What is digital curation?

Policies and practices for maintaining and adding value to trusted digital content now and into the indefinite future

- Curation is a superset of preservation

PODS (Permanent Objects, Disposable Systems)

- Preservation (curation) is not a place
- Systems come and go (but not *our* system :-)

The new curation landscape

Increasing number, size, and diversity of content, and content producers and consumers

- More stuff, smaller budget

Inevitability of disruptive changes in technology, user expectation, institutional mission, and resources

- "My grant requires a data sustainability plan"
- "I know I should be doing something more to protect my stuff, but I don't know what"
- "I don't want to preserve my stuff, just store it forever"

Assumptions

Curated content gains

- Safety through redundancy
- Meaning through context (description)
- Utility through service
- Value through use (feeds back to safety)

Integrated and decentralized curation can be as effective as centralized curation

Curation stewardship is a relay

The micro-services approach

Want low barrier, low commitment tools

Avoid monolithic, single-culture systems

Compose repository function from small, independent, and interoperable micro-services – complexity *emerges*

True success: micro-services absorbed into the OS infrastructure

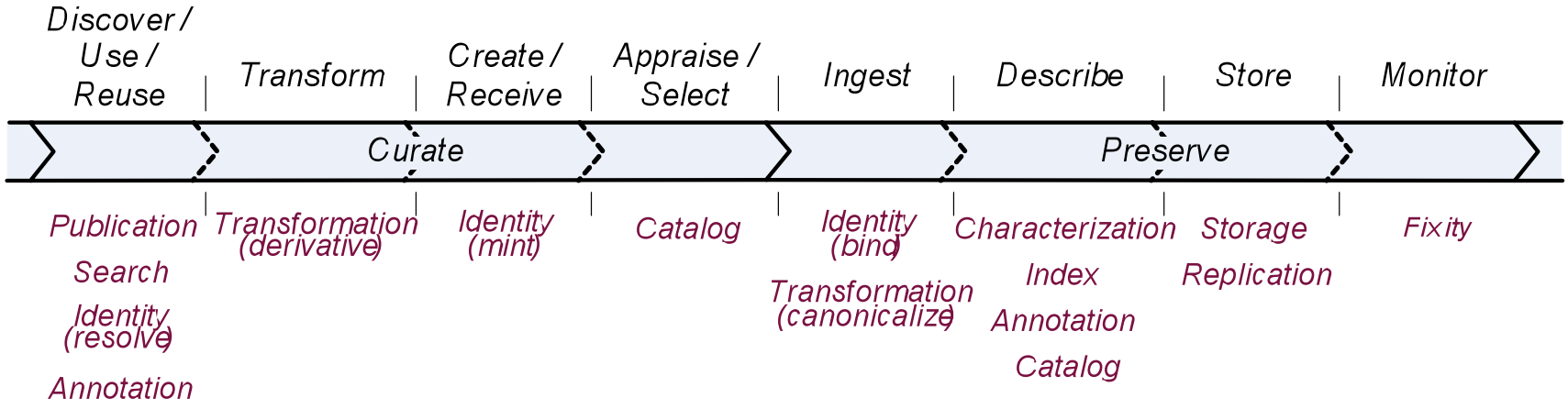
The wisdom of files

- After 30 years, we're *good* at modern filesystems
  - Files and directories (folders) are fast, plentiful, stable, and highly interoperable across platforms
  - Native OS tools will create, list, change, and backup
- File-based micro-services will be easier…
  - to develop, maintain, and to *escape* from
  - to recombine in flexible ways
  - to move upstream into use by content producers

Curation micro-services

Curation	*Interoperation with value*	**Annotation**	Consumer enrichment of managed content
		Publication	Notification of content availability
	Application through service	**Transformation**	Creation of new derivative representations of content
		Search	Index-based search and browse of content and metadata
		Index	Management of indexes of content and metadata
		Ingest	Acquisition of content into a curation environment
Preservation	*Interpretation of context*	**Characterization**	Automated extraction of content properties
		Catalog	Management of structured descriptive metadata
	Protection of state	**Replication**	Synchronization of distributed replicas of content
		Fixity	Verification of bit-level integrity of stored content
		Storage	Secure, persistent storage of digital content
		Identity	Persistent content identification of digital content

Curation throughout the lifecycle



Micro-services can be brought to the content rather than requiring that content be brought to micro-services.

Taking a closer look

What is the thinnest smear of functionality that we can add to a filesystem to make it an effective object store?

- Namaste
- CAN
- Pairtree
- Dflat
- ReDD

Name As Text (Namaste) Tags

Directory-level signature files extending Dublin Core Kernel metadata

– [Magic h0]	`0=name_version`
– Who h1	`1=who`
– What h2	`2=what`
– When h3	`3=when`
– Where h4	`4=where`

Content Access Node (CAN)

File system conventions (structure and reserved names) for an instance of a repository.

```
can/
    0=can_0.01
    can-info.txt
    log/
    store/
        pairtree...
```

Pairtree

Use pairs of object identifier characters to create its file system path.

```
pairtree/
        0=pairtree_0.01
        pairtree-info.txt
        pairtree_root/
                      id/
                         en/
                            ti/
                               fi/
                                  er/
                                     dflat...
```

Dflat

A “digital flat”: a residence for object data and metadata.

```
dflat/
      0=dflat_0.01
      dflat-info.txt
      v001/
           d-manifest.txt
           delta/
                redd...
      v003/
           manifest.txt
           full/
                data/
                metadata/
                enrichment/
                annotation/
```

Reverse Delta Directory (ReDD)

File-level reverse delta compression.

```
redd/
      0=redd_0.01
      add/
      delete.txt
```

Putting a repository together

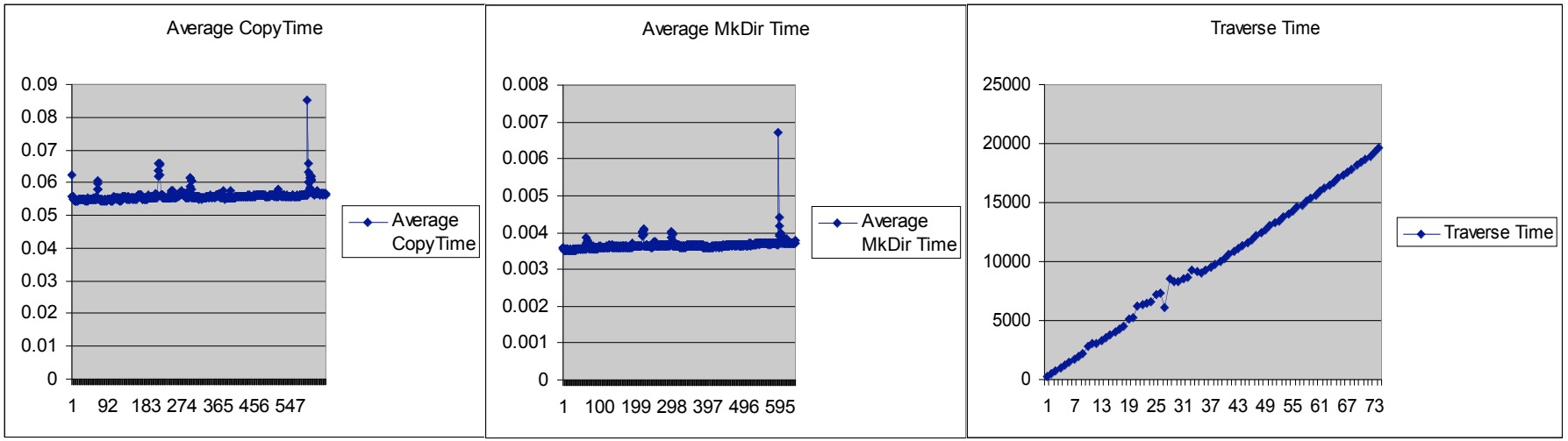
- A *CAN* (content access node) is a repository instance
  - A Pairtree with Dflats for leaves,
  - ReDD-tinged versions, and Namaste tags to greet the visitor who requests a directory listing of the pairpath…

```
$ ls 12/34/5
0=dflat_0.01      admin/
1=Twain,_Mark     v001/
2=Huckleberry..   v002/
3=1898            v003/
4=12345
```

  - 0 = one of {bagit, redd, dflat, pairtree, can, etc.}

Performance Scaling

Modern file systems, e.g. ZFS, exhibit good performance characteristics at reasonable scale



2,272,000 files = 28.5 TB

127,058,820 files = 25.7 TB

Early success story 1

- *Pairtree* (storage service) creates paths from object id/en/ti/fi/er/s, and the resulting directory collection holds objects of any type
  - We invited UM to co-author the Pairtree specification, and Hathi Trust uses our software to store Google books

cyocum

Import a pairtree and you can

- Enumerate all objects and their ids
- Produce any object by requested id
- Maintain and back up the tree with ordinary OS tools
- Rebuild a broken catalog simply by walking the filesystem

Early success story 2

- *BagIt* is a file package (“bag”) suitable for disk-based or fast network-based transfer of generic content
  - We wrote the BagIt specification with the Library of Congress, who now uses BagIt to receive most of its grant-funded partner content

Speaking of recycling, we are building on lots of ongoing success stories:

- JHOVE/2 (characterization service)
- ARK/NOID (identity service)
- XTF (index service)

Our micro-service specifications, and some software, are summarized at

`http://www.cdlib.org/inside/diglib/`

Summary

- Provide
  - Safety through redundancy
  - Meaning through context
  - Utility through service
  - Value through use
- Low commitment leads to high integration
- Complexity through composition, not addition
- Persistent interfaces, evolving implementations
- Early prototyping, frequent refactoring

Questions?

www.cdlib.org/inside/diglib

Stephen.Abrams@ucop.edu
Patricia.Cruse@ucop.edu
John.Kunze@ucop.edu