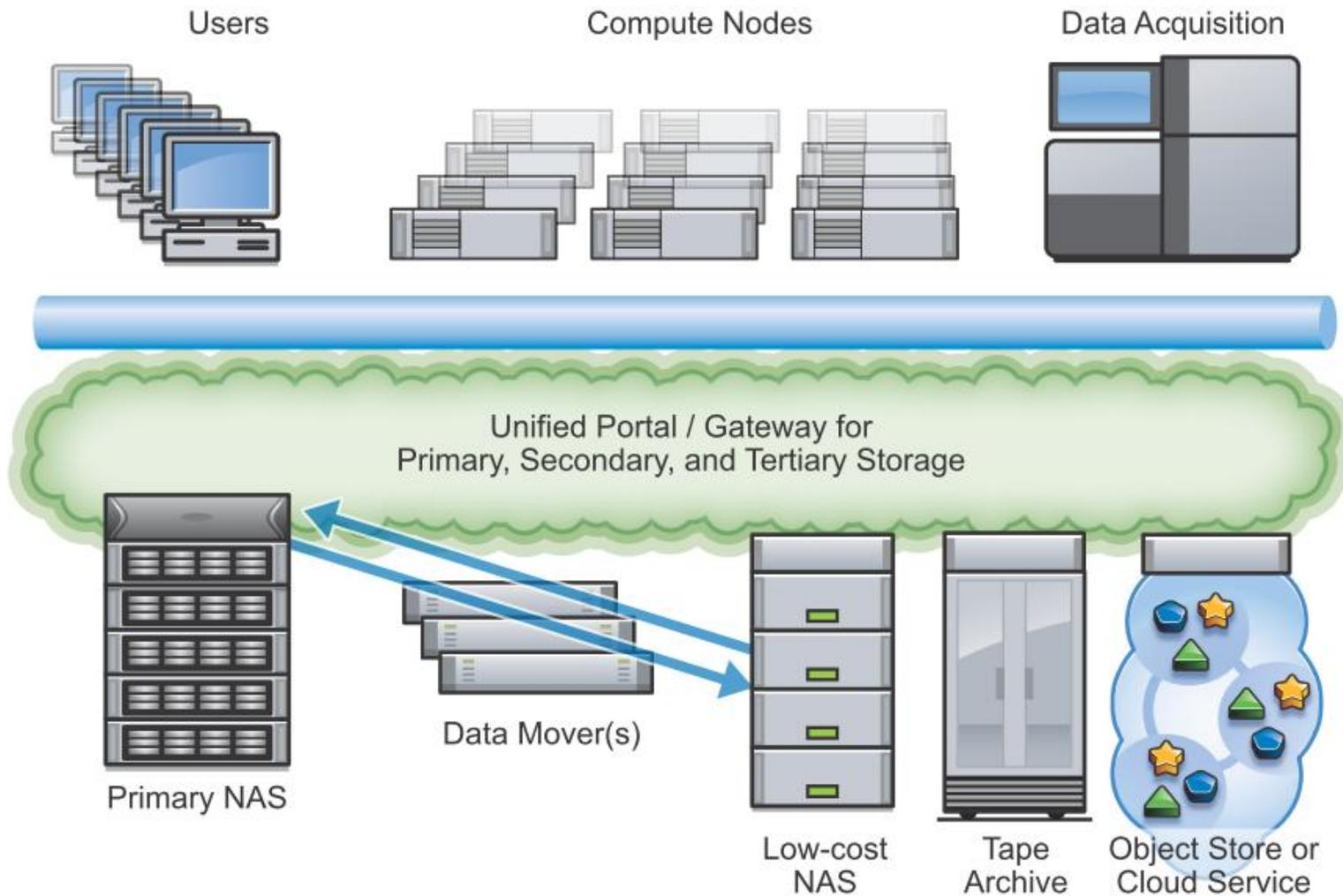
Tiered Storage Management



Hybrid of In-band and Out-of-Band File System Virtualization



A "Virtual" Global File System



Metadata is the Great Enabler



- Collaboration
 - How else would researchers know what to do with one another's data?
 - How can data be organized to meet different groups' needs?
- Storage management policies
 - How does a storage management system know what to do with your files? File system attributes are not descriptive enough.
- Preservation / retrieval / provenance
 - How do you know what to keep?
 - How do you find it again?
 - How do you know what it was used for and when?
- Reporting / chargeback
 - File system permissions are not descriptive enough.



The Real Trick – Getting the Metadata



- The Golden Rule of Data Preservation – “Preserve at the time of creation”
 - Translation: Capture metadata throughout the research pipeline
- Perhaps capture metadata when storage is provisioned
 - The presumes that there is a structured process for provisioning storage
- Capture metadata through an API
 - This requires a simple API that anyone can use
- Programmatically extract metadata from file headers, tags, and content
- Capture metadata through a GUI
 - Try to create incentives for users to key in metadata



Summary of Starfish Vision



- The storage infrastructure needs to know more about the data it stores
 - We need tools to deduce metadata
 - We need a place to put that metadata
- We then need to act on metadata
 - Copy, move, delete, versioning
 - Reporting
 - Search
 - Collaboration frameworks
- Minimize impact on users and applications
 - Make sure that we enable and not hinder them





CAMBRIDGE
Computer
ARTISTS IN DATA STORAGE

Questions – Time Permitting