



Education

# Exploiting multi-tier file storage effectively

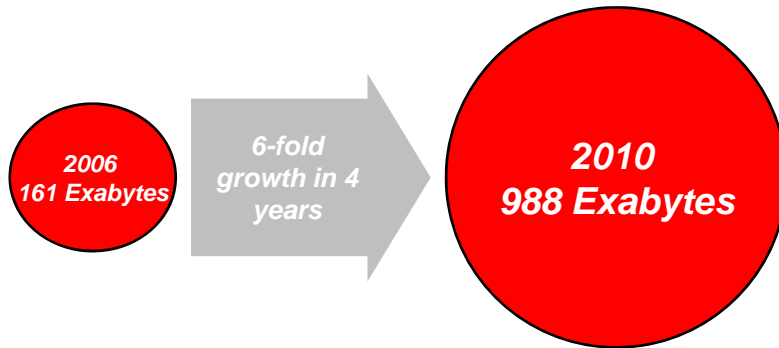
**Paul Massiglia**  
**Symantec Corporation**

- The material contained in this tutorial is copyrighted by the SNIA.
  - Member companies and individuals may use this material in presentations and literature under the following conditions:
    - ◆ Any slide or slides used must be reproduced without modification
    - ◆ The SNIA must be acknowledged as source of any material used in the body of any document containing material from these presentations.
  - This presentation is a project of the SNIA Education Committee.
  - Neither the Author nor the Presenter is an attorney and nothing in this presentation is intended to be nor should be construed as legal advice or opinion. If you need legal advice or legal opinion please contact an attorney.
  - The information presented herein represents the Author's personal opinion and current understanding of the issues involved. The Author, the Presenter, and the SNIA do not assume any responsibility or liability for damages arising out of any reliance on or use of this information.
- NO WARRANTIES, EXPRESS OR IMPLIED. USE AT YOUR OWN RISK.**

## ➤ Exploiting Multi-Tier File Storage Effectively

- ◆ In the past 12 months, a wave of solid state disk introductions has made storage administrator life more interesting. There is now literally a 10-15x cost differential in enterprise storage technologies, and a greater than 100x performance differential. It has become all the more critical to place the right files on the right storage media. This tutorial will survey the various technologies available for file placement and relocation for best effect in a multi-tier storage environment. It will compare HSM, multi-tier file systems, FAN switches and front ends, and manual file placement techniques in terms of efficacy and effort, and contrast them with block level storage tiering strategies. The tutorial should be of interest to storage administrators and others concerned with optimizing online data storage cost and quality of service.

Information created, captured and replicated



• IDC/EMC Study – The Expanding Digital Universe (March 2007)

## ➤ Causes

- Regulatory compliance
  - Redundant & replicated content
- New applications and services
  - Non-text data types
- Online direct access is sooooo much better
- *Because we can*
  - Plummeting raw storage cost

## ➤ Consequences

- The data center is increasingly about data
- Storage cost matters

## Multi-tier storage

### ➤ Part I (the easy part)

- ◆ Differentiated types of online storage
- ◆ Based on delivered storage cost per terabyte
  
- ◆ *But also on storage properties...*
  - › I/O performance
  - › Data reliability
  - › Features

- Consider 100 terabytes of data
  - ◆ 20% active, critical
  - ◆ 80% relatively inactive, backup available

	<b>Storage cost per terabyte</b>	<b>Cost of storage</b>
Single-tier strategy	Tier I: \$7,500	Tier I: \$750,000

- Consider 100 terabytes of data
  - ◆ 20% active, critical
  - ◆ 80% relatively inactive, backup available

	Storage cost per terabyte	Cost of storage	Savings
Single-tier strategy	Tier 1: \$7,500	Tier 1: \$750,000	
Two-tier strategy	Tier 1: \$7,500 Tier 2: \$2,000	Tier 1: \$150,000 Tier 2: \$160,000 <hr/> Total: \$310,000	<b>\$440,000</b> <b>(59%)</b>

- 100 terabytes
  - ◆ 1 terabyte extremely performance-sensitive
  - ◆ 19 terabytes active, critical
  - ◆ 80 terabytes inactive with backup available

	Storage cost per terabyte	Cost of storage	Savings
Single-tier strategy	Tier 1: \$20,000	Tier 1: \$2,000,000	
Three-tier strategy	Tier 1: \$20,000 Tier 2: \$7,500 Tier 3: \$2,000	Tier 1: \$ 20,000 Tier 2: \$142,500 Tier 3: \$160,000 Total: \$322,500	<b>\$1,677,500</b>

# The *rest* of the story

- 100 terabytes
  - ◆ 1 terabyte extremely performance-sensitive
  - ◆ 19 terabytes active, critical
  - ◆ 80 terabytes inactive with backup available

	<b>Available IOPS per terabyte</b>	<b>Total IOPS available</b>
Single-tier strategy	Tier 1: 2,000	Tier 1: 200,000
Three-tier strategy	Tier 1: 2,000 Tier 2: 500 Tier 3: 100	Tier 1: 2,000 Tier 2: 9,500 <u>Tier 3: 8,000</u> Total: 19,500

# The *rest* of the story

- 100 terabytes
  - ◆ 1 terabyte extremely performance-sensitive
  - ◆ 19 terabytes active, critical
  - ◆ 80 terabytes inactive with backup available



Check out SNIA Tutorials:  
*Solid State Storage track*

	Available IOPS per terabyte	Total IOPS available	IOPS required
Single-tier strategy	Tier 1: 2,000	Tier 1: 200,000	Tier 1: 1,000
Three-tier strategy	Tier 1: 2,000 Tier 2: 500 Tier 3: 100	Tier 1: 2,000 Tier 2: 9,500 <u>Tier 3: 8,000</u> Total: 19,500	Tier 1: 1,000 Tier 2: 4,000 <u>Tier 3: 1,000</u> Total: 7,000

## ➤ Compression & deduplication

- ◆ Space-efficient
- ◆ Performance-constrained
- ◆ Data-type specific benefits

## ➤ Encryption

- ◆ More secure against intrusion
- ◆ Performance-constrained
- ◆ Introduces key management issues

## ➤ Disaster tolerance (replication)

- ◆ “Disaster-proofing” for data
- ◆ Data storage is the tip of the disaster tolerance iceberg (with today’s technology and services, at least)

# Back to file storage tiering

## ➤ Part I (easy)

- ◆ Differentiated types of online storage
- ◆ Based on storage
  - › Price
  - › Performance
  - › Reliability

## ➤ Part II (not so easy)

- ◆ Match data value with storage type throughout the data life cycle...*for millions of data objects*

*Automation in some form is a sine qua non*

# Why files, particularly ?

- **Files are the business objects**
  - ◆ Documents
  - ◆ Images
  - ◆ etc.
  
- **Business objects = business value**
  - ◆ Correlating storage cost with data value is straightforward
  
- **Not the be-all and end-all**
  - ◆ File-level tiering cannot easily deal with small “hot spots” in large files

# Implementing multi-tier file storage

1. Classify files
  - ◆ Business value
  - ◆ Access requirements (frequency and consistency)
  
2. Classify storage “devices”
  - ◆ Cost
  - ◆ Performance
  - ◆ Reliability
  
3. Define file placement “policies”
  - ◆ Where to create
  - ◆ When to relocate and to where
  
4. Deploy file relocation technology
  - ◆ File location =  $f(\text{policy}, \text{time})$ ...automatically

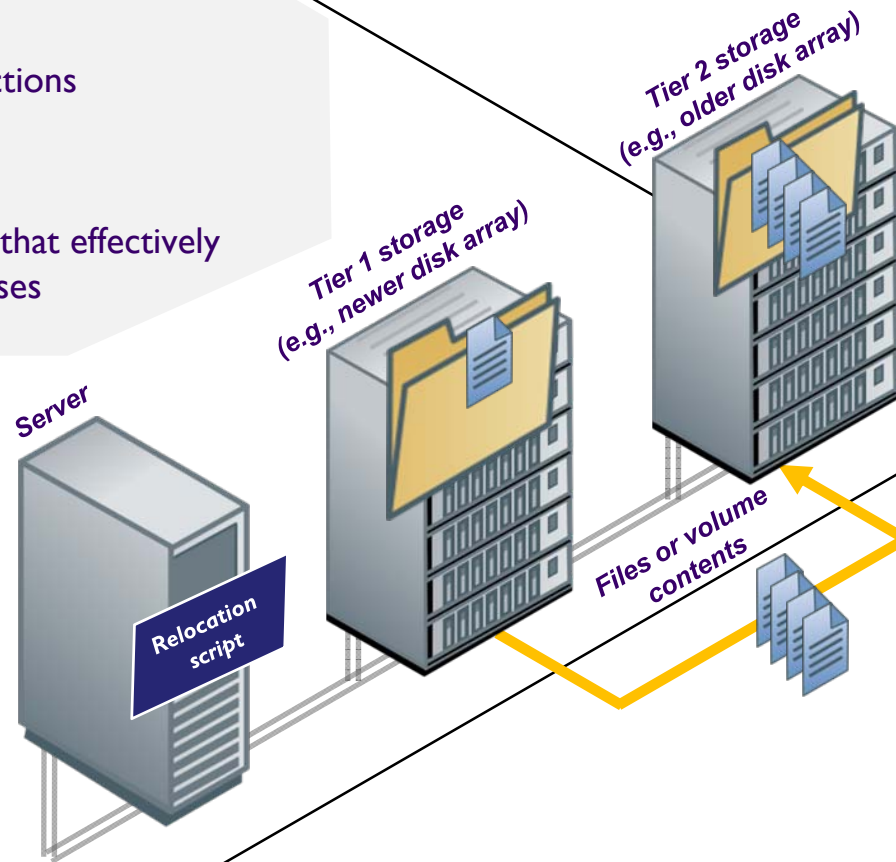
- Ad hoc relocation
- Hierarchical Storage Management (HSM)
- Multi-tier file systems
- File Area Networks (FAN)

# Approach 1: ad hoc data relocation

“When it gets old, move it to less expensive storage”  
e.g.,

- Last month’s transactions
- Project delivered
- etc.

Based on scripts or apps that effectively describe business processes



# ad hoc data relocation

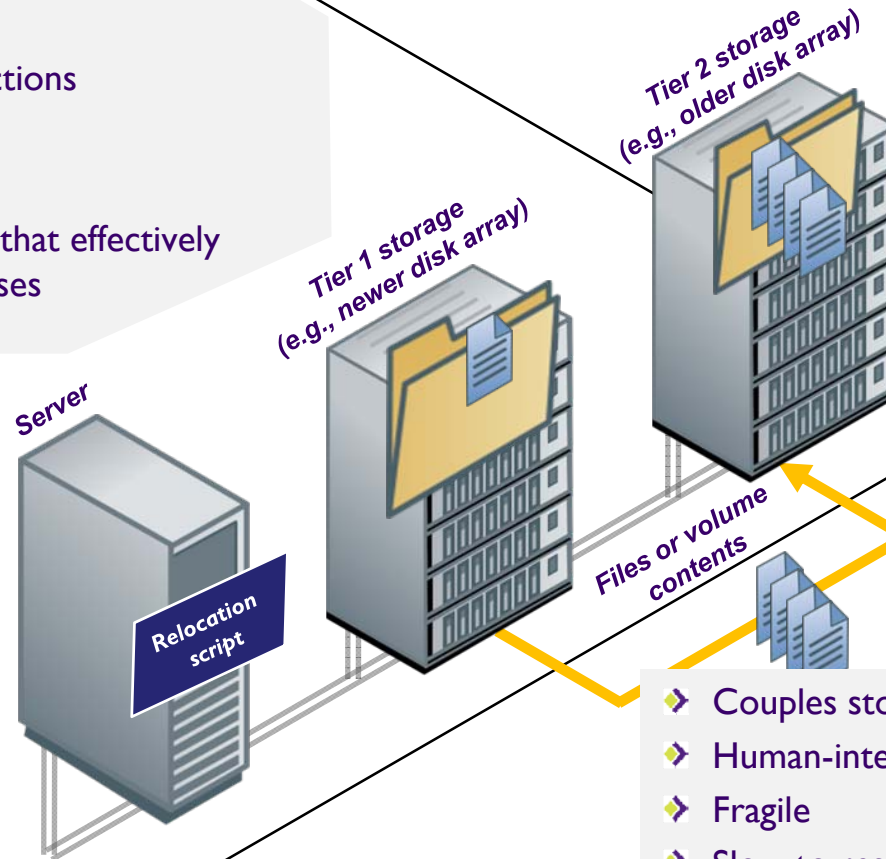
*...the rest of the story...*

“When it gets old, move it to less expensive storage”

e.g.,

- Last month’s transactions
- Project delivered
- etc.

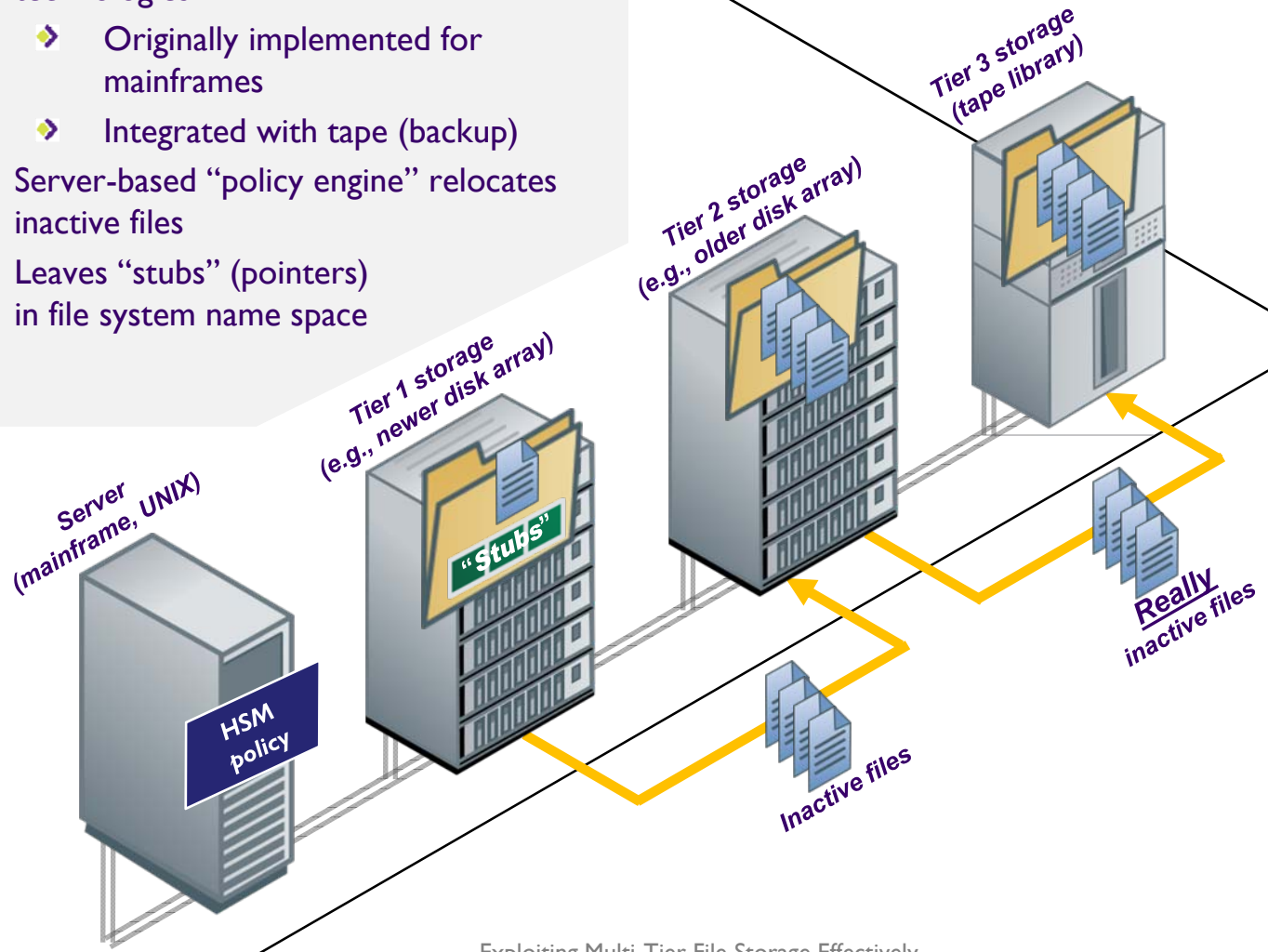
Based on scripts or apps that effectively describe business processes



- Couples storage layout to business requirements
- Human-intensive
- Fragile
- Slow to respond to business process changes
- Upward relocation is typically ad hoc
- Integration with backup and restore processes

# Approach 2: Hierarchical Storage Management (HSM)

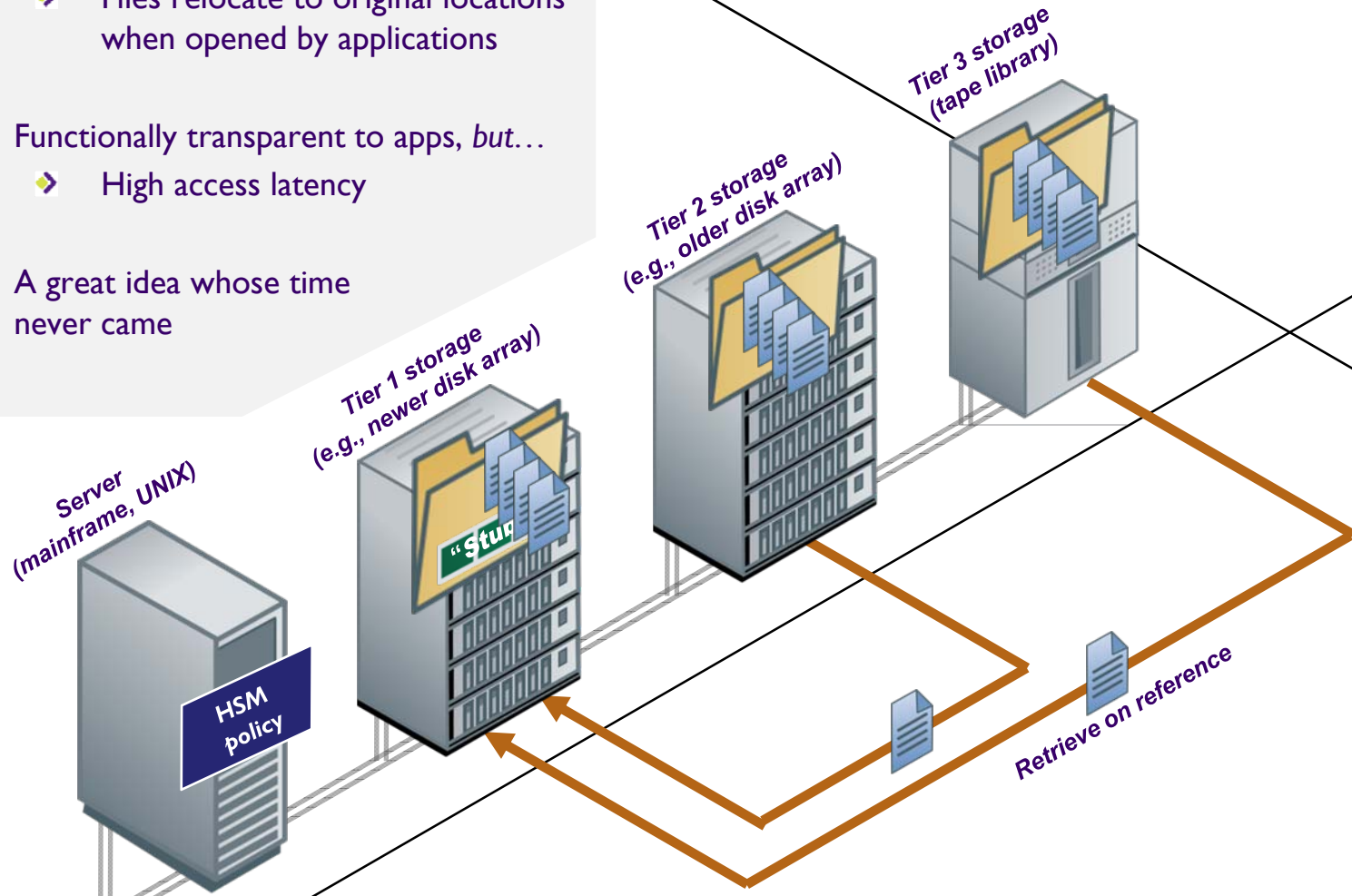
- ▶ The granddaddy of file storage tiering technologies
  - ▶ Originally implemented for mainframes
  - ▶ Integrated with tape (backup)
- ▶ Server-based “policy engine” relocates inactive files
- ▶ Leaves “stubs” (pointers) in file system name space



# Approach 2: Hierarchical Storage Management (HSM)

22 HSM implementations listed at  
[http://en.wikipedia.org/wiki/Hierarchical\\_storage\\_management](http://en.wikipedia.org/wiki/Hierarchical_storage_management)

- “Retrieve on reference”
  - Files relocate to original locations when opened by applications
- Functionally transparent to apps, but...
  - High access latency
- A great idea whose time never came



# Approach 3: Tier-aware file systems

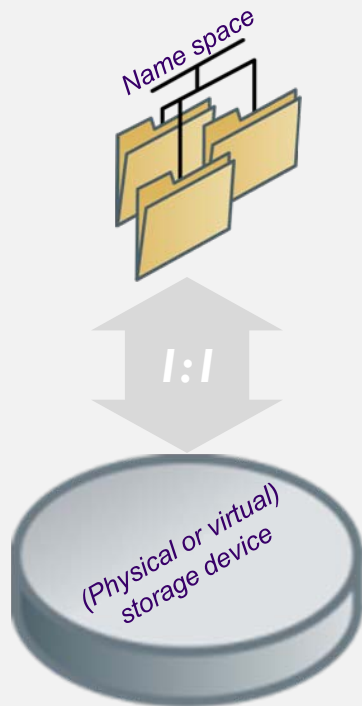
- **Basic concept**
  - ◆ One file system manages two or more storage devices
  
- **Transparent file relocation between tiers**
  - ◆ No stubs
  - ◆ No “time to first byte” retrieval penalty
  
- **Customizable behavior**
  - ◆ Flexible policies
    - › File placement and relocation
    - › File system space management

# Approach 3: Storage tier-aware file system

## File system “classic”

### ◆ Key concepts

- ◆ Underlying storage is homogeneous
- ◆ One name space = one device

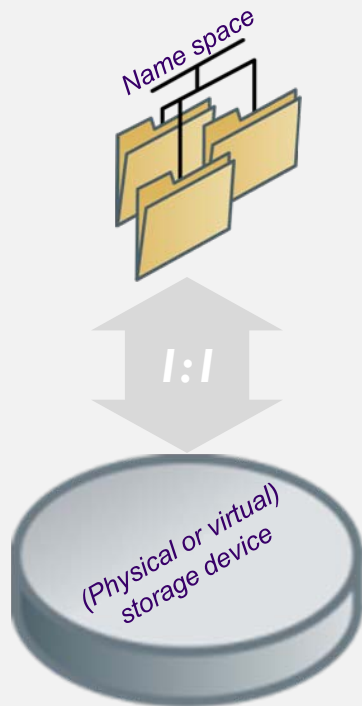


# Approach 3: Storage tier-aware file system

## File system “classic”

### ◆ Key concepts

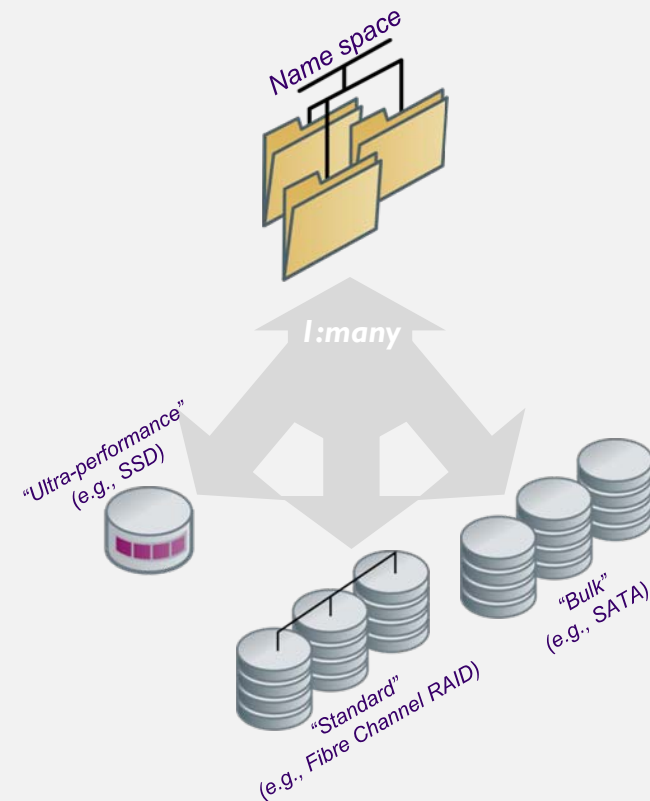
- ◆ Underlying storage is homogeneous
- ◆ One name space = one device



## Storage tier-aware file system

### ◆ Key concepts

- ◆ One name space = multiple devices
- ◆ File system is aware of abstract device “types”



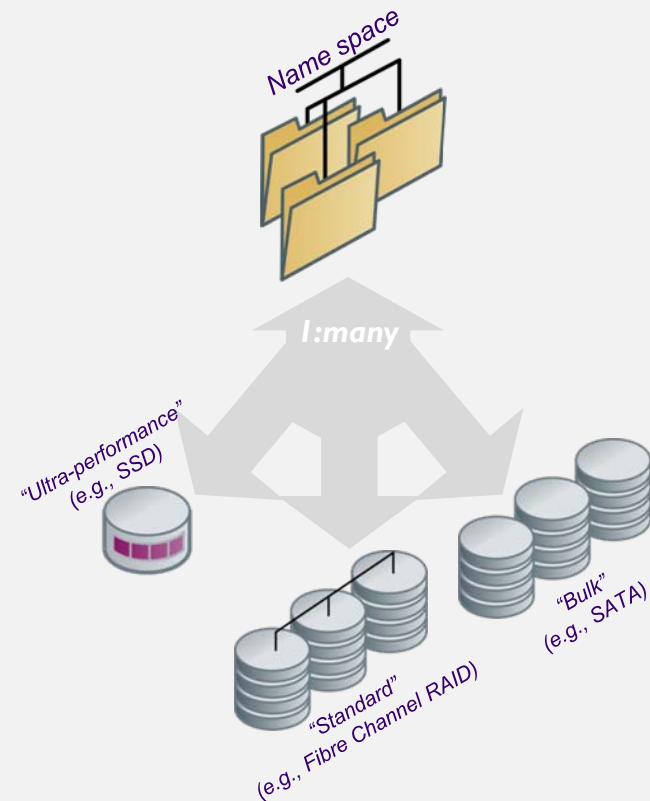
# Approach 3: Storage tier-aware file system

- ◆ Key differentiators from HSM
  - ◆ Control over initial placement as well as movement
  - ◆ Physical movement without logical movement
    - › i.e., files are accessed directly wherever they reside (no “retrieve on reference”)

- ◆ For example
    - ◆ Applications would access /mydir/clip.mp4 as /mydir/clip.mp4 directly whether it resides on
      - › Ultra-performance (Tier1)
      - › Standard (Tier2)
      - › Bulk (Tier3)
- storage

## Storage tier-aware file system

- ◆ Key concepts
  - ◆ One name space = multiple devices
  - ◆ File system aware of abstract device “types”



# Storage tier-aware file system

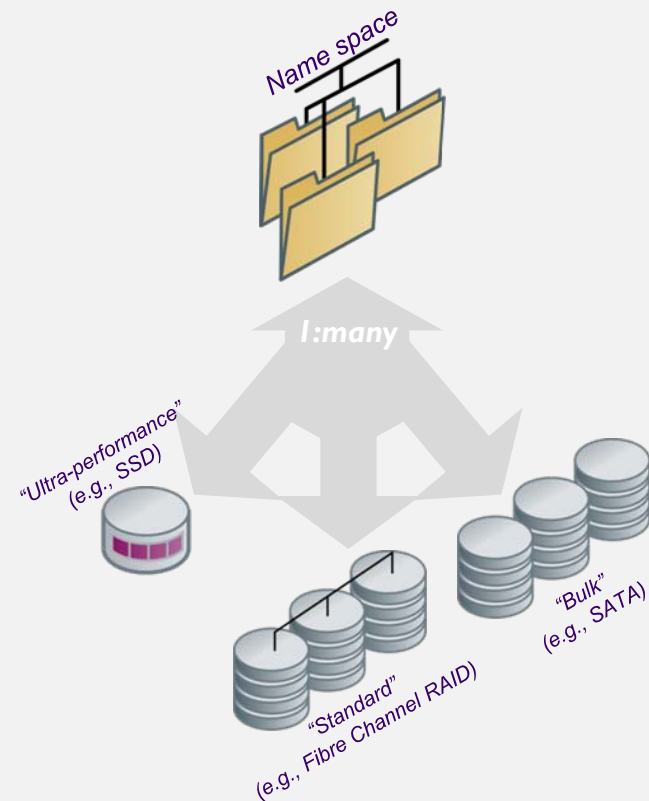
## *the rest of the story*

- A newer concept than HSM
  - ◆ Architected for different use cases
    - › e.g., 24x7 file systems
    - › e.g., for more general storage tiers
    - › e.g., for applications that can't tolerate "recall on reference" delays
  - ◆ Takes advantage of more modern file system capabilities
    - › e.g., abstract (user-defined) storage tiers
    - › e.g., piecewise "move while open"
- Includes automatic "policies"
  - ◆ Which data to consider moving
  - ◆ Circumstances under which to actually move it
  - ◆ Schedule on which to scan candidate files

### Storage tier-aware file system

#### ➤ Key concepts

- ◆ One name space = multiple devices
- ◆ File system aware of abstract device "types"



# Storage tier-aware file system

## *the rest of the story*

### ➤ Abstract storage tiers

- ◆ Alphanumeric storage device 'tags'
  - › e.g., "Tier1", "Platinum", "Gold"...
  - › e.g., "current", "last year",...
  - › e.g., "mirrored", "SATA", etc...
  - › etc.
- ◆ Tier = multiple devices

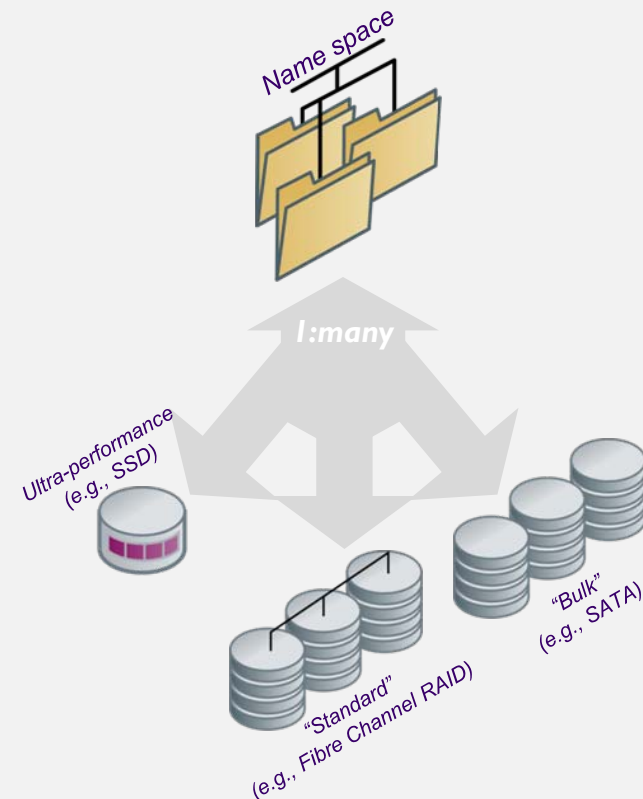
### ➤ Policies

- ◆ Which data to move
  - › e.g., by path & file name
  - › e.g., by ownership
  - › e.g., by current location (tier)
- ◆ Why move it
  - › e.g., was idle
  - › e.g., was reactivated
  - › e.g., grew or shrank
- ◆ When to move it
  - › e.g., Saturday at midnight
  - › etc.

### Storage tier-aware file system

#### ➤ Key concepts

- ◆ One name space = multiple devices
- ◆ File system aware of abstract device "types"

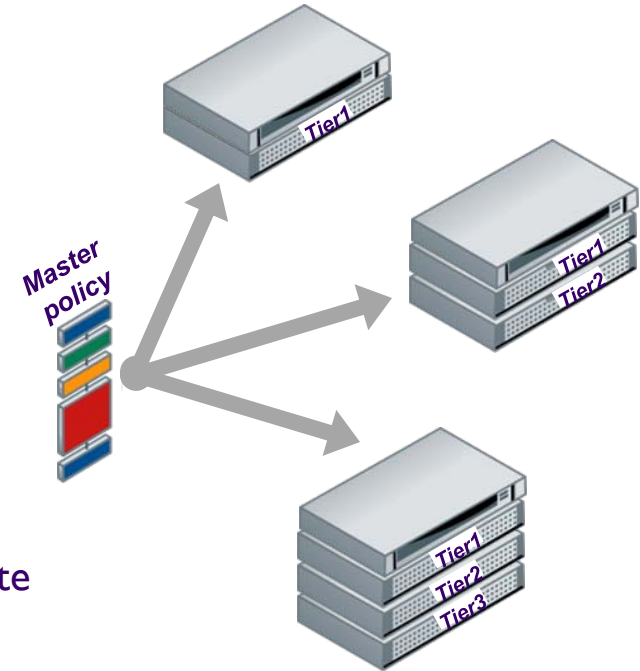


# Storage tier-aware file system

## Ups and downs

### ➤ Ups

- ◆ Greater administrative scope
  - › Abstraction = ‘standard’ policy
  - › ‘Missing’ tier = full tier
  - › Result: master policy to fit an entire data center
- ◆ (Optional) metadata segregation
  - › Partially ‘crippled’ file system can mount and operate



### ➤ Downs

- ◆ Few implementations to choose from (today)
- ◆ Little application integration/awareness
  - › e.g., apps’ ability to operate with missing files
- ◆ “Disk”-only (in-place access = inherently tape-unfriendly)
- ◆ Limited integration with backup managers

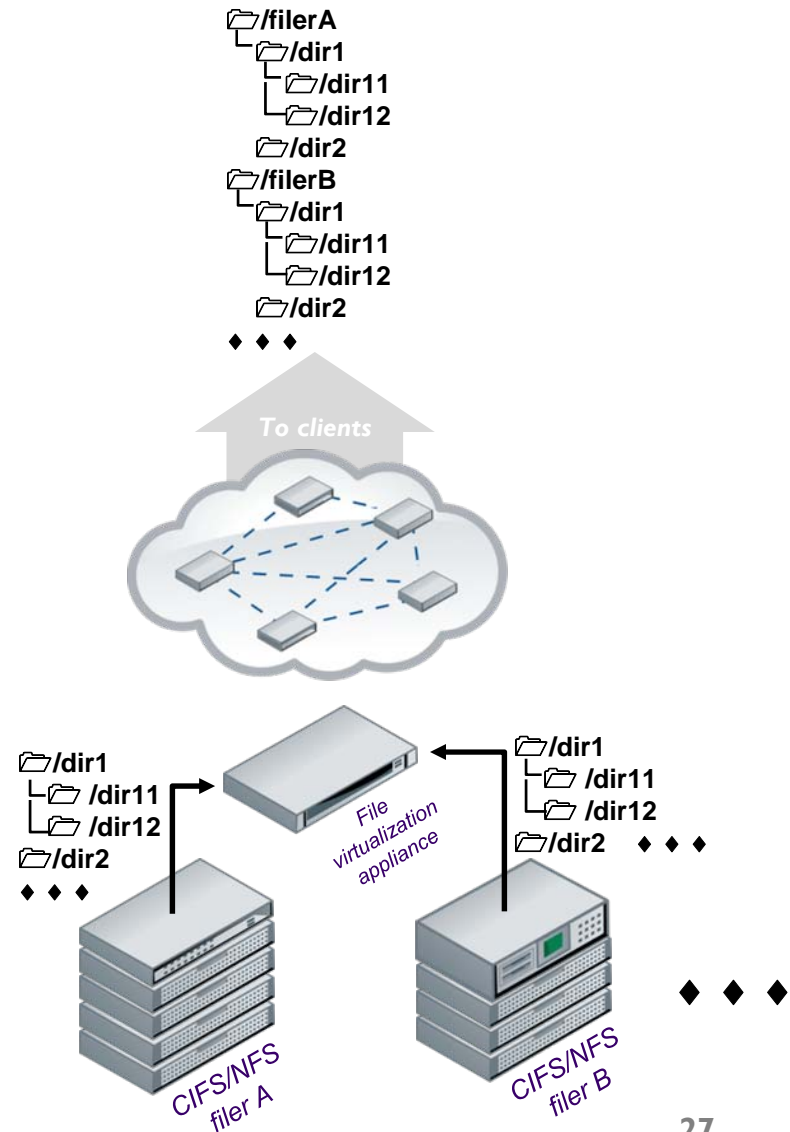
# Approach 4: file area network

## Virtualization appliance

- ◆ Gateway between filers and client access network
- ◆ Presents file systems' name spaces to clients as a single global name space
- ◆ Maps between global name space and file locations

## Multiple applications

- ◆ Multi-tier storage
  - › Archiving
  - › Relocation to minimize average storage cost
- ◆ Data migration
  - › e.g., technology refresh
- ◆ Transparent data replication
- ◆ Cross-filer load balancing



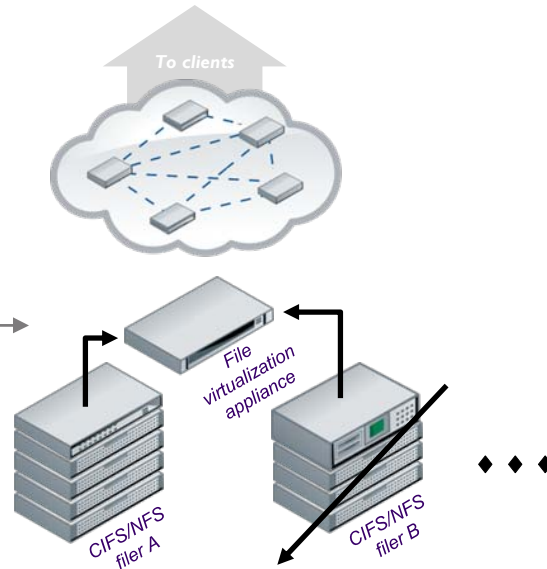
# Approach 4: file area network

## Virtualization appliance models

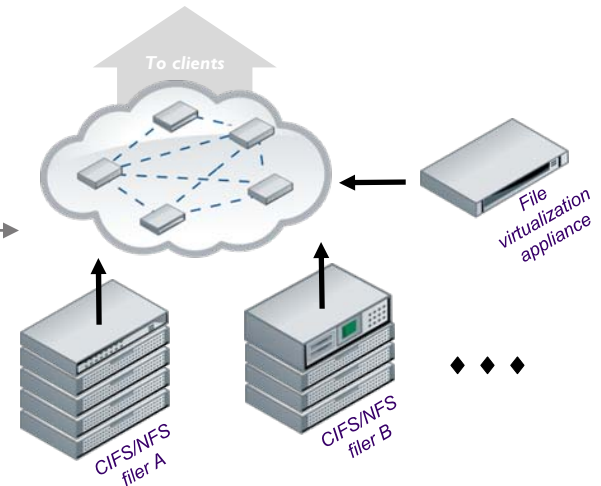
- ◆ Hardware (intelligent switch)
- ◆ Software

## Topologies

- ◆ “In-band”
  - › Store and forward appliance in the data path
  - › Potential bottleneck



- ◆ “Out-of-band”
  - › “Appliance off to the side”
  - › Requires filer awareness



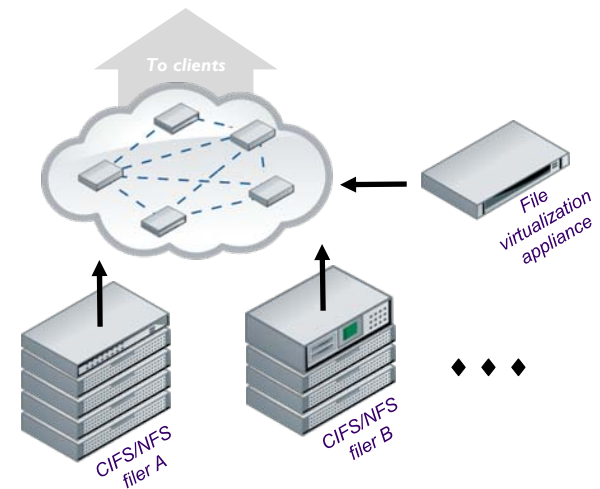
# Approach 4: file area network

## ➤ Archiving (similar to HSM)

- ◆ Move file to archive tier
- ◆ Replace with “stub”
- ◆ “Retrieve on reference”

## ➤ Relocation policies

- ◆ Similar to storage tier-aware file system
  - › Rules for relocation
  - › Periodic scheduling and ad hoc options
- ◆ “What if” previewing
  - › Resulting storage consumption profile
  - › Number of files and bytes to be moved
- ◆ Capacity and performance balancing
  - › Hot spot elimination
  - › Balanced space allocation



# Approach 4: file area network

## *Ups and downs*

### ➤ Ups

- ◆ Integration of heterogeneous NAS filers
- ◆ Name space scaling to billions of files
- ◆ Non-disruptive data migration and relocation
  - › Technology refresh
  - › Blended storage cost containment
- ◆ No increase in average data path length (out-of-band approach)

### ➤ Downs

- ◆ NAS filer-specific
- ◆ File system policy scans can consume significant network bandwidth
  - › More time-consuming than with storage tier-aware file systems
- ◆ Policy manager complexity
  - › Very high internal bandwidth required
  - › Inter-operability with all types of filers
- ◆ Potential bottleneck (in-band implementations)
- ◆ Integration with backup managers is an “exercise for the user”

# What can't be done with file-based storage tier management

## Hot spots within a file

- Poster child example: database table space files
  - ◆ Very large files
  - ◆ Very small hot spots
  
- Dilemma
  - ◆ Move file to lower tier: performance and/or availability suffer
  - ◆ Move file to higher tier: storage cost increases
  
- One solution
  - ◆ Monitor activity within files (e.g., block number ranges)
  - ◆ Relocate individual data block ranges based on I/O activity level
  - ◆ Storage tiering controlled by storage administration (not application administration)

# Summarizing file-level storage tiering

	Application Transparency	“Retrieve on reference”	Policy flexibility	Backup/restore integration	Properties and applications
Ad hoc	No	User-defined	User-defined	None	+Zero capital outlay —Suitable for simple environments stable requirements & configurations —Requires skilled staff
HSM	Yes	Yes	Typically path name access time	Good	—Minimal policy flexibility —Requires application tolerance for “retrieve on reference”
Storage tier-aware file system	Yes	No	Path name, file type, I/O activity, size,	Limited (as yet)	+Per-file system tiering Central policy —Limited availability: Platforms (UNIX, Linux) File systems
File area network	Yes	Some... others are access-in-place	Path name, file type, I/O activity, size,	Case-by-case	+Multi-NAS environments (especially heterogeneous) —Some platform-centricity

- Please send any questions or comments on this presentation to SNIA:  
[trackfilemgmt@snia.org](mailto:trackfilemgmt@snia.org)

**Many thanks to the following individuals  
for their contributions to this tutorial.**

**- SNIA Education Committee**

**Ashvin Kamaraju  
Murthy Mamidi  
Oliver Robinson**

**Thomas Cornely  
Nikhil Raj**