



The Future of Content Storage

Paul Carpentier
CTO, Caringo Inc.

European Microsoft
SharePoint Conference
2007

February 12th - 14th, 2007
Convention Center
Hotel Estrel, Berlin, Germany

 Microsoft
**Office SharePoint
Server 2007**

The Future of **Content** Storage

Dynamic ~ 10% *Databases*

- Large files, few in number
- High-speed, block-level read-write-update
- Concurrent access considerations
- Metric: I/O transactions per second
- SAN/NAS best suited storage architectures

Fixed ~ 90% *Everything but databases*

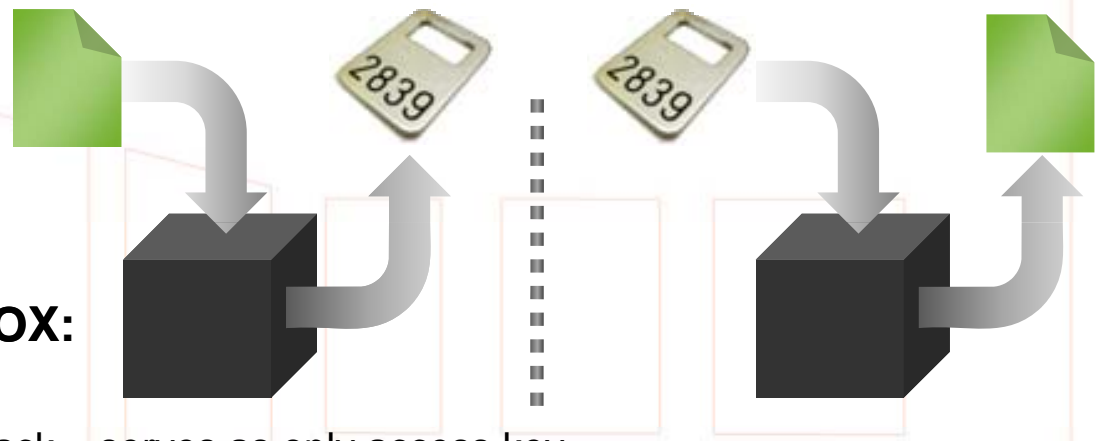
- Large and small files, huge quantities
- File level WORM type access: write, read, delete, never update
- Retention policies for compliance
- No concurrent access considerations
- Metric: objects per second
- **CAS** (Content Addressed Storage) **best suited storage architecture**

My brief history of CAS

- 1995** First concepts of Fixed Content in FileWave software for software distribution on LANs
- 1999** FilePool peer-to-peer Content Addressable Networking, using MD5 hash as unique identifiers
- 1999-2000** FilePool files 4 basic CAS patent applications
- 2001** EMC acquires FilePool
- 2002** EMC launches Centera HW/SW Content Addressed Storage (CAS) system based on FilePool technology
- 2005** Concept confusion as several vendors enter market, all with slightly different interpretations of CAS – some even label it “Content Aware Storage”
- 2006** CAS now \$ 2B market
- 2006** Caringo launches CAStor: pure software appliance, runs CAS on any standard PC commodity hardware



The basic concept of CAS: the “Garderober”



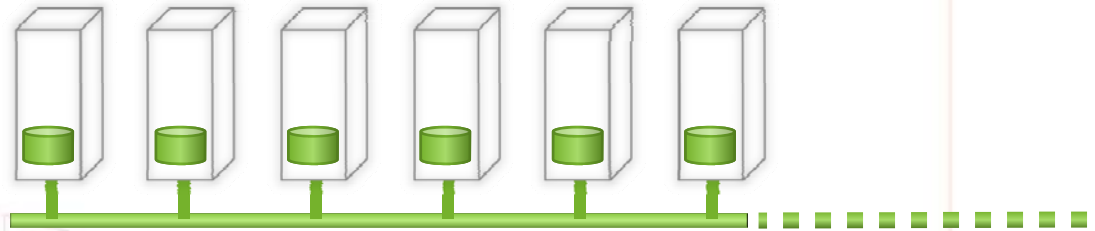
Think of CAS as a **BLACK BOX**:

- Put a file into it
- Get a unique identifier back – serves as only access key
- Keep the identifier: in document or database record
- Provide the identifier to the black box to get the file back
- Yes, it really **is** that simple!

Additional functions desirable for real-world applications:

- Ability to define retention policies and number of desired replicas for each file
- Allow user metadata to be included with each file
- Use a universal interface so any environment can use it

CAS architecture



Typical CAS Cluster of x86-based commodity PCs (rack, tower,...)

- Connected via standard networking (typically Gigabyte Ethernet)
- Containing large capacity disk drives (typically SATA)
- Local replication for redundancy
- Remote replication for disaster recovery
- Self-healing and scalable

Additional characteristics are “battleground” between vendors

- Unique identifiers & protective hash: same or separate?
- SW installation and maintenance of cluster: strong influence on TCO
- HW/SW bundled solution or SW only?
If SW: allow mix of different configurations/vendors?
- Flexibility in node provisioning & decommissioning: down time or not?
- Allow parallel network I/O for throughput?
- Also provide CIFS/NFS access?

CAS identifiers controversy: the finer detail

Most important value of CAS identifiers:

Huge (intergalactic!) flat address space

Scenario 1: Same MD5 hash for integrity AND identifiers

- + Simple
- + Elegant
- + Robust
- Integrity guarantee (longer term) at risk: MD5 “broken” by researchers
- Performance/throughput design limitations

Scenario 2: Separate identifier and integrity hash

- More involved
- + Allows for integrity hash upgrade while maintaining same identifier
- + Performance/throughput design bonus

ILM – Information Lifecycle Management

Traditional scenario – COMPLEX

3 or 4 totally different storage tier infrastructure platforms

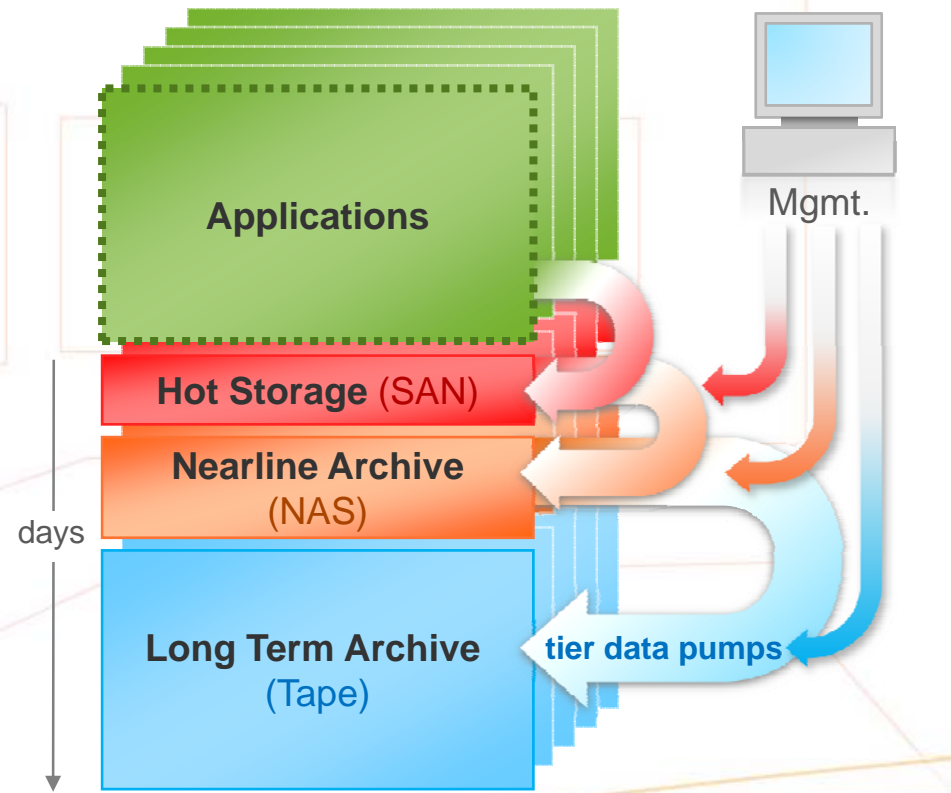
- Overhead costs multiplied
- Runaway data duplication

Separately defined, maintained and managed tier data pumps for each application

- Complex, manual, labor-intensive operations
- Costly & time-consuming management

Fragile & inflexible data environment

Overall economics could improve



“Content Type Management”

Innovative scenario – MUCH SIMPLER

Dynamic content ~ 10% databases → goes into SAN or NAS

- Less capacity needed on expensive high speed storage...
- ... so you can afford top notch servers, specially tuned for database I/O!

Fixed content ~ 90% all the rest → goes into CAS – ideally

- Huge and scalable capacity, high throughput via parallel fabric
- Fast enough for primary storage, cheap enough to keep it there
- No need for data classification

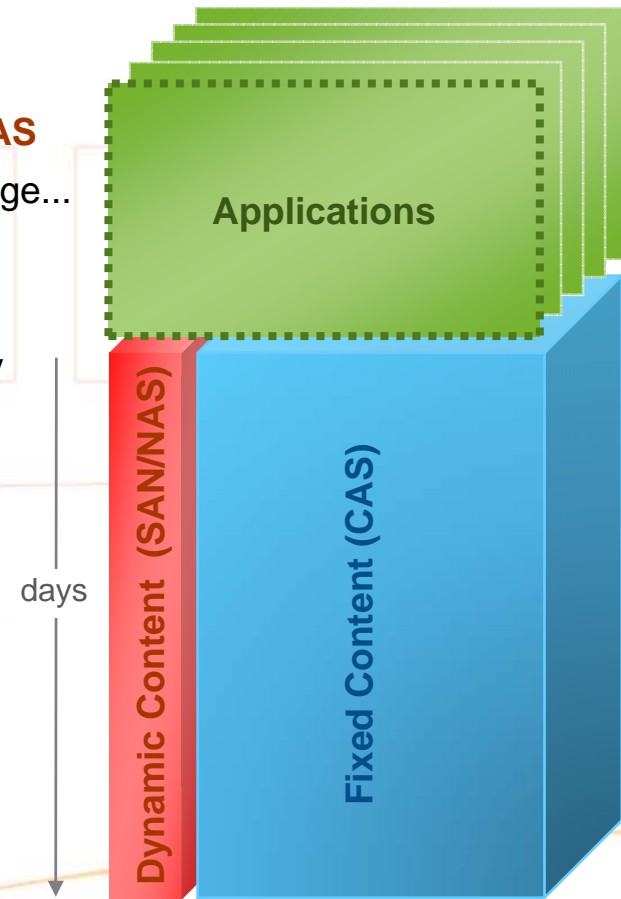
More robust, less expensive

More flexible, faster to implement

- no development and maintenance

Rule of thumb: save at least costs of:

- **Nearline NAS** layer
- **ILM data pumps:** costs of design, development, operation, maintenance



SharePoint on CAS: a good example

Storage mapping layer splits
SharePoint storage requests between:

SQL Server

- Structures
- Folders
- Metadata

CAS cluster

- Documents
- Files
- Media

The total solution is extremely:

Simple

- No tiers, no data pumps

Scalable

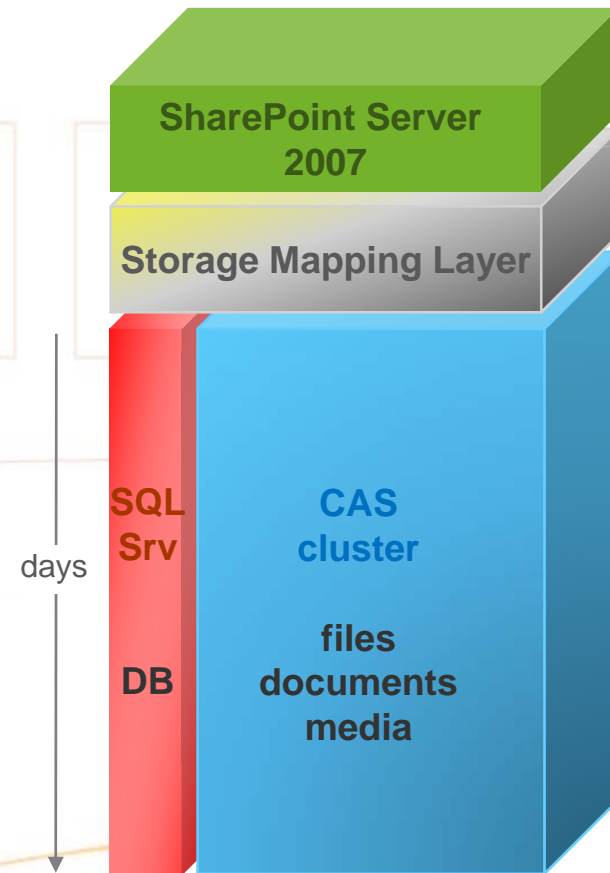
- Add nodes as needed
- 1TB → 1PB → ...

Robust

- Built-in permanent check & backup

Economical

- 2 euro/GB for the bulk of your data!
- Lights out operation, no interventions



Simple Summary

- Fixed Content – “*Everything but databases*” – 90% of your storage!
- Use CAS to split content by type, not by age
- Slash cost, complexity and risk while increasing flexibility
- Hands-off operation, no backup window/headache
- Keep everything online
- Scale up as you go
- Come out **way ahead** in cost and speed
- SharePoint is an excellent target for a CAS implementation

Next Steps

Take a look at your own SharePoint installation/installed base

- Assess your potential TCO savings
- and operations streamlining

Find out more about CAS

- EMC Centera
- HP RISS
- Caringo CAStor
- ...

Take a lesson from SharePoint

- *The world is all about fixed content today*
- maybe it's time to rethink your ILM strategy...